Table of Contents

Getting Started:

Configuration:

Other Topics:

FAQ

Sections of the FAQ

Licensing, Upgrading, and the Trial Version

Q: What's the difference between the full version of Sawmill and the Trial version?
A: The Trial version is identical to the full version, except that it expires after 30 days. For full details, see **Difference Between Trial and Full**.

Q: What's the difference between Sawmill Enterprise and Sawmill Professional?
A: Enterprise supports MySQL, WebNibbler, multithreaded database builds, and full interface customization. For full details, see **Difference Between Enterprise and Professional**.

Q: When I purchase, do I have to download a new version of Sawmill, or can I "unlock" my existing trial installation?
A: You can unlock your trial installation by entering your license key in the Licensing page. For full details, see **Unlocking a Trial Installation**.

Q: My 30-day trial has expired, and I haven't finished evaluating Sawmill yet. How can I get a new trial?
A: Go to the Licensing page, delete your expired license, and click "Try Sawmill For 30 Days." For full details, see **Resetting the Trial Period**.

Q: How can I upgrade to a new version of Sawmill without losing my profiles, databases, and other data?
A: When upgrading from an older 7.x version to a newer 7.x version (except on Windows), start with the new LogAnalysisInfo and copy files as described in the long answer. On Windows simply install Sawmill over the existing installation. When upgrading from 6.x to 7, copy Configs and run from the command line with -a cc. For full details, see **Upgrading Without Losing Data**.

Major Features

Q: What platforms does Sawmill run on?
A: Windows ME/NT/2000/XP/2003, MacOS, most versions and variants of UNIX. For full details, see **Available Platforms**.

Q: How much memory, CPU power, and disk space do I need to run Sawmill?
A: At least 256 MB RAM, 1 GB preferred; 500 MB disk space for an average database; and as much CPU power as you can get. For full details, see **System Requirements**.

Q: What sorts of log files can Sawmill process?
A: Sawmill can handle all major log formats and many minor formats, and you can create your own custom formats. For full details, see **Supported Log Formats**.

Q: How is Sawmill different from other log analysis tools?
A: Among other things, Sawmill does not generate static reports -- it generates dynamic, interlined reports. For full details, see **Sawmill vs. The Competition**.

Q: How does a typical company use Sawmill; what does a typical Sawmill setup look like?
A: Installations vary from customer to customer--Sawmill provides enough flexibility to let you choose the model that works best for you. For full details, see **Typical Usage Patterns**.

Q: How large of a log file can Sawmill process?
A: There are no limits, except those imposed by the limitations of your server. For full details, see **Processing Large Log Files**.

Q: How can I use a grid (cluster) of computers to process logs faster?
A: Use an internal database, build a separate database on each computer, and merge them. For full details, see

Using a grid of computers to process more data.

Q: Does the log data I feed to Sawmill need to be in chronological order?
A: No; your log entries can be in any order. For full details, see **Log Entry Ordering**.

Installation and Setup

Q: What is a log file?
A: Log files are text files created by your server, recording each hit on your site. Sawmill generates its statistics by analyzing log files. For full details, see **What is a Log File?**.

Q: Can Sawmill be configured to automatically analyze the access log for my site on a shared server once a day at a given time?
A: Yes, if you run it stand-alone, or if your server has a scheduling program. For full details, see **Scheduling**.

Q: I'm running Sawmill on Windows, and it automatically starts itself up on IP 127.0.0.1 and port 8987. How can I tell it to use another IP address and port?
A: Set the Server Hostname option and the Web Server Port option in the Network section of the Preferences. For full details, see **Running on a Different IP**.

Q: How do I see referrer (referring URL, search engines, and search terms), agent (browser and OS), or error statistics?
A: Use "extended" or "combined" log format to see referrer and agent information, or analyze the log files with a separate profile. For error logs, analyze them with a separate profile. For full details, see **Referrer, Agent, and Error Logs**.

Q: Is Sawmill available in languages other than English? How can I change the output of Sawmill to be in a different language, or to use different wording?
A: Sawmill is currently available in English, German, and Japanese, and can be translated into any language fairly easily. Customization of output text is also easy. For full details, see **Language Modules--Localization and Customization**.

Q: Can I set up Sawmill to start automatically when the computer starts up?
A: Yes; run it as a Service on Windows; use StartupItems under MacOS X; use the /etc/rc.d mechanism on UNIX systems that support it. For full details, see **Running Sawmill at System Startup**.

Q: When I run Sawmill in a UNIX terminal window, and then close the window, Sawmill stops working. What can I do about that?
A: Add an ampersand (&) to the end of the command line to run it in the background. For full details, see **Running Sawmill in the Background**.

Q: How can I move the LogAnalysisInfo folder somewhere else?
A: Install Sawmill somewhere else, or make a symbolic link to LogAnalysisInfo, or put the pathname of the new location in the file LogAnalysisInfoDirLoc For full details, see **Relocating LogAnalysisInfo**.

Q: How can I run Sawmill in CGI mode, and still use the Sawmill Scheduler?
A: Use an external Scheduler to run jobs or to call the Sawmill Scheduler, or run Sawmill in both CGI and web server modes. For full details, see **Using the Scheduler with CGI Mode**.

Q: Can Sawmill be configured to automatically FTP log files from multiple servers, and add them daily to a database?
A: Yes. For full details, see **Downloading Log Data by FTP**.

Q: Can Sawmill use scp, or sftp, or ssh, or https, to download log data? Can I download a whole directory of files via HTTP? Can it uncompress tar, or arc, or sea, or hqx, etc.?
A: Not directly, but you can do it by using a command-line log source to run a command line, script, or program that does whatever is necessary to fetch the data, and prints it to Sawmill. For full details, see **Using a Command-line Log Source**.

Q: Can I run Sawmill as a Service on Windows? Can I run Sawmill while I'm logged out?
A: As of version 7, Sawmill is installed as a service when you run the normal installer. For full details, see **Running Sawmill as a Service**.

Q: My web site is hosted in another state. Does Sawmill provide browser based admin tools I can use to configure it and retrieve reports?
A: Yes, Sawmill's interface is entirely browser based. For full details, see **Remote Administration**.

Q: Can Sawmill generate separate analyses for all the websites hosted on my server?
A: Yes, Sawmill includes a number of features for just this purpose. For full details, see **Statistics for Multiple Sites**.

Q: Can Sawmill process ZIPped, gzipped, or bzipped log data?
A: Yes, all three. For full details, see **Processing zipped, gzipped, or bzipped Log Data**.

Q: Can Sawmill combine the logs from multiple clustered or load balanced web servers, so that the user has one view

of the data? Can it report separately on the different servers?
A: Yes. For full details, see **Clustered Servers**.

Q: Can Sawmill be configured to limit access to statistics, so that a customer can only see the statistics associated with their section of my website?
A: Yes, you can password protect statistics in several ways. For full details, see **Protecting Clients' Statistics**.

Q: I want to deploy Sawmill to my customers, but I want it to look like part of my site. I don't want the name Sawmill to appear -- I want my own name to appear. Can I relabel or white-label Sawmill?
A: Yes, but the degree to which you can relabel depends on your license. For full details, see **Relabeling/White-labeling Sawmill**.

Q: What features can I use in Sawmill's regular expressions?
A: You can use whatever's documented (**Regular Expressions**), and possibly more. How much more you can use depends on your platform. For full details, see **Regular Expression Features**.

Q: Are Sawmill's regular expressions case-sensitive?
A: Yes. For full details, see **Regular Expression Case-sensitivity**.

Q: How can I debug my custom log format, or my log filters?
A: Build the database from the command line with the -v option: `SawmillCL.exe -p` *`profilename`* `-a bd -v egblpfd`. For full details, see **Using Debugging Output**.

Log Filters

Q: How can I exclude hits from my own IP address, or from my organization's domain?
A: Add a Log Filter to exclude those hits. For full details, see **Excluding an IP Address or Domain**.

Q: How can I throw away all the spider hits, so I only see statistics on non-spider hits?
A: Use a Log Filter to reject all hits from spiders (and worms). For full details, see **Discarding hits from spiders**.

Q: Can Sawmill generate statistics on just one domain, from a log file containing log data from many domains?
A: Yes. Add a log filter that rejects hits from all other domains. For full details, see **Filtering All but One Domain**.

Q: How can I remove a particular file or directory from the statistics?
A: Use a log filter to reject all hits on that file or directory. For full details, see **Excluding a File or folder**.

Q: How can I group my events in broad categories (like "internal" vs. "external" or "monitoring" vs. "actual"), and see the events on each category separately, or see them combined? How can I create content groups? How can I include information from an external database in my reports, e.g., include the full names of users based on the logged username, or the full names of pages based on the logged URL? How can I extract parts of the URL and report them as separate fields?
A: Create a new log field, database field, report and report menu item to track and show the category or custom value, and then use a log filter to set the log field appropriately for each entry. For full details, see **Creating Custom Fields**.

Q: How do I remove fields from the database to save space?
A: Delete the database.fields entry from the profile .cfg file, and delete any xref groups and reports that use it. For full details, see **Removing Database Fields**.

Q: Most of the referrers listed in the "Top referrers" view are from my own site. Why is that, and how can I eliminate referrers from my own site from the statistics?
A: These are "internal referrers"; they represent visitors going from one page of your site to another page of your site. You can eliminate them by modifying the default "(internal referrer)" log filter, changing http://www.mydomain.com/ in that filter to your website URL. For full details, see **Eliminating Internal Referrers**.

Q: I use parameters on my pages (e.g. index.html?param1+param2), but Sawmill just shows "index.html?(parameters)." How can I see my page parameters?
A: Delete the Log Filter that converts the parameters to "(parameters)." For full details, see **Page Parameters**.

Q: How can I see just the most recent day/week/month of statistics?
A: Use the Calendar, or the Filters, or use a `recentdays`filter on the command line. For full details, see **Recent Statistics**.

Q: How can I combine referrers, so hits from http://search.yahoo.com, http://dir.yahoo.com, and http://google.yahoo.com are combined into a single entry?
A: Create a log filter converting all the hostnames to the same hostname. For full details, see **Combining Referring Domains**.

Q: How can I debug my custom log format, or my log filters?
A: Build the database from the command line with the -v option: `SawmillCL.exe -p` *`profilename`* `-a bd -v egblpfd`. For full details, see **Using Debugging Output**.

Q: When I look at the top hosts and top domains, all I see are numbers (IP addresses). How do I get the domain

information?
A: Turn on reverse DNS lookup in the Network options (or in your web server), or use Sawmill's "look up IP numbers" feature. For full details, see **Resolving IP Numbers**.

Q: Can I configure Sawmill to recognize search engines other than the ones it knows already?
A: Yes -- just edit the search_engines.cfg file in the LogAnalysisInfo folder with a text editor. For full details, see **Adding Search Engines**.

Q: My server logs times in GMT or UTC, but I'm in a different time zone. How can I get the statistics in my own time zone?
A: Set the date_offset option in the profile. For full details, see **Changing the Time Zone in Statistics**.

Reports

Q: What are "hits"? What are "page views"? What is "bandwidth"? What are "visitors"? What are "sessions"?
A: Hits are accesses to the server; page views are accesses to HTML pages; visitors are unique visitors to the site, and sessions are visits to the site. For full details, see **Hits, Visitors, etc.**.

Q: My website uses dynamic URLs instead of static pages; i.e., I have a lot of machine-generated URLs that look like /file?param1=value1&param2=value2.... Can Sawmill report on those?
A: Yes, but you need to delete the "(parameters)" log filter first. For full details, see **Dynamic URLs**.

Q: There's a line above some of the tables in the statistics that says, "parenthesized items omitted." What does that mean?
A: It means that some items (probably useless ones) have been omitted from the table to make the information more useful--you can show them by choosing "show parenthesized items" from the Options menu. For full details, see **Parenthesized Items Omitted**.

Q: In my reports, I see entries for /somedir/, and /somedir/{default}, and /somedir, and /somedir/ (default page). What's the difference? I seem to have two hits for each hit because of this; one on /somedir and then one on /somedir/; what can I do to show that as one hit?
A: /somedir/ is the total hits on a directory and all its contents; /somedir is an attempt to hit that directory which was directed because it did not have the trailing slash; and the default page ones both indicate the number of hits on the directory itself (e.g., on the default page of the directory). For full details, see **Default Page Hits**.

Q: How do I see the number of downloads for a particular file, i.e., a newsletter PDF, or a template file PDF?
A: Select PDF from the 'File Types' table and then use the Zoom Menu to zoom to the URL's report, then Select the PDF you need to get an overview of that file. For full details, see **Zooming on single files**.

Q: How do I see more levels of statistics, i.e., how can I zoom in further?
A: Increase the "suppress below" level for this database field in the profile options. For full details, see **Zooming Further**.

Q: Can I see the number of hits per week? Can I see a "top weeks" report?
A: Yes, by using the Calendar, and/or creating a database field and a report tracking "weeks of the year." For full details, see **Weekly Statistics**.

Q: Can Sawmill count unique visitors?
A: Yes, using unique hostname or using cookies. For full details, see **Unique Visitors**.

Q: Can Sawmill count visitors using cookies, rather than unique hostnames?
A: Yes -- it includes a built-in log format to do this for Apache, and other servers can be set up manually. For full details, see **Counting Visitors With Cookies**.

Q: Can Sawmill show me the paths visitors took through my web site?
A: Yes; its "session paths (clickstreams)" report is very powerful. For full details, see **Clickstreams (Paths Through the Site)**.

Q: I want to track conversions-- i.e. I want to know which of my ads are actually resulting in sales. Can Sawmill do that?
A: Yes -- encode source information in your URLs and use global filters to show the top entry pages for your "success" page. For full details, see **Tracking Conversions**.

Q: How can I see the top (insert field here) for each (insert field here)? For instance, how can I see the pages hit by a particular visitor? Or the top visitors who hit a particular page? Or the top referrers for a particular day, or the top days for a particular referrer? Or the top search phrases for a search engine, the top authenticated users for a directory, the top directories accessed by an authenticated user, etc.?
A: Click on the item you're interested in, and chose the other field from "default report on zoom". For full details, see **Finding the Top (field) for a Particular (field)**.

Q: How can I only see the visitors that entered at a particular page, or only the visitors that hit a particular page at some point in their session?
A: Use the global filters to show only sessions containing that page; reports will only show sessions including that page. For full details, see **Sessions For A Particular Page**.

Q: How can I see only the visitors that came from a particular search engine?
A: Direct that search engine to a particular entry page, and then use global filters to show only sessions for that page. For full details, see **Sessions For A Particular Search Engine**.

Q: Why doesn't the number of visitors in the Overview match the number of session users in the "Sessions Overview" report?
A: Session information only shows users contributing page views, and other views show all visitors. Also, long sessions are discarded from the session information. For full details, see **Visitors vs. Session Users**.

Q: How can I see just the most recent day/week/month of statistics?
A: Use the Calendar, or the Filters, or use a `recentdays`filter on the command line. For full details, see **Recent Statistics**.

Q: Why do my emailed reports from Outlook 2003 not line up, everything is out of alignment?
A: Change the settings in Outlook to not load content automatically. For full details, see **Emailed Reports in Outlook 2003**.

Q: Can I export the data from Sawmill reports to Excel or other programs?
A: Yes; click the "export" link in the toolbar above reports to export the data from that report's table in CSV format. Many programs, including Excel, can import CSV format files. For full details, see **Exporting Data From Statistics**.

Q: I've heard that statistics like visitors, "sessions," and "paths through the site" can't be computed accurately. Is that true? Are the statistics reported by Sawmill an accurate description of the actual traffic on my site?
A: Sawmill accurately reports the data *as it appears in the log file*. However, many factors skew the data in the log file. The statistics are still useful, and the skew can be minimized through server configuration. For full details, see **Are the Statistics Accurate?**.

Q: How does Sawmill compute session information, like total sessions, repeat visitors, paths through the site, entry pages, exit pages, time spent per page, etc.?
A: Sawmill uses the visitor id field to identify unique visitors. It decides that a new session has begun if a visitor has been idle for 30 minutes. It rejects sessions longer than 2 hours. For full details, see **Session Computation**.

Q: How do I change the field which is already graphed, e.g., from page view to bandwidth?
A: Edit the profile .cfg file, and change the field name in the numerical_fields section of that report element. For full details, see **Changing the graph field**.

Q: Does Sawmill do "peak period" reports (by weekday, or hour)?
A: Yes. For full details, see **Peak Period Reports**.

Q: Does Sawmill do time of day?
A: Yes. For full details, see **Time of Day Statistics**.

Q: How can I tell where visitors went when they left the site?
A: Normally, you can't. However, you can set up "reflector" pages if you need this information. For full details, see **Tracking Exit URLs**.

Q: How can I see *all*files that were hit on my website, not just the pages?
A: Delete or disable the 'Strip non-page-views' log filter, and rebuild the database For full details, see **Showing All Files**.

Q: Why do I see hits on a file called "robots.txt" in my statistics?
A: robots.txt is a file that tells search engine spiders and robots what they can do, so a hit on robots.txt means that a spider visited your site. For full details, see **robots.txt**.

Q: Why do I see a hits on a file called "favicon.ico" in my statistics?
A: favicon.ico is a special icon file that Internet Explorer looks for when it first visits the site. For full details, see **favicon.ico**.

Q: How can I add additional columns to report tables, e.g., to add a single report which reports source IP, destination IP, source port, and destination port?
A: Edit the report in the profile .cfg file to add a new item to the columns group. For full details, see **Adding columns to report tables**.

Q: Does Sawmill produce reports for HIPAA and Sarbanes-Oxley (SOX) compliance?
A: Yes, run the Single-Page Summary report. For full details, see **Support for HIPAA and Sarbanes-Oxley Compliance**.

Troubleshooting

Q: When I run Sawmill, it tells me that the server is started (it shows me the URL), but when I try to access that URL, the browser says it's not available. How can I fix this?
A: You may be using a proxy server which prevents you from accessing a server running on your own machine. Try

reconfiguring the proxy to allow it, or try running Sawmill on IP 127.0.0.1 (the loopback interface). For full details, see **Can't Access the Server**.

Q: On Windows 2003, I can't access the Sawmill server using Internet Explorer. Why not?
A: The "Internet Explorer Enhanced Security Configuration" may be enabled, blocking access; uninstall it or add 127.0.0.1:8987 to the trusted sites. For full details, see **Can't access server with Windows 2003 and IE**.

Q: When I try to log in to Sawmill, I get to the Admin page, but the next thing I click on takes me back to the login page. Why?
A: Your browser isn't storing the cookie Sawmill needs to maintain the login, or something is blocking the browser from sending the cookie. Make sure cookies are enabled in the browser, firewalls aren't blocking cookies, and don't use Safari 1.2.1 or earlier as your browser. For full details, see **Login Loops Back to Login**.

Q: Why can't Sawmill see my mapped drive, share, directory, or mount points when I run it as a Windows Service?
A: The Service must run with the same privileged user account that has the mapped drive, share, directory, or mount point privilege. For full details, see **Can't See Network Drives with Sawmill as Service**.

Q: Why can't Sawmill see my mapped drive, share, directory, or mount points when I run it under Windows 2003?
A: Windows 2003 has a strict security policy which prevents access to network drives from Sawmill. To make it work, you need to let "everyone" permissions apply to anonymous, and remove the restriction on anonymous access to named pipes and shares (in Administrative Tools). For full details, see **Can't See Network Drives in Windows 2003**.

Q: When I build or update my database with Sawmill, it uses a huge amount of memory. Then, when I view statistics, it's very slow. What can I do about that?
A: Decrease the complexity of the database. For full details, see **Sawmill uses too much memory for builds/ updates, and is slow to view**.

Q: I get an error 'Unable to allocate *N*bytes of memory' while building a database, and Sawmill seems to have used all my available memory. What can I do about it?
A: Use a MySQL database, and/or use a 64-bit computer and operating system, and/or simplify your database For full details, see **Database Memory Usage**.

Q: I can't access Sawmill where I usually do (http://*www.xxx.yyy.zzz*:8987/) -- is your (Flowerfire's) server down?
A: No -- *your*server is down. Sawmill runs on your computer, not on ours -- contact your network administrator if you're having problems accessing it. For full details, see **Sawmill Server is Down**.

Q: Sawmill displays the following error: "The background process terminated unexpectedly, without returning a result." What does that mean, and how can I fix it?
A: Sawmill has probably crashed, so this could be a bug in Sawmill. See the long answer for suggestions. For full details, see **The background process terminated unexpectedly**.

Q: When I run Sawmill on Windows, I get an error: "A required DLL is missing: WS2_32.DLL." What's going on?
A: You need **Winsock 2**. For full details, see **Winsock 2**.

Q: When I run Sawmill on Windows 98, I get an error: "A required DLL is missing: OLEACC.DLL." What's going on?
A: You need to download and install the latest **Service Pack**for Windows 98. For full details, see **Missing DLL: OLEACC.DLL**.

Q: When I run Sawmill on Windows, I get an error: "A required DLL is missing: URLMON.DLL." What's going on?
A: Install the latest Internet Explorer, and the problem should go away. For full details, see **Missing DLL: URLMON. DLL**.

Q: When I run Sawmill, I get an error: './sawmill: error while loading shared libraries: libstdc++.so.5: cannot open shared object file: No such file or directory'. What's going on?
A: Sawmill requires the libstdc++ library. This is available by default on many platforms, and is included in the Sawmill distribution on others (including Solaris) For full details, see **libstdc++ missing**.

Q: When I try to run Sawmill, I get an error "relocation error: sawmill: undefined symbol: __dynamic_cast_2". How can I fix this?
A: This is a GNU library incompatibility; build Sawmill from source instead of using the binary distribution. For full details, see **Relocation error: __dynamic_cast_2**.

Q: Sawmill only shows me the IP addresses of my visitors, even when I turn on DNS lookup. Why?
A: Try deleting the IPNumbersCache file in LogAnalysisInfo -- see the long answer for other solutions. For full details, see **Problems With DNS Lookup**.

Q: I run Sawmill in CGI mode, and all the images in the menus and the reports are missing or broken. Why?
A: You may have set the "temporary folder" incorrectly during installation. Try deleting the preferences.cfg file in LogAnalysisInfo, and access Sawmill to try again. For full details, see **No Images in CGI Mode**.

Q: The statistics show the wrong years -- when I analyze data from previous years, it appears as this year, or data from this year appears in last year. Why?
A: Your log format does not include year information, so Sawmill has to guess the year. Use a different log format if possible (one which includes year information). See the long answer for a way of manually setting the year for blocks

of log data. For full details, see **Years are wrong in the statistics**.

Q: When I run Sawmill as a CGI program under IIS, I get an error message "CGI Timeout: The specified CGI application exceeded the allowed time for processing. The server has deleted the process." What can I do about that?
A: Set the IIS CGI timeout to a high value, like 999999. For full details, see **IIS CGI Timeout**.

Q: I've forgotten the password I chose for Sawmill when I first installed it; how can I reset it?
A: Delete the users.cfg file in the LogAnalysisInfo folder/directory of your installation. For full details, see **Resetting the Administrative Password**.

Q: When I run Sawmill as a CGI, it runs as a special user (nobody, web, apache, etc.). Then when I want to use Sawmill from the command line or in web server mode, the permissions don't allow it. What can I do about this?
A: Loosen the permissions in the Preferences, or run your CGI programs as a different user, or run your command line programs as the CGI user. For full details, see **CGI User Permissions**.

Q: How much memory/disk space/time does Sawmill use?
A: It depends on how much detail you ask for in the database. It uses very little if you use the default detail levels. For full details, see **Resource Usage**.

Q: When I add up the number of visitors on each day of the month, and I compare it to the total visitors for the month, they're not equal. Why not? Also, why doesn't the sum of visitors on subpages/subdirectories add up to the total for the directory, and why doesn't the sum of visitors on subdomains add up to the total for the domain, etc.? Why are there dashes (-) for the visitor totals?
A: Because "visitors" is the number of *unique*visitors, a visitor who visits every day will show up as a single visitor in each day's visitors count, but also as a single visitor for the whole month -- not 30 visitors! Therefore, simple summation of visitor numbers gives meaningless results. For full details, see **Visitor Totals Don't Add Up**.

Q: When I look at my statistics, I see that some days are missing. I know I had traffic on those days. Why aren't they shown?
A: Your ISP may be regularly deleting or rotating your log data. Ask them to leave all your log data, or rotate it over a longer interval. It's also possible that your log data does not contain those days for another reason. For full details, see **Days Are Missing from the Log Data**.

Q: My log data contains referrer information, but I don't see referrer reports, or search engines, or search phrases. Why not?
A: Sawmill includes referrer reports if the *beginning*of the log data includes referrers. If your log data starts without referrers, and adds it later, you won't see referrer reports. Create a new profile from the latest log file (with referrers), and change the log source to include all log data. For full details, see **Referrer Reports Missing**.

Q: When I process log data with Sawmill, it uses most or all of my processor; it says it's using 90%, or even 100% of the CPU. Should it be doing that? Is that a problem?
A: Yes, it should do that, and it's not usually a problem. Any CPU-intensive program will do the same. However, you can lower the CPU usage if you need to with **Maximum CPU usage**. For full details, see **Sawmill Uses Too High a Percentage of CPU**.

Q: Is Sawmill Year 2000 Compatible?
A: Yes. For full details, see **Year 2000 Compatibility**.

Q: How do I build a database from the command line?
A: Run "*executable*-p *profilename*-a bd" from the command line window of your operating system. For full details, see **Building a Database from the Command Line**.

Q: Sawmill does not recognize my Symantec SGS/SEF log data, because it is binary. How can I export this data to a text format so Sawmill can process it?
A: Use flatten8, or remorelog8 For full details, see **Exporting Symantec SGS/SEF data to text format**.

Q: How can I track full URLs, or HTTP domains, or resolved hostnames, when analyzing PIX log data?
A: You can't track full URLs or HTTP domains, because PIX doesn't log them; but you can turn on DNS lookup in the PIX or in Sawmill to report resolved hostnames. For full details, see **Tracking URLs in Cisco PIX log format**.

Miscellaneous

Q: Where did the name "Sawmill" come from?
A: A sawmill is a tool that processes logs, and so is Sawmill. For full details, see **The Name "Sawmill"**.

Q: Why are new versions of Sawmill released so often? Is it buggy? Do I need to download every new version?
A: We ship new versions to provide our customers with the latest minor features and bug fixes quickly. Sawmill is no buggier than any other software, and you don't need to download a new release unless you're having problems with the current one. For full details, see **Frequent New Versions of Sawmill**.

Q: How do I duplicate or clone a profile? I want to create a new profile with all the settings in my existing profile.
A: Copy the profile .cfg file, change the first line to match the filename, and change the label in the file; or use Create Many Profiles. For full details, see **How to Duplicate a Profile**.

User Guide Index

Welcome to the Sawmill User Guide

This guide has been written for Sawmill Users, and not for Sawmill Administrators. If you are using an Administrator level access, then it will not be in this Guide. If you do not find the information you need here, contact your Sawmill Administrator, or Sawmill support.

NOTE: This Guide has been written with examples from web site analysis. If you're analyzing other types of log data, this guide will still be useful, but the report names may not match those shown in your Reports. The basic concepts are the same for all types of log data.

This Guide has the following sections:

Installation

Web Server Mode or CGI Mode?

Installation Modes

Sawmill can run either in web server mode (stand-alone) or in CGI mode. In both modes, you can access Sawmill through a web browser on your desktop.

This section will help you decide whether you want to run Sawmill in "web server mode," or in "CGI mode." In web server mode the installation is easy, Sawmill runs its own web server, and serves statistics using it. CGI mode is better if you want to run Sawmill on a shared web server system, or a system you have only limited access to.

Here are the advantages and disadvantages of web server mode vs. CGI mode:

Advantages of Web Server Mode:

- Sawmill uses its own web server -- there does not need to be an existing web server on the computer where it is running.
- Sawmill is extremely simple to install in web server mode. You just run it, point your browser at it, choose a password, and you're ready to start using it.
- Sawmill Scheduler is available.

Disadvantages of the Web Server Mode:

- Memory is being used, even when Sawmill isnâ€™t being used.
- Server features, like HTTPS authentication, and other features can not be used in this mode.
- May not be possible to use because of firewall limitations and other restrictions.

Advantages of CGI Mode:

- In CGI mode, Sawmill only uses memory and other resources while it's actively in use.
- There is no extra configuration required to start Sawmill at system boot time -- it is always available.
- In CGI mode, Sawmill can use the services of the web server that's running it. This makes it possible to use HTTPS, server authentication, and other powerful server features with Sawmill.
- In some environments, web server mode may not be possible or permissible, due to restrictions of the server, firewall limitations, and other considerations. For instance, if you have only FTP access to your web server, and you want to run Sawmill on the server, you *must* use CGI mode.

Disadvantages of CGI Mode:

- In CGI mode, the installation and startup process is more involved, although fairly easy.
- The Sawmill Scheduler is not easily available, an external scheduler must be used to schedule builds/updates/etc.

The two different installations are described below:

Web Server Mode Installation

Sawmill needs to be installed differently depending on the platform you intend to run it on:

- **Windows**: Sawmill is a standard Windows installer. Just double-click the program to start the installer, and follow the instructions.

  Once Sawmill is installed, it will be running as a Windows service. You can access it at http://127.0.0.1:8987/ with a web browser. Sawmill runs as the SYSTEM user by default, which is the most secure approach, but restricts access to network shares or mapped drives. See **Can't See Network Drives with Sawmill as Service** for instructions for running Sawmill as a different user, to get access to that user's mapped drives and network privileges.

- **MacOS**: Sawmill is a disk image. Mount the image, and drag the Sawmill folder to the Applications folder. Once Sawmill is installed, you can start using it by double-clicking the Sawmill application icon (in the Applications/Sawmill folder). Once it's running, click Use Sawmill to start using it.

- **UNIX**: Sawmill is a gzipped tar archive file. You need to transfer that to the UNIX machine where you'll be running Sawmill, if it's not already there. Then you'll need to open a "shell" prompt using telnet, ssh, or the way you normally get to the UNIX command line. Next, gunzip and untar the file using the following command:

  ```
  gunzip -c (sawmill.tgz) | tar xf -
  ```

  You will need to change (sawmill.tgz) to match the name of the file you downloaded.

  Once the archive is uncompressed and extracted, you can run Sawmill by changing to the installation directory, and typing the name of the executable file from the command line:

  ```
  cd (installation-directory)
  ./sawmill
  ```

  You may need to change the filename to match the actual version you downloaded. Sawmill will start running, and it will start its own web server on the UNIX machine (using port 8987, so it won't conflict with any web server you may already be running there). To start using Sawmill, copy the URL printed by Sawmill in your window, and paste it into the URL field of your web browser, and press return. You should see Sawmill appear in your web browser window.

  Note: You can add a single ampersand (&) to the end of the command line that starts Sawmill, to run Sawmill "in the background," which allows you to close your terminal window without killing Sawmill. On some systems, you may also need to add `nohup` to the beginning of the command line for this to work properly.

If you have any problems installing Sawmill in web server mode, please see **Troubleshooting Web Server Mode**

CGI Mode Installation

To install Sawmill in CGI mode, you will need to extract the Sawmill program, and copy it to your web server's CGI directory.

IMPORTANT: There are different versions of Sawmill for each platform (e.g. Windows, Macintosh, Linux, etc.) and a version designed for one platform will not work on another. In CGI mode, you must install the version of Sawmill that matches your **server's** platform, *not* the version that matches your desktop computer. For instance, if you're running Windows at home, and you install Sawmill on your ISP's web server, and the web server is running Linux, you need to install the Linux version of Sawmill on the web server. If you don't know what platform your web server is running, you can find out by asking your ISP or system administrator, if it's a UNIX system, you can also find out by logging in by telnet and typing "uname -a".

Make sure you understand the previous paragraph! And now download the correct version of Sawmill, the one that matches your ***web server*** platform. Make sure you do the FTP download in BINARY mode, or Sawmill will not work.

You install Sawmill in CGI mode differently depending on the type of your web server. See **Web Server Information** if you need help finding your cgi-bin directory.

- **Windows**. If the web server is IIS, this is a difficult installation due to the security features of IIS, which make it difficult to run a binary CGI program -- consider using web server mode instead. If you need CGI mode, then it *is* possible, but it's not as easy as on other platforms. See **Installing Sawmill as a CGI Program Under IIS** for more information on installing in CGI mode under IIS.

  You need to upload the SawmillCL.exe file and the LogAnalysisInfo folder to your server's cgi-bin directory (Windows may hide the .exe part of the filename, but that *is* its actual full filename). The easiest way to get this file and this folder is to install the Windows version of Sawmill on a local Windows desktop machine, and then look in the Sawmill installation directory (C:\Program Files$PRODUCT_NAME\ by default); the SawmillCL.exe file and LogAnalysisInfo folder will be there. If you don't have access to a Windows machine locally, please contact **support@flowerfire.com** and we will send you this file and folder. Make sure you do the upload in BINARY mode, or Sawmill will not work! Once you've uploaded it to your cgi-bin directory, you can access it using the URL http://(yourserver)/cgi-bin/SawmillCL.exe (replace yourserver with the name of your domain). Sawmill should appear in the web browser window. If it still doesn't work, see **Troubleshooting CGI Mode** below.

- **UNIX or MacOS with SSH access**. If your server is UNIX or similar, and you have SSH access to your server, then download the file to your server and gunzip/untar it according to the instructions above (in the web server mode installation section). Then copy the executable file and LogAnalysisInfo directory from the installation directory to your cgi-bin directory, using this command:

  ```
  cp (installation-directory)/sawmill (cgi-bin)/sawmill.cgi
  cp -r (installation-directory)/LogAnalysisInfo (cgi-bin)
  ```

  You may need to change the name of sawmill to match the version you downloaded, and you will definitely need to change the (cgi-bin) part to match your web server's cgi-bin directory.

  You can access Sawmill now using the URL http://(yourserver)/cgi-bin/sawmill.cgi replacing (yourserver) with the

actual name of your server. Sawmill should appear in the web browser window.

Make the Sawmill executable file and LogAnalysisInfo directory accessible by the CGI user. The CGI user depends on the configuration of the web server, but it is often a user like "web" or "apache" or "www". If you have root access you can use this command, after cd'ing to the cgi-bin directory, to change ownership of the files:

```
chown -R apache (PRODUCT_EXECUTABLE_DOCS).cgi LogAnalysisInfo
```

If you do not have root access, you may need to open up permissions completely to allow the root user to access this:

```
chmod -R 777 (PRODUCT_EXECUTABLE_DOCS).cgi LogAnalysisInfo
```

However, please note that using chmod 777 is *much* less secure than using chown--anyone logged on to the server will be able to see or edit your Sawmill installation, so in a shared server environment, this is generally not safe. If possible, use chown as root instead.

For more information, see **Troubleshooting CGI Mode** below.

- **UNIX with only FTP access**. If you have a UNIX server, and you have only FTP access to your server (you cannot log in and run commands by ssh, or in some other way), then you need to do things a bit differently. Here's how to do it:

    1. Download the file to your desktop system, remember you need the version that matches your ***server***, not the version that matches your desktop system. Download in BINARY mode.

    2. Use a gunzipping and untarring utility on your desktop system to decompress the file, **WinZip** on Windows, **StuffIt Expander** on Mac, or gunzip/tar if your desktop is also UNIX.

    3. Rename the sawmill file to sawmill.cgi.

    4. Upload sawmill.cgi to your server's cgi-bin directory. Make sure you use BINARY mode to do the transfer, otherwise it won't work.

    5. Upload the entire LogAnalysisInfo directory including all the files and directories in it, to your server's cgi-bin directory. In order to do this conveniently, you will need to use an FTP client which supports recursive uploads of directories, including all subdirectories and files. Make sure you use BINARY mode to do the transfer, or it won't work.

    6. Make sawmill.cgi executable in your web server using "chmod 555 sawmill.cgi" if you're using a command-line FTP program, or using your FTP program's permission-setting feature otherwise.

You can now access Sawmill using the URL http://(yourserver)/cgi-bin/sawmill.cgi replacing "yourserver" with the actual name of your server. Sawmill should appear in the web browser window. For more information, see the section on **Troubleshooting CGI Mode.**

Troubleshooting Web Server Mode

If Sawmill is not working in web server mode (if you're not getting a page back when you enter the URL, or if you're getting an error page), try these suggestions:

1. Make sure you installed the version of Sawmill that matches the computer you're running Sawmill on (for instance, you can't run the Solaris version of Sawmill on Windows). If there were several choices available for your platform (e.g. Old and New, static and dynamic), try all of them.

2. Make sure you downloaded Sawmill in BINARY mode.

3. In UNIX, make sure the Sawmill program is executable.

4. Contact **support@flowerfire.com**.

There are more troubleshooting suggestions in **Troubleshooting**.

Troubleshooting CGI Mode

If Sawmill is not working in CGI mode (if you're not getting a page back when you enter the URL), try these suggestions:

1. Make sure you installed the version of Sawmill that matches your server, not your desktop system. If there were

several choices available for your platform (e.g. Old and New, static and dynamic), try all of them.

2. Make sure you uploaded the Sawmill program in BINARY mode.

3. On UNIX, make your cgi-bin directory writable, using "chmod a+w (cgi-bin)" if you have telnet access, or using your FTP client's "change permissions" feature if you have only FTP access.

4. Create a directory called LogAnalysisInfo in your cgi-bin directory, using "mkdir (cgi-bin)/LogAnalysisInfo" on UNIX with telnet access, or by using your FTP client's "make directory" feature if you don't have telnet access. If your server is UNIX, also make it writable using "chmod -R a+rwx (cgi-bin)/LogAnalysisInfo" if you have telnet access, or by using your FTP client's "change permissions" feature if you have only FTP access.

5. Contact **support@flowerfire.com**. If you're having trouble, we will install Sawmill for you at no cost, even if it's just the trial version.

There are more troubleshooting suggestions in **Troubleshooting**.

Troubleshooting

You should consult this page when something goes wrong with the installation or the use of Sawmill.

For additional installation and related troubleshooting tips, see the end of the **Installation** chapter, and the Troubleshooting section of the **FAQ**.

Sawmill doesn't run properly when I try to run or launch it

On Windows and MacOS, this should never happen; double-clicking Sawmill or running it from the Start menu, should bring up a window with the Sawmill logo and some text. If it doesn't, please contact us at **support@flowerfire.com**, so we can investigate.

On UNIX, this can be the major initial obstacle to using Sawmill. There are many types of UNIX out there, and often several variants of each type. Since Sawmill is a binary compiled program, it will only run on a system similar to the one it was built on. We have attempted to build Sawmill on as many different systems as possible, to ensure that it runs on the vast majority of customers' machines, but it is a difficult task due to the great variety available.

To start, you need to find a version of Sawmill with a hardware architecture and operating system which matches yours. For instance, if you're running Linux on an x86/Pentium system, you need to download the version of Sawmill for x86/Linux. The x86/FreeBSD version won't work, and neither will the Alpha/Linux version. If there are several versions of Sawmill which seem to approximately match your setup, try them *all* -- keep running them until one of them works. Remember that you have to gunzip and untar them before you can use them -- see **Installation**. If possible, run them from the command line; e.g., using ssh or telnet first, even if you intend to run them as CGI -- there are a lot fewer things that can go wrong with a command-line run, so you'll know right away if it doesn't work. If none of them work when run from the command line (via ssh or telnet), please contact us at **support@flowerfire.com**, so we can help you get Sawmill running.

If you can't run Sawmill from the command line; i.e., if you have only FTP access to your server), you can still use it. You'll need to gunzip it and untar it somewhere else (gunzip and untar programs are available for all platforms), and then upload the `sawmill` binary program (the large file in the resulting folder) and the LogAnalysisInfo directory in *binary* mode to your cgi-bin folder. Change it to readable and executable by "all" or "world" using your ftp client (using chmod or whatever command your ftp client uses to change permissions). If possible, change your cgi-bin folder to world writable temporarily during the Sawmill install; it makes the install simpler. Some servers won't allow that, so if you have problems, change it back to world read-only and try again. In any event, you should change it back to read-only after installation. Once the binary is there and ready to run, try running it from a web browser, using the appropriate URL for your server (it's often something like http://www.myserver.com/cgi-bin/sawmill). If you get an error, try looking in your web server's error log, if one is available, to see what went wrong. If it simply didn't run, or crashed, and if there's another version available for your platform, try that version instead. If none of the versions work, please contact us at **support@flowerfire.com**.

I can't "zoom in" past a certain level

If you're finding that beyond a certain depth, Sawmill won't let you zoom in any further, the items in the tables are no longer clickable, you probably have a limit on the depth of that field in the Database Structure Options. Edit your profile, go to the Database Structure Options, and increase the Suppress value for the field you want more detail on. Then rebuild your database and the items should become clickable.

The web browser crashes

It is important to distinguish this problem from Sawmill crashes, listed below. If you are using Sawmill through a web browser, and the web browser window, the one displaying the Sawmill menu, profile information or statistics disappears, then it is your web browser that has crashed, not Sawmill. This is usually caused by a bug in your web browser -- make sure you are running the latest version available of your browser.

Sawmill crashes

See the section above, **The web browser crashes**, to verify that it was really Sawmill, and not your web browser, which crashed. The most common cause of crashes is when Sawmill runs out of memory while building a database. Try watching Sawmill's memory usage (using `top` on UNIX, or the Process Manager on Windows, or the About This Computer window on MacOS) while it is processing, to see if it is consuming all available memory. Sawmill will often generate an error when it runs out of memory, but due to technical reasons, this is not always possible, and sometimes running out of memory can cause Sawmill to crash. See **Sawmill runs out of memory**, below, for suggestions on limiting Sawmill's memory usage.

With the exception of out-of-memory problems, Sawmill should never crash; if it does, it is probably a significant bug in

Sawmill. We do our best to ensure that Sawmill is bug-free, but all software has bugs, including Sawmill. If Sawmill is crashing on your computer, we would like to hear about it -- please send email to **support@flowerfire.com**, describing the type of computer you are using and the circumstances surrounding the crash. We will track down the cause of the crash, fix it, and send you a fixed version of Sawmill.

Sawmill runs out of memory

Sawmill can use a lot of memory when it processes large amounts of data. If there is not enough memory available for Sawmill to perform the task you have requested, it will generate an error message, reporting that it could not allocate as much memory as it needed, and it will stop doing whatever it was doing.

One way to fix this problem is to increase the amount of memory available to Sawmill. On any platform, you can add more physical memory, or increase the amount of virtual memory, to let Sawmill have more memory to work with.

On Windows and MacOS, there is usually no operating-system memory restriction on Sawmill, but if you're using a shared server, there may be one -- if you find Sawmill is running out of memory or crashing immediately when used, check with your server administrator to see if there are any restrictions on memory usage. On UNIX, you can increase the amount of memory available to individual processes; how this is done varies, but try the `limit` command. See **Memory, Disk, and Time Usage** for a discussion of memory usage, disk usage, processing time, and how they relate to each other.

The Temporary Folder

The most difficult part of the CGI mode Sawmill installation is choosing the correct Temporary folder and Temporary folder URL. When you run Sawmill in web server mode, with Sawmill running its own internal web server, and serving its pages via HTTP, this step is not necessary, which is one of the advantages of web server mode. However, if you prefer to run Sawmill in CGI mode, you will have to choose the Temporary folder and URL.

Sawmill includes images as part of its output, including pie charts, line graphs, bar charts, and icons. To display these images, it first creates GIF image files in the Temporary folder, and then embeds the URLs of those images in the HTML of the pages it generates, using the Temporary folder URL as the basis for choosing the correct URL.

The Temporary folder and the Temporary folder URL are two different ways of describing the *same* folder. They must point to the same folder for Sawmill's output to look correct. The temporary folder:

- must be within the HTML pages folder of the web server which is serving Sawmill. Web servers serve all their pages from a particular folder on the hard drive; the Temporary folder must be somewhere within this folder. This folder is called different things by different web servers, but some common names are "htdocs", "html", "data", "wwwroot", and "Web Pages." If you do not know where this folder is for your web server, you can find out in your web server's documentation. See **Web Server Information** for a list of web servers, and where the HTML pages folder is for them.

- must either exist and be writable by Sawmill (running as a CGI program), or it must be possible for Sawmill to create it. UNIX users may need to use the `chmod` command to set permissions correctly.

The Temporary folder should be described using your platform's standard pathname format (see **Pathnames**). The Temporary folder URL should describe the same folder as the Temporary folder, but as a URL, as it might be accessed by using a web browser, and by browsing the server that Sawmill is running under. The following examples illustrate how it might be set for various platforms; see **Web Server Information** for more specific information.

As the hostname part of the URL, you will need to specify the machine Sawmill is running on. In the examples below, this is chosen to be www.mysys.com; you will need to replace this part of the URL with your machine's actual hostname.

MacOS: If the root of your web server's HTML pages is at /Library/WebServer/Documents (i.e. the Documents folder, which is in the WebServer folder, which is in the Library folder), and this is accessible from your web browser as http://www.mysys.com/, then you could enter /Library/WebServer/Documents/sawmill/ as the Temporary folder and http://www.mydomain.com/sawmill/ as the Temporary folder URL.

Windows: If the root of your web server is at C:\inetpub\wwwroot\ (i.e. the wwwroot folder, which is in the inetpub folder, which is on the C drive), and this is accessible from your web browser as http://www.mysys.com/, then you could enter C:\\inetpub\\wwwroot\\ sawmill\\ as the Temporary folder and http://www.mydomain.com/sawmill/ as the Temporary folder URL.

UNIX: If the root of your web server is at /home/httpd/html/, and this is accessible from your web browser as http://www.mydomain.com/, then you could enter /home/httpd/html/sawmill/ as the Temporary folder and http://www.mysys.com/sawmill/ as the Temporary folder URL.

It is also possible to use relative pathnames, which sometimes makes it easier; e.g. on UNIX if both the cgi-bin directory and the html directory are in the same directory, you can use ../html/sawmill as the temporary directory, without having to specify the full pathname.

See **Web Server Information** for more information.

Web Server Information

Sawmill can run as its own web server, serving its HTML pages to all of the most common web browsers via HTTP. This is often the best way to run Sawmill, but if it's not possible or desirable to run it that way, you can also run it as a CGI program under another web server.

Sawmill is designed to work with any web server which supports the very common CGI standard. For ease of installation and profile, however, the following list includes a number of known web servers which Sawmill has been fully tested with, and information about how Sawmill should be configured when using it with them.

Each entry assumes that the default installation options were used, and that the full hostname of the computer running the web server is www.myhost.com; you will need to replace "www.myhost.com" by the actual hostname of your server. The CGI folder is the folder where the Sawmill executable program (application) should be placed to begin using it. The Web pages root is the root of all web pages for the server, and is used to compute the Temporary folder. The Temporary folder and Temporary folder URL are *suggested* values for the Temporary folder and the Temporary folder URL (see **The Temporary Folder**).

Web Server	Information
Xitami (Windows)	CGI folder: C:\Program Files\Xitami\cgi-bin\ Web pages root: C:\Program Files\Xitami\webpages\ Temporary folder: C:\Program Files\Xitami\webpages\{=PRODUCT_EXECUTABLE_DOCS=}\ Temporary folder URL: http://www.myhost.com/sawmill/
OmniHTTP (Windows)	CGI folder: C:\httpd\cgi-bin\ Web pages root: C:\httpd\htdocs\ Temporary folder: C:\httpd\htdocs\{=PRODUCT_EXECUTABLE_DOCS=}\ Temporary folder URL: http://www.myhost.com/sawmill/
Microsoft Peer Web Services (Windows)	CGI folder: C:\inetpub\cgi-bin\ Web pages root: C:\inetpub\wwwroot\ Temporary folder: C:\inetpub\wwwroot\{=PRODUCT_EXECUTABLE_DOCS=}\ Temporary folder URL: http://www.myhost.com/sawmill/
Microsoft IIS 3 (Windows)	Sawmill and IIS version 3 do not get along! Please upgrade to IIS 4, which works great.
Microsoft IIS 4/5 (Windows)	CGI folder: C:\inetpub\cgi-bin\ Web pages root: C:\inetpub\wwwroot\ Temporary folder: C:\inetpub\wwwrootsawmill\ Temporary folder URL: http://www.myhost.com/sawmill/ Log File Location: c:\Windows\System32\LogFiles\W3svc1
MacOS X Web Server (Apache-based MacOS X Standard Server)	CGI folder: /Library/WebServer/CGI-Executables Web pages root: /Library/Webserver/Documents Temporary folder: /Library/Webserver/Documents/sawmill Temporary folder URL: http://www.myhost.com/sawmill/ Log File Location: /private/var/log/httpd/access_log Notes:
apache (UNIX)	CGI folder: /home/httpd/cgi-bin/ Web pages root: /home/httpd/html/ Temporary folder: /home/httpd/html/sawmill/ Temporary folder URL: http://www.myhost.com/sawmill/ Log File Location: varies; try /var/log/httpd/access_log Notes: The Web pages root varies from installation to installation; it may be somewhere else. Consult your apache profile files (often in /etc/httpd/conf/, but this varies too) for the exact location (search for DocumentRoot).

If you're running Sawmill on a server not listed above and need help configuring it, please contact us at **support@flowerfire.com**. If you're using a different server, and have it working, we'd really appreciate it if you could forward to us the above information for your server, so that we can add your server to the list, and save time for future users.

DOCUMENTATION

Using Sawmill with WebSTAR V on MacOS X

This chapter was contributed by David Wrubel.

If you choose to run Sawmill in web server mode, it will run as a separate webserver application with its own default port number 8987 on the same Mac as WebStar V.

Installation

Because of the way WebStar V is installed on MacOS X, the installation process of Sawmill is a little tricky. Part of the Sawmill (for WebStar V) installation process involves command line work in the OS X UNIX "Terminal". This work is done, using the root user privileges and should be done with extreme caution since the root user has write access to all files, including system files.

Because of security reasons, 4D instructs WebStar users only to install the WebStar application when logged in MacOS X as the admin user. 4D advises WebStar V users to create a MacOS X user that does NOT have admin privileges. The user is normally called "webstar" and is used when WebStar V is running. This setup protects system files and the WebStar V application against some types of unauthorized intrusions. At the same time there is full access to all website files that users upload and download.

The above imposes a problem for Sawmill, since Sawmill must be installed in MacOS X when the user is logged in as the admin user. When Sawmill is installed and you log out and then log in as the "WebStar" user, to run WebStar V, you will not have access to run Sawmill, since Sawmill will only run when logged in as the MacOS X admin user.

You have to change the ownership ID (or valid user name) of all Sawmill files from the admin user (typically the user that installed Mac OS X) to the "webstar" user, to be sure that Sawmill will run when logged in as the webstar user.

To do this follow these steps, remember to be careful, as you will be logged in as the root user as long as you are working in the Terminal:

1. Webstar V and the "webstar" user must already be installed and configured.
2. If you haven't done it already, login as the admin MacOS X user and install Sawmill. The default installation process will create a "Sawmill" folder in the MacOS X "Applications" folder. All Sawmill files and folders will be installed in the Sawmill folder. Do not start Sawmill after the installation.
3. In the /Applications/Utilities folder, start the "Terminal" application.
   At the command prompt use the following commands and use your admin user password when asked, in this example the admin user is "davidw". Also be sure that the Terminal gives you the same returns. If you experience any differences, you can always type "Exit" to log out from the root user and thereby (hopefully) protect against damage:

Commandline (bold type = your input)	Comments
Welcome to Darwin! [localhost:~] davidw%	When you start the Terminal and choose "New" in the "Shell" menu, you will see this in a new terminal window.
[localhost:~] davidw% **sudo su** Password:	SUDO SU gives you root access. Type your admin password and hit the return key.
[localhost:/Users/davidw] root# **cd /applications/sawmill/**	Navigates you to the Sawmill folder. Note that the commandline says you are now "root#".
[localhost:/applications/sawmill] **root# ls -ls** total 9104 0 drwxrwxr-x 3 davidw admin 58 Apr 21 09:43 Extras 24 -rwxr-xr-x 1 davidw admin 9348 Apr 21 09:43 License.txt 8 -rwxrwxr-x 1 davidw admin 3113 Apr 21 09:43 ReadMe.txt 0 drwxrwxr-x 3 davidw admin 58 Apr 21 09:43 Sawmill.app 0 drwxrwxr-x 6 davidw admin 160 Apr 21 09:43 Startup 9072 -rwxr-xr-x 1 davidw admin 4641768 Apr 21 09:43 sawmill	
[localhost:/applications/sawmill] root# **cd..**	Navigate back to the Applications folder.
[localhost:/applications] root# **chown -R webstar sawmill**	Now be careful: key the command exactly as shown. This changes the ownership of all files in the Sawmill folder including the Sawmill folder itself.
[localhost:/applications] root# **cd sawmill**	Navigate into the Sawmill folder
[localhost:/applications/sawmill] root# **ls -ls** total 9104 0 drwxrwxr-x 3 webstar admin 58 Apr 21 09:43 Extras 24 -rwxr-xr-x 1 webstar admin 9348 Apr 21 09:43 License.txt 8 -rwxrwxr-x 1 webstar admin 3113 Apr 21 09:43 ReadMe.txt 0 drwxrwxr-x 3 webstar admin 58 Apr 21 09:43 Sawmill.app 0 drwxrwxr-x 6 webstar admin 160 Apr 21 09:43 Startup 9072 -rwxr-xr-x 1 webstar admin 4641768 Apr 21 09:43 sawmill	This lists the files and folders in the Sawmill folder to check that the ownerships have really been changed. Note that the owner is now "webstar" instead of "davidw".
[localhost:/applications/sawmill] root# **exit** exit [localhost:~] davidw%	Log out the root user by typing "exit" and hit the return key. Note that the admin user is returned.

4. Be sure that you have typed "exit" and you are now the admin user. Close Terminal.
5. Log out of Mac OS X (not shut down, only log out).
6. Log in as the "webstar" user.

You can now use Sawmill at the same time as running Webstar V.

Log formats

When choosing log formats in WebStar V's "Admin Client" for a specific website, use "WebSTAR Log Format". Select all tokens and schedule WebStar V to archive log files once every day.

In Sawmill's "Quick Start" you will be asked where the log files are located. WebStar V always creates log files for each defined website in the "logs" folder that is in the "WS_Admin" folder for each website.

Sawmill will automatically choose "WebSTAR Log format" and ONLY that format, if, and only if, the log is written correctly. If Sawmill gives you another log file format to choose from (including the correct format), then there is an error. Sawmill will only choose the correct log format when the first line in the log file, in the first log file that you want to be processed by Sawmill, has the correct log file header.

If you experience this error do the following:
1. Open the first log file that is supposed to be processed by Sawmill, in your favorite text editor on a MacOS X machine (this is important, since WebStar V log files now have file names that can be longer than 32 characters).

2. Find the first instance of the following text and move your cursor just before the text (with your date and time):
!!WebSTARSTARTUP 25/May/02 01:19:40
!!LOG_FORMAT BYTES_RECEIVED BYTES_SENT C-DNS.....

3. Mark all text from before the text and to the top of the file and delete the marked text. The above text should now be the very first text in the log. Save the file. Run "Quick Start" again.

Installing Sawmill as a CGI Program Under IIS

This chapter was contributed by a customer.

This describes how to install Sawmill in CGI mode under IIS. It was done on IIS 4 (NT 4, SP6a) in this example, but the basic method should work for any IIS installation.

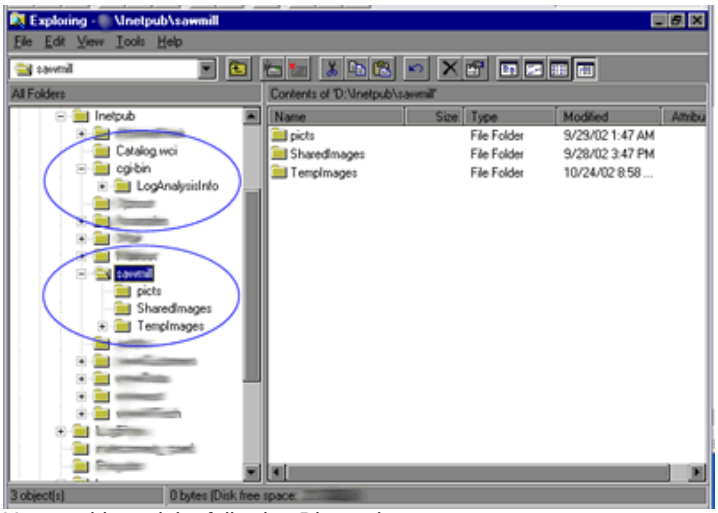
CGI based or Sawmill Web Server?
There are two ways to install Sawmill -- as a CGI program or as a web server. this document deals only with CGI mode installation. See [docs-installation] for information about web server mode, and the advantages and disadvantages of each way. Because of the security issues involved with installing Sawmill under IIS, web server mode may be a better solution for Windows installation -- in web server mode none of the following configuration steps are necessary.

Initial Installation
Start following the CGI installation instructions ([docs-installation]). On some installations of IIS, that's all that is needed. If you're able to see the Administrative Menu with images in your CGI installation, then you're done. If you have problems, continue on:

URLScan
One common source of problems is URLScan. Microsoft has a tool called "IIS Lockdown" to help block all those nasty worms. By default, URLScan blocks are attempts to run EXE (among many other things) files on the system. Unfortunately, this also blocks all accesses to Sawmill, which is an EXE file. By editing the EXE blocking, and giving all the Sawmill directories FULL permissions, you can get it to work. This is a simple approach, but not very secure. A more secure approach is described below.

Installation Procedures:

Sawmill Directories:

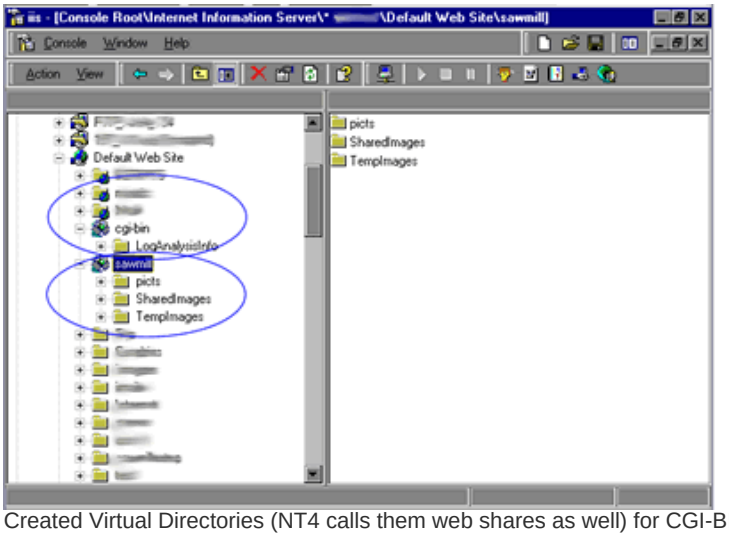


You would need the following Directories:

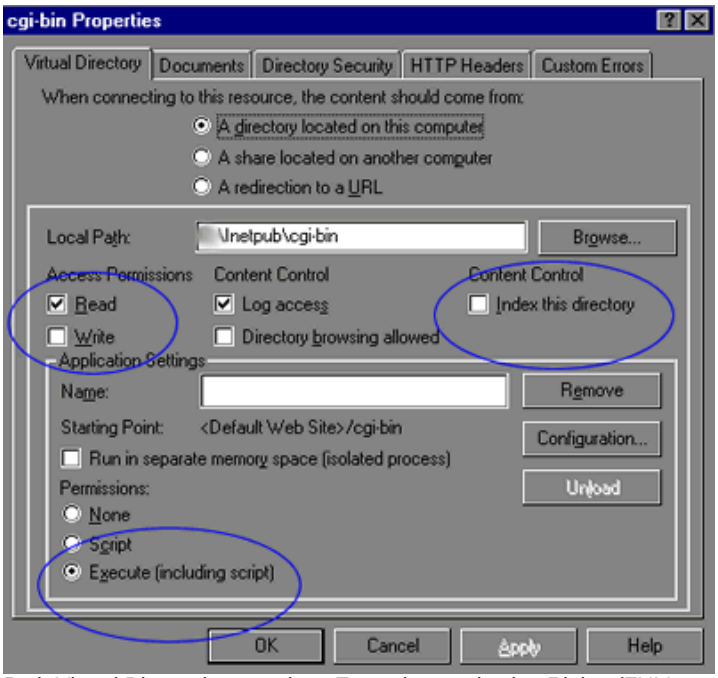
{BASE}\inetpub\wwwroot\ ==> **your website**
{BASE}\inetpub\ sawmill
{BASE}\inetpub\cgi-bin\
{BASE}\inetpub\cgi-bin\LogAnalysisInfo (sawmill creates this automatically)

Initially give both cgi-bin and sawmill FULL permissions.

IIS Console setup:



Created Virtual Directories (NT4 calls them web shares as well) for CGI-BIN & Sawmill in IIS Management Console.



Both Virtual Directories are given Execution, and write, Rights (FULL permission)
Make sure the "Index this directory" is checked **OFF**
After the installation is completed we will come back and change this to a more secure setting.
Keep in mind the **\cgi-bin** and and **sawmill** directories, are in fact virtual directories under your website, and are not physically under your "website" directory.

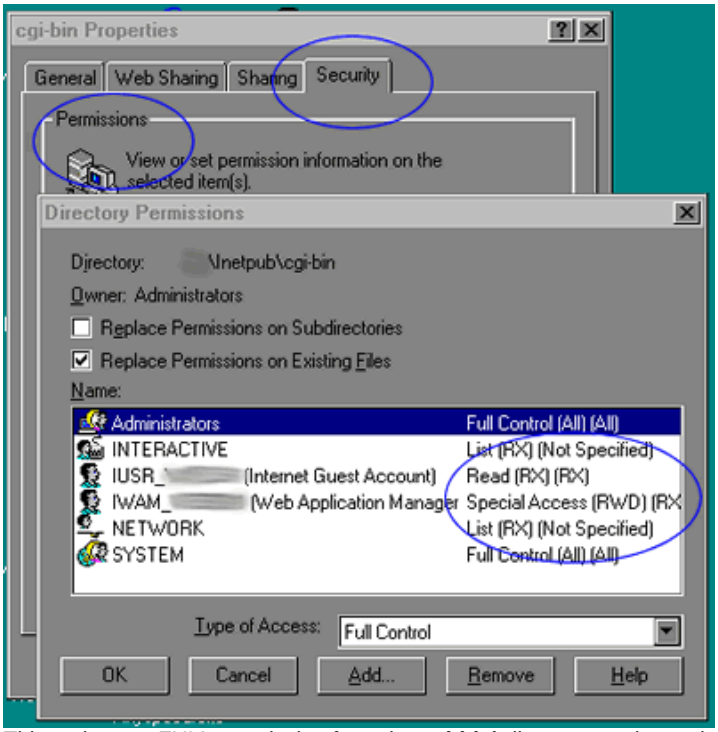
Execution and Sawmill Installation:
Once we have all the Subdirectories and Virtual directories in place then:

- Copy "SawmillCL.exe" to **{BASE}\inetpub\cgi-bin** directory.
- Execute (run) "SawmillCL.exe".
- Following the Sawmill Installation procedures from the manual. <link to the Sawmill documentation>
- Establish passwords and Temp Directory. **{BASE}\inetpub\{=PRODUCT_EXECUTABLE_DOCS=}**
- Create your first "configuration", and add the Log Files, and look at the "Statistics".
- **Enjoy your hard work ;-)**

Securing the Installation:

Sawmill needs access to many different subdirectories, which it automatically creates when you execute the program, or try to view the statistics. Therefore, it needs permissions to Write, Read, Create Subdirectory, and Execute!!
The reason we gave **FULL permission** rights to all the subdirectories was the fact that Sawmill creates many additional subdirectories during it's installation routines. Therefore we need to give it the ability to create these subdirectories. However, after the initial installation, we can take away permissions from **{BASE}\inetpub\cgi-bin** and **{BASE}\inetpub\{=PRODUCT_EXECUTABLE_DOCS=}**, to run a more secure server.

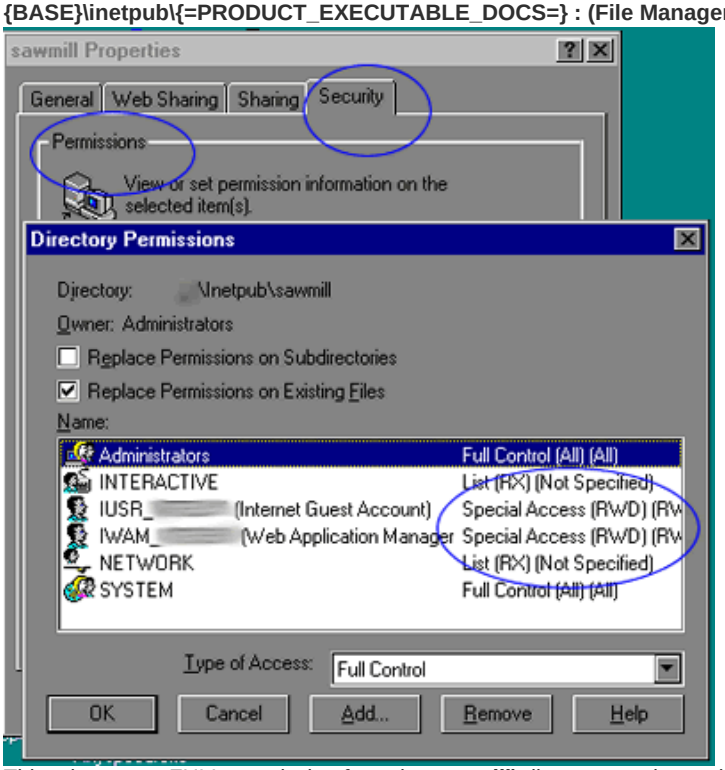
{BASE}\inetpub\cgi-bin\ : (File Manager)



This took away FULL permission from the **cgi-bin** directory, and gave it **Read/Execute ONLY.**
Note: When you make the change here, make sure the "Replace Permission on Subdirectories" is checked **OFF.**

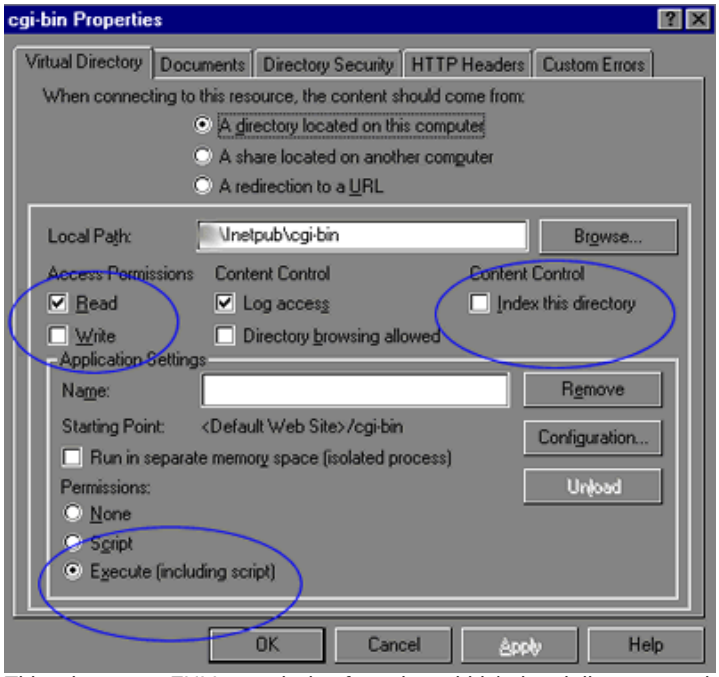
{BASE}\inetpub\cgi-bin\LogAnalysisInfo : (File Manager)
Here, Sawmill still needs to create directories for all additional websites, or if there are any changes to the "configuration".
However, there is no need to Execute any scripts here. So give **Read/Write/Delete** Permission.
Note: When you make the change here, make sure the "Replace Permission on Subdirectories" is checked **ON.**

{BASE}\inetpub\{=PRODUCT_EXECUTABLE_DOCS=} : (File Manager)



This takes away FULL permission from the **sawmill** directory, and gave it **Read/Write/Delete** permission, (no Execution)
Note: When you make the change here, make sure the "Replace Permission on Subdirectories" is checked **ON.**

\cgi-bin : (IIS Console)



This takes away FULL permission from the cgi-bin\ virtual directory, and gave it **Read/Execute** permission.
Note: Make sure the "Index this directory" is checked **OFF.**

\{=PRODUCT_EXECUTABLE_DOCS=} : (IIS Console)
"/picts/docs/english/docs_sawmill_VD.gif"}
This takes away FULL permission from the sawmill\ virtual directory, and gave it **Read/Write** permission. (No Execution)
Note: Make sure the "Index this directory" is checked **OFF.**

NOW, your Sawmill Installation on NT should be complete !

Tips:

1. cgi-bin Directory:
By default there is a cgi-bin directory under the "default web site" of the IIS. You could use this Virtual Directory under any web site. However, if you try to delete the directory and create another one (IIS keeps the "cgi-bin" in it's metafile !), it will show up as cgi-bin2 !!
In order to fully delete the old cgi-bin, you will need to use **Microsoft Meta Editor 2.2** (mtaedt22.exe). PLEASE, do this with great care, you COULD kill the whole IIS here!!!

2. Sawmill Instances:
During the installation/debugging, reduce to only one instance of Sawmill .

3. DNS Lookup:
I've also disabled "DNS Lookup", in my Sawmill Configurations.

4. Open Files / IIS Index Server:
In IIS, all Virtual Directories are indexed by default. However, I think there may be a conflict between Sawmill, and the Index Server, due to Open Files. My updates routinely crashed, and sometimes they even crashed the Server !
After I disabled the Indexing on both sawmill\ and cgi-bin\ directories, I've have very few issues with open files and Sawmill.

5. NT Server Scheduler:
By default NT 4 does not have a Scheduler (Windows 2000 server does). We used the Schedule Wizard by **www.authord.com** for periodical scheduling of events. You could even get a version which runs as a service on NT.

6. URLScan.ini:
I have blocked access to the following in the urlscan.ini file:
cmd.exe
root.exe
shell.exe
shtml.exe
.bat
.cmd
.com
.dll

.pl
.cgi
.ini
.dat
.log

Reference: **Microsoft URLScan Security Tool**
Reference: **IISFAQ - URLScan Setup**

The Administrative Menu

The Administrative Menu appears at the left of the main administrative screen. It provides basic administrative functions.

Profiles

Clicking this menu item will show the list of profiles. In this list, you can create new profiles, delete existing ones, edit profile configuration information ("Show Config"), or view reports for a profile ("Show Reporter").

Licensing

Clicking this link will show the licensing page. In this page, you can add and remove licensing, or change licensing from Professional to Enterprise (if you're using a Trial license).

Users

Clicking this link will show the User Editor. In this page, you can add and remove users, and change the options for each user; e.g. you can specify which users have administrative access, and which profiles they are permitted to view.

Scheduler

Clicking this link will show the Scheduler (see **Using the Sawmill Scheduler**). You can create, delete, and edit scheduled tasks in this section. For instance, you can create a task to update all your databases every night, or to send a report of the previous month by email on the 1st of each month.

Preferences

Clicking this link will show the Preferences editor. This lets you change global preferences, including server IP and port, language, charset, and more.

Using Sawmill

This page provides some instructions and guidelines for using Sawmill. It describes and discusses the most common uses with examples.

Creating a profile

When you create a profile Sawmill will ask you for the log source. The log data can come from a file, a collection of files, or from the output of an arbitrary command (this last option is available only on UNIX and Windows). The files can be local, or Sawmill can download them from an FTP site. To use an FTP site, you can use a standard FTP URL to specify the location of the files; click Show Examples on the "Where is the log data" page to see examples of FTP URLs. They can also come from a single file downloaded by HTTP. Sawmill supports and automatically detects a wide range of log formats.

Once you have told Sawmill where your log data is, Sawmill will create a profile for you using reasonable values for all the options. You can then click "Show Reports" to see the reports -- Sawmill will read and process the log data to build its database, and then will display the reports. You could also click "Show Config" to customize any of the profile settings. Sawmill's options can be confusing at first because there are so many of them, but you don't need to change *any* of them if you don't want to. A good way to start is to leave them at their default settings at first, and just look at the reports. Once you're familiar with your statistics using the default settings, you can go back and start changing them if necessary. Options you may eventually decide to change are the Log Filters, which let you include or exclude log data from your database (see **Using Log Filters**) and the Reports, which lets you customize existing reports and create new ones. For performance tuning, you may also want to edit the cross-references or the database fields (see **Cross-Referencing and Simultaneous Filters**).

You can use the Log Filter Options to specify a complex set of filters to control exactly which log entries you are interested in, and which you are not. Some common filters, for instance, those that categorize image files as hits for web server logs are pre-defined, but you may want to enhance the pre-defined filter set with your own filters, or you may want to remove pre-defined filters. See **Using Log Filters** for more information and examples.

Viewing Reports

You can view reports at any time by clicking Show Reports next to the name of a profile in the administrative profile list. You can also switch to the reports when you're editing the profile options by clicking the Reports link in the upper left. For information on using and navigating the statistics pages, see **Reports**.

Building and Updating Databases from the Command Line

You can build and update databases from the graphical interface, but in some cases you may prefer to use the command line to build or update databases. The command line is useful, for instance, if you want to automatically and regularly update the database with the latest log entries (this can also be done from the scheduler; see **Using the Sawmill Scheduler**). For instance, it is possible to set up a cron job on a UNIX system to automatically update the database every day with the previous day's log. The command line would look something like this:

```
sawmill -p profilename -a ud
```

This command line updates the database for the profile `profilename`. See **Power User Techniques** for more examples of command line usage.

For More Information

The Create Profile process creates a profile with default settings, which is generally a good starting point. After that, it's up to you to fine-tune it if you want. There is a wealth of options in **The Config Page**, letting you control everything from the speed of log file processing to the color of graphs bars. To get the most out of Sawmill, you should spend some time looking over these options, or browsing the **All Options** page.

Reports

Reports, created with Sawmill, present log file statistics in an attractive and easily navigable format.

The Reports Header

The header of the Reports page is a bar containing the following:

- **Logo**: The logo for Sawmill
- **Profile Name**: The name of the active profile, the profile whose reports are being displayed
- **Admin**: A link to the administrative interface, the profiles list, and other administrative functions
- **Logout**: A link to log out of Sawmill
- **Help**: A link which opens a new window containing the Sawmill documentation

The Reports Tool Bar

Below the header is a bar which contains the following:

- **Reports**: A link to the Reports, which should be selected if you are current in the Reports. Click this link to get to the Reports from another page.
- **Config**: A link to the Config page, where you can perform profile actions (like rebuilding the database) and change profile options.
- **Calendar**: Click this to open the Calendar window, where you can select a single day, month, or year to use as the date/time filter. When you have selected an item in the Calendar, all reports will show only information from that time period, until the date/time filter is removed (by clicking "Show All" in the Calendar).
- **Date Range**: Click this to open the Date Range window, where you can select a range of days to use as the date/time filter. When you have selected a range in the Date Range, all reports will show only information from that time period, until the date/time filter is removed (by clicking "Show All" in the Calendar).
- **Filter**: Click this to open the Global Filter window, where you can create filters for any fields, in any combination. Filters created here dynamically affect all reports; once you have set a Global Filter, all reports will show only information for that section of the data. Global Filters remain in effect until you remove them in the Global Filter window.

The Reports Menu

At the left of the window is the Reports Menu, which lets you select the report to view. Clicking a category will expand or collapse that category; clicking a report name will change the report display to show that one. Clicking a report name will remove any Zoom filters, but will not remove Global Filters or Date/Time Filters.

The Report

The main portion of the window (lower right) is occupied by the report itself. This is a view of the data selected by the filters (global filters, date/time filters, and zoom filters). This provides one breakdown of the data specified by the filters -- you can select another report in the Reports Menu to break down the same data in a different way.

There are several parts of the report:

The Report Bar

At the top of the report is a bar containing the report label and the current global and date/time filters, if any.

The Zoom Display

The Zoom Display shows what you're zoomed in on, if you're using Zoom filters. For instance, if you've clicked on item X in a table for field F, you will see "Zoomed in on F: > X". You can apply a zoom filter by clicking on a linked table item. Zoom filters disappear when you click a new report name in the Reports Menu at the left.

The Zoom To Menu

The Zoom To Menu shows the name of the report that will be displayed when you zoom. For instance, if you select R from the

menu, and then click an item X in a table for field F, it will zoom you in on X, and simultaneously switch to report R. This can be useful if you want to break down each item in a table by some other report, for instance to see what traffic looked like by hours of day, for a particular day; choose Hour Of Day from the menu in the Days report, and then click on a day to zoom in on that day and see the hours of the day for that day.

If you are not zoomed on anything, there will be no immediate effect when you select a new item in the Zoom To Menu -- it will just change the menu selection. But when you click, the selection in the Zoom To Menu will be used to determine which report to display.

The Report Graph

For some reports, there will be a graph above the table. The existence of this graph, its size, type; e.g., pie chart or bar or line, and other characteristics, varies from report to report and can be edited in the profile config. The graph displays the same information as the table below it.

The Report Table

The Report Table contains the main information of the report. It displays one row per database field item, with the aggregated numerical values (e.g. sum of hits) in columns next to it. It may also include columns showing bar graph representations of the numbers, and/or percentages. The Table Options link above and to the right of the table can be used to change which columns are visible, the sort order, and other aspects of the report. The Number Of Rows menu above and to the right of the table can be used to change the number of rows that are displayed in the table. The sort order can also be changed by clicking a column name; click once to sort by that column, or again to sort in reverse. You can zoom in on a particular item by clicking it -- that will switch you to the report specified in the Zoom menu, and will set a Zoom filter for that item.

Filters

There are many levels of filters when viewing reports:

- **Global Filters**. These remain in effect until they are removed in the Global Filters page.
- **Date/Time Filters**. These remain in effect until they are removed in the Calendar page.
- **Zoom Filters**. These remain in effect until they are removed in the Calendar page.
- **Report Filters**. These are set per report in the profile config, and cannot be removed from the web interface.
- **Report Element Filters**. These are set per report-element in the profile config, and cannot be removed from the web interface.

All these filters are "anded" together; i.e., an item is in the total filter set if it is selected by the Global Filters AND by the Date/Time Filters AND by the Zoom Filters AND by the Report Filters AND by the Report Element Filters. For instance, if the Global Filters show events during 1am-2am, and the Zoom Filters show events on January 1, then the table will show event from January 1, during 1am-2am.

Power User Techniques

Power User Tips

This page describes some of the faster, more powerful, and less intuitive ways of using Sawmill. Power users may find themselves using these techniques, rather than the friendlier but slower graphical interface. These techniques involve direct editing of Sawmill text-based profile options, and will be most easily mastered by users already comfortable with command line programs. You may want to familiarize yourself with the **Configuration Options** before reading this page.

Using Sawmill from the Command Line

Sawmill can be directly invoked from the command line (see **The Command Line**).

For instance, you may prefer to build and update your profile databases from the command line, avoiding the use of the web interface. To rebuild a profile database from the command line, do this:

```
sawmill -p {profile-name} -a bd
```

(replacing {profile-name} with the name of your profile), or update it using

```
sawmill -p {profile-name} -a ud
```

`rfcf` is short for `profile`; see **Profile to use**. `cm` is short for `action`. You can then browse that database from the web browser interface. If you'd prefer to generate offline HTML for the profile, you can do this:

```
sawmill -p {profile-name} -a grf -rn {reportname}
```

You can add any options to the command line that you can add to a profile file (and vice versa), so the Sawmill command line provides a great deal of flexibility. See **Configuration Options** for information on the available options.

Many command-line actions are available, including CSV export, HTML file generation, HTML email, and actions to build a database in stages. See **Action** for a list of all available actions.

Creating and Editing Profile Files

Sawmill stores profile options in profile files (see **Configuration Files**), in the profiles folder of the LogAnalysisInfo folder.

Profile files are structured in groups of options, with subgroups inside some groups. For instance, a profile might start like this (for simplicity, only some options are listed):

```
myprofile = {
  database = {
    options = {
      database_type = "internal"
      automatically_update_when_older_than = "0"
      lock_database_when_in_use = "true"
      prompt_before_erasing_database = "true"
    } # options
    tuning = {
      hash_table_starting_size = "4096"
      hash_table_expansion_factor = "2"
    } # tuning
  } # database
  ...
```

This profile is named "myprofile", and the first group shown the database group, containing all database options for the profile. Within that group, there are groups for database options ("options") and database tuning ("tuning"). You can edit this file with a text editor to change what the profile does -- all options available in the graphical interface are also available by editing the text file. Advanced users also often write scripts which edit or create profile files automatically, and then call Sawmill using the command line to use those profiles. Some advanced users do most of their profile editing with a text editor, rather than using the graphical interface.

Of course, you can still edit the profile from the graphical interface whenever you want, even to make modifications to profiles you have changed with a text editor. However, Sawmill's web browser interface will recreate the profile file using its own formatting, so don't use it if you've added your own comments or changed the text formatting!

Overriding Profile Options from the Command Line

On the command line, you can modify these options by listing the groups, separated by dots. For instance, if you wanted to use a hash table size of 8192, you could add this to the command line:

```
-database.tuning.hash_table_starting_size 8192
```

Command line options are listed with a dash followed by the name or shortcut of the option; followed by a space and the option value. Any profile options can be overridden in this way. Command-line overrides persist only for the duration of that command-line action -- they do not result in permanent modifications of the profile file.

The Config Page

The Profile Editor provides a graphical interface for editing most aspects of profiles, including the log source, log filters, database fields, and other options. By clicking "Edit Config" while in the profile list, or clicking "Config" while in the statistics, takes you to the Profile Editor.

The Config page is available only with Professional or Enterprise licensing; the functionality of this page is not available in Sawmill Lite.

It is also possible to edit profile options manually using a text editor -- see **Power User Techniques** for more information.

The Profile Editor is divided into categories; clicking a category name at the left of the screen takes you to that category. The categories are listed below.

Log Source Editor

This category provides an editor for the log sources for this profile. A log source specifies the location of log data to analyze in this profile, or the method of acquiring the log data.

Log Filters

Log Filters perform translations, conversions, or selective inclusion ("filter out") operations. For instance, a log filter could be used to reject (exclude) all log entries from a particular IP, or all log entries during a particular time. Log Filters could also be used to convert usernames to full names, or to simplify a field (for instance by chopping off the end of a URL, which is sometimes necessary to analyze a large proxy dataset efficiently).

Log Filters are also used for some log formats (including web log formats) to differentiate "hits" from "page views" based on file extensions. For instance, GIF files are considered "hits" but not "page views", so the default filters set the "hit" field to 1 and the "page view" field to 0 for GIF hits. The same method can be used for any profile, to perform any kind of categorization (e.g. external vs. internal hits for a web log).

Log Filters are written in **The Configuration Language**, which provides full programming language flexibility, including the use of if/then/else clauses, and/or/not expressions, loops, and more.

Reports Editor

This editor lets you create and edit reports which appear in the statistics.

The 'Rebuild Database' Button

This button appears at the top of the profile config editor. Clicking it rebuilds the database from scratch.

The 'Update Database' Button

This button appears at the top of the profile config editor. Clicking it updates the database, by adding any new log data in the log source (data which is in the log source but not in the database).

Configuration Options

Sawmill is highly configurable and can be customized for each profile. Global options are stored in the preferences; options specific to a particular profile are stored in each profile file. For a full list of available options, see **All Options**.

Each profile has its own set of options which controls what that profile does; it specifies the location of the log data, the log format, the database parameters, the reports to display, and more. A profile's options can be changed from the web browser interface using simple graphical forms and controls.

Options can also be used on **The Command Line**, or in **Configuration Files**.

The Command Line

The Sawmill command line can accept a wide variety of options, including any preference options and any options which can be put in a profile file. See **All Options** for a list of all available options.

The "Extra Options" line of the Scheduler accepts all options available from the command line, so this section applies to Extra Options also.

Every option has a location in the configuration hierarchy; e.g., the page header for the profile "MyProfile" is at profiles.myprofile.statistics.miscellaneous.page_header. This means that it's an option in the "miscellaneous" group of the "statistics" group of the "myprofile" profile (myprofile.cfg) in the "profiles" folder of LogAnalysisInfo folder.

Once you know the full name of the option, you can use it on the command line; so in this case you could add this to the command line:

```
-profiles.myprofile.statistics.miscellaneous.page_header "SOME HEADER"
```

to override that value of that option for the duration of the command line action. If you have specified a `-p` option on the command line (as you usually must), you can also shorten the option to this:

```
-statistics.miscellaneous.page_header "SOME HEADER"
```

Most options also have shortcuts, and if you know the shortcut (you can find it out from the option documentation, e.g., **Header text** in this case, which shows the shortcut for this option is `ph`), you can use that on the command line, like this:

```
-ph "SOME HEADER"
```

For a complete list of all options and shortcuts, see **All Options**. In most cases, the shortcut is the first letter of each word in the option name (e.g. ph for page_header), but there are a few exceptions where non-standard shortcuts were required because two options would have had the same shortcut. Some options also have no shortcuts; in that case, the full option name must be used instead.

Command line options can be specified by just typing them after the executable name (on Windows, make sure you use the command line executable, sawmill) on the command line as shown below. To improve ease of reading, a hyphen (`-`) can be added to the beginning of any profile option.

Below is a sample command line which checks the log data from a profile, and adds any new log data into the existing database for that profile (see **Databases**).

```
sawmill -p myconfig -a ud
```

The `-p` option (**Profile to use**) specifies the profile name; the `-a` option (**Action**) specifies the action. Most command lines include a `-p` and a `-a`. See **Power User Techniques** for more command line examples.

Configuration Files

You can usually avoid dealing with profile files by using Sawmill through the web browser interface. You only need to know about profile files if you want to edit them directly which is usually faster than using the web interface, use them from the command line, or if you need to change options which are not available through the web interface.

All Sawmill options are stored in text files called *configuration files* (or *profile files* if they contain the options of a particular profile). Configuration files use a very simple format; each option is given in the format

```
name = value
```

and options can be grouped like this:

```
groupname = {
  name1 = value1
  name2 = value2
  name3 = value3
} # groupname
```

Within this group, you can refer to the second value using the syntax `groupname.name2`. Groups can be within subgroups like this:

```
groupname = {
  name1 = value1
  subgroupname = {
    subname1 = subvalue1
    subname2 = subvalue2
  } # subgroupname
  name3 = value3
} # groupname
```

Hash characters (#) are comment markers; everything after a # is ignored, until the end of the line. Multiline comments can be created using {# before the command and #} after it. In this case, the subgroup name has been listed as a comment on the closing bracket; this is customary, and improves legibility, but is not required. In this case, *subvalue2* can be referred to as `groupname.subgroupname.subname`.

There are no limits to the number of levels, the number of values per level, the length of names or labels, or for anything else.

In addition to groupings within a file, groupings also follow the directory structure on the disk. The LogAnalysisInfo folder is the root of the configuration hierarchy, and files and directories within it function exactly as though they were curly-bracket groups like the ones above. For instance, the preferences.cfg file (cfg stands for configuration group) can be referred to as `preferences`; the server group within preferences.cfg can be referred to as `preferences.server`, and the web_server_port option within the server group can be referred to as `preferences.server.web_server_port`. So for instance, in a web server command line you can change the default port like this:

```
sawmill -ws t -preferences.server.web_server_port 8111
```

If you happen to know that the shortcut for web_server_port is wsp, you could also shorten this to:

```
sawmill -ws t -wsp 8111
```

In this example, -ws t is required to tell Sawmill to start its server.

Through this type of hierarchical grouping by directories within LogAnalysisInfo, and by curly-bracket groups within each configuration file, all configuration options in the entire hierarchy can be uniquely specified by a sequence of group names, separated by dots, and ending with an option name. All options in Sawmill are specified in this way, including profile options, preferences, language module (translation) variables, users, scheduling options, documentation, spider/worm/search engines information, command line and internal options, and more.

Sawmill creates a profile file in the profiles subfolder of the Sawmill folder when you create a profile from the graphical interface (see **The Administrative Menu**). Profile files can also be created by hand using a text editor, though the large number of options makes this a difficult task to do manually -- it is best scripted, or done by copying an existing profile file and editing it. To use files as profile files, you will need to put them in the profiles folder.

Any profile which can be specified in a profile file can also be specified in **The Command Line** by using the same profile options. Command line syntax is longer if full profile names are used, because each option on the command line must be specified using the full `group1.group2.group3.option`, when in the profile it appears only as `option` (within the groups). However, most options have shortcuts which can be used instead; see the option documentation for each option's shortcut (**All Options**),

To see a sample profile file, use the web browser interface to create a profile, and then examine the file in the profile folder.

Setting up Multiple Users (ISP Setup)

Internet Service Providers (ISPs) can use Sawmill to publish statistics for many sites to the sites' owners. This is the standard mode of operation for an ISP -- you want to provide all of your customers with statistics, with each customer seeing only statistics for their own site, and no other sites. Non-ISP users of Sawmill may also need a similar setup to provide statistics for different sites, or sections of the same site, to several different people. Sawmill makes this easy.

In a nutshell, what you'll do is create a profile for each user using the Create/Update Many Profiles feature. Then in the Users editor, you create a separate user for each user who should access statistics, and set it up so that user can access only those profiles (often, just one profile) that belongs to them. You'll set up each profile to process only the relevant log data (that user's log data). Finally, you'll deliver the statistics to the users as a URL, pointing to an active copy of Sawmill, and they'll use a browser to view their statistics.

Creating the First Profile

The first thing you'll need to do is to create a profile for each user. Start by creating one profile and setting it up the way you like. From the profiles list, click Create New Profile, and follow the instructions to create the profile. For an ISP (web hosting provider), you may want to use the name of the domain as the profile name. In this example, we'll assume that you're an ISP setting up a site for mysite.com, so you'll name the profile mysite.com.

If each user has separate log files, point this profile's Log Source to the appropriate log files. Otherwise, point the Log Source at the combined log data; you can skip over all the information about filters below.

If all the users' log data is combined into a single file, you'll need to create a log filter to separate out the hits for just this user. For instance, if you're logging 100 web sites to a single file, then you'll need to create a log filter to reject all hits except those from mysite.com. Exactly how this is done depends on how your server indicates which site each hit was on, but usually it's logged in a "server domain" field. The filter in that case is

```
if (!ends_with(server_domain, "mysite.com")) then "reject"
```

If your server logs the server domain information as part of the page field instead, you can use a similar filter:

```
if (!starts_with(page, "/mysite.com/")) then "reject"
```

For more information about log filters, see **Using Log Filters**.

If your server logs each domain to a separate log file, then you're in luck-- you don't need to do anything with filters. Just point your profile to the proper log files. In addition to being simpler to set up, this is also more efficient, because Sawmill will only have to read each log file once. Because of these advantages, it may be worth reconfiguring your web server to log each domain to a separate location.

At this point, you've done the most important steps in creating the first profile. All your other profiles will be based on this one, so if you have some options that you know you're going to use in every profile (for instance, a custom header or footer), you can set them now. You'll also be able to change options later, if you want.

Creating the Rest of the Profiles

Now you need to set up the Create/Update Many Profiles feature to automatically create all the rest of your profiles. Start by editing the file create_many_profiles.cfg in the miscellaneous folder of the LogAnalysisInfo folder. In this file you can specify the names and labels of each profile you wish to create, the name of the profile which should be duplicated to create them (the template you created above), and any other options you want to be different from the rest of the profiles. For instance, it is common for each profile to have its own log source, so you might override log.source.0.pathname to pull log data from a different pathname. Or you might override a particular log filter to filter the data differently for this profile. Any profile option can be overridden in the "changes" group of the profile group in the create_many_profiles.cfg file.

When you're done setting up the create_many_profiles.cfg file, you can create all the profiles in a single step with the following command line:

```
sawmill -dp templates.admin.profiles.create_many_profiles
```

and the following command line (Windows):

```
SawmillCL.exe -dp templates.admin.profiles.create_many_profiles
```

This will create all the profiles you've specified, apply the changes you've made to them and save them to the profiles list.

In the future, if you want to make a change to affect all profiles, modify the template profile, and rerun the command above to recreate the profiles.

Sending the Statistics to your Users

There are many ways of sending statistics to your users, but the main one is to send them a URL pointing to an active installation of Sawmill. The URL will allow them to view their statistics, but will not allow them to do anything else--they will not have access to other profiles, or to the administrative menu. For instance, if the profile name is mysite.com and Sawmill is running in CGI mode on the server stats.isp.com, then you can send your user the following URL:

http://stats.isp.com/cgi-bin/sawmill.cgi?dp+templates.profile.index+p+mysite.com+volatile.profile_name+mysite.com

In web server mode, the URL would be:

http://stats.isp.com:8987/?dp+templates.profile.index+p+mysite.com+volatile.profile_name+mysite.com

There are also ways to let them select their profile from a list; see **Security** for complete details.

You can also send any view by email to your users. The single-page summary view is a good choice, but you can send any view, or create your own. The view will be sent as HTML with embedded images, viewable in any current graphical email client, and it can contain links to a live version of Sawmill if you wish. You can set Sawmill to send views regularly using the Scheduler (see **Using the Sawmill Scheduler**).

Sawmill can also generate "static" HTML pages, which can then be uploaded to a web server, to provide statistics without a running copy of Sawmill. However, these pages are much less powerful than Sawmill's normal dynamic statistics, take up more space, and require an extra upload step, so it is usually a better idea to use the dynamic statistics. If you need static statistics, you can profile Sawmill to generate them regularly using the Scheduler.

Keeping the Statistics Up to Date

There are two ways to keep the databases up to date, manually or by using the Scheduler. Sawmill will keep the databases up to date even if you never explicitly set it up. If someone views the statistics when the database hasn't been built yet, Sawmill builds it; if they view the statistics when the database hasn't been updated for more than a day, Sawmill updates it. There are some advantages to updating the databases manually. Most importantly, the build/update step can take a long time, especially for large sites, and that delay may not be acceptable every time a user wants to see their statistics. You can eliminate the delay by updating the database regularly (usually once daily, in the middle of the night) so that the statistics are never more than a day old. If you use the Scheduler (see **Using the Sawmill Scheduler**); Sawmill can be configured to automatically update all databases every night. The Scheduler only works in web server mode, though; if you're using Sawmill in CGI mode (often a good choice for multi-user environments), you'll need to use an external scheduler like cron or Windows Scheduler to do the updates. You can update a database from the command line (or a scheduler) using a command line like this:

```
sawmill -p config-name -a ud
```

Or you can set up the schedules in the Sawmill Scheduler, and call Sawmill like this every minute to run any necessary schedules:

```
sawmill -scheduler
```

You can also run a copy of Sawmill in web server mode simultaneously, to only handle the schedules.

Maintaining the Profile

Once you've set up the multi-user environment, it will run without any maintenance on your part. However, if you want to change something about the profiles (change the header, add a new report, add new cross-references, etc.), you'll need to regenerate your profiles. The easiest way to do this is to continue to treat one of your profiles as a "template" profile -- make all your changes to that profile, and never change any other profile directly. Then, you'll only have to make each change only once to the template. Once your template is working the way you like, you can go back to the Create/Update Many Profiles field, paste in the profiles list you saved from the first time you used it, and click the button to recreate all the profiles from the template. Create/Update Many Profiles does not change or delete the databases for the profiles; it only changes the profile options. So you can safely Create/Update Many Profiles to create profiles even if the databases exist; the data will not be lost.

Creating and Editing Profiles by Hand

If you use Sawmill to manage a large number of profiles, you may find the web configuration interface inadequate to your needs. For instance, you may want to be able to create multiple profiles from the command line with a single command, without having to run a web browser or manually enter the information. In cases like these, you can create profiles "by hand," by creating your own profile files.

Several other documentation sections discuss the use of profile files; see **Configuration Files**, **Configuration Options**, and **Power User Techniques**. In general, you will want to create profiles using a text file editor or a script, and use them from the command line with the **Profile to use** option.

It's often most convenient to create a "template" profile using the web browser interface. That profile can then be duplicated and modified manually when a new, similar one is needed.

Profile files must be in the "profiles" subfolder of the LogAnalysisInfo folder.

When running from the command line, you will usually want to choose which action Sawmill should perform with the **Action** command.

Log Files

Sawmill is a log file analysis tool. It reads one or more log files, and generates a graphical statistical report of a chosen aspect of the contents of the log data.

Sawmill can handle a wide range of log formats. Hundreds of formats are supported by default (see **Supported Log Formats**), and others can be supported by creating a new log format description file (see **Creating Log Format Plug-ins (Custom Log Formats)**). If the format of the log data you wish to analyze is not supported, and the log format is the standard format of a publicly-available device or software, please send mail to **support@flowerfire.com** and we will create the log format description file for you. If it's a custom format, we can create the plug-in for you, for a fee.

Databases

Sawmill uses a database on your local hard drive or server to store information about log data. The database contains a compact version of the log data in the "main table", and a series of secondary tables which provide hierarchy information and improve performance of some queries. When a new log entry is read, the information contained in that entry is added to the database. To generate a new statistics page, the information needed is read from the database.

Multiple filters can be used to query data from the database and generate reports. For instance, it is possible in a virus log filter to show only the source IPs for a particular virus, and for a web log it's possible to see the pages hit by a particular visitor. In general *any* combination of filters can be used; if it's possible to create complex and/or/not expressions to zoom in on any part of the dataset.

For large datasets, it can be slow to query data directly from the main table. Query performance for some types of tables can be improved using cross-reference tables, which "roll up" data for certain fields into smaller, fast-access tables. For instance, for a web log, you can create a cross-reference table containing page, hit, and page view information; the table will pre-compute the number of hits and page views for each page, so the standard Pages report can be generated very quickly. See **Cross-Referencing and Simultaneous Filters** for more information.

The **Database folder** option specifies the location of the database on disk; if the option is blank, Sawmill stores the database in the Databases folder, in the LogAnalysisInfo folder, using the name of the profile as the name of the database folder.

New log data can be added to the database at any time. This allows a database to be quickly and incrementally updated, for instance, every day with that day's new log entries. This can be done from the web browser interface by using the Update Database option in **The Config Page**. A command line (see **The Command Line**) which would accomplish the same thing is

```
sawmill -p config-file -a ud
```

If your log files are very large, or if your database is extensively cross-referenced, building a database can take a long time, and use a lot of memory and disk space. See **Memory, Disk, and Time Usage** for information on limiting your memory and disk usage, and increasing the database build speed.

A number of advanced options exist to fine-tune database performance. To get the most out of the database feature, you may want to adjust the values of the database parameters.

The Configuration Language

Salang is the programming Sawmill language, which is used throughout Sawmill for many purposes from displaying pages like this one, including administrative or statistics pages, to writing log filters.

Salang is similar in some ways to perl, and borrows syntactic elements from perl, C, and other languages. Because Salang code is intended to be written inline, rather than as a stand-alone program, it deviates from perl and C in various ways, including the use of parentheses instead of brackets to group statements and expressions. Programmers acquainted with perl or C will find themselves in familiar territory with Salang.

Operators

Operator	Purpose
==	Compares two numbers; true if they are equal; e.g. 1 == 1 is true.
!=	Compares two numbers; true if they are not equal; e.g. 1 != 1 is false.
<=	Compares two numbers; true if the left number is less than or equal to the right; e.g. 1 <= 2 is true, and so is 1 <= 1.
>=	Compares two numbers; true if the left number is greater than or equal to the right; e.g. 2 >= 1 is true, and so is 1 >= 1.
<	Compares two numbers; true if the left number is less than the right; e.g. 1 < 2 is true, but 1 < 1 is false.
>	Compares two numbers; true if the left number is greater than the right; e.g. 2 > 1 is true, but 1 > 1 is false.
eq	Compares two strings; true if they are equal; e.g. "a" eq "a" is true.
ne	Compares two strings; true if they are not equal; e.g. "a" ne "a" is false.
le	Compares two strings; true if the left string is lexically less than or equal to the right; e.g. "a" le "b" is true, and so is "a" le "a".
ge	Compares two strings; true if the left string is lexically greater than or equal to the right; e.g. "b" ge "a" is true, and so is "a" ge "a".
lt	Compares two strings; true if the left string is lexically less than the right; e.g. "a" lt "b" is true, but "a" lt "a" is false.
gt	Compares two strings; true if the left string is lexically greater than the right; e.g. "b" gt "a" is true, but "a" gt "a" is false.
or	True if either left or right values, or both, are true; e.g. true or true is true; true or false is true.
and	True if both left and right values are true; e.g. true and true is true; true and false is false.
+	Adds the right value to the left value; e.g. 1+2 is 3.
-	Subtracts the right value from the left value; e.g. 2-1 is 1.
*****	Multiplies the right value and the left value; e.g. 2*3 is 6.
%	Modulo (division remainder) operation on the left value by the right value; e.g. 5%2 is 1 and 6%2 is 0.
/	Divides the left value by the right value; e.g. 12/4 is 3.
+=	Adds the right value numerically to the left variable; e.g. x += 1 adds 1 to x.
-=	Subtracts the right value numerically from the left variable; e.g. x -= 1 subtracts 1 from x.
++	Adds 1 numerically to the left variable; e.g. x++ adds 1 to x.
--	Subtracts 1 numerically from the left variable; e.g. x-- subtracts 1 from x.
.	Concatenates the right string to the end of the left string; e.g. "a"."b" is "ab".
.=	Concatenates the right value to the left variable; e.g. x .= "X" concatenates "X" to the end of x.
=	Assigns the right hand side to the left hand side; e.g. x = 1 assigns a value of 1 to the variable x.
!	Performs a boolean negation ("not" operation) of its unary parameter; e.g. !true is false, and !false is true.
not	Same as !
matches	True if the left value matches the wildcard pattern specified by the right value.
matches_regexp	True if the left value matches the regular expression specified by the right value.
within	Used in report filtering; true if the value of the database field specified by the left value is within the value specified by the right value; e.g. page within '/directory/'.

session contains	Used in report filtering; selects only sessions where the right value matches the session page field of some event in that session (as a wildcard expression). There is no left value, so the syntax is e.g., "(session contains '*/thanks.html')".
$	Treats its unary string parameter as a variable name, and evaluates the value of the variable; e.g. if the value of the variable named "somevariable" is 1, then the value of the expression $("somevariable") is 1. Important: This uses the *value* of the expression immediately after it as the name of the variable, so if variable x has value "valueX" then $x means the same as $("valueX"); i.e. it is the value of the variable valueX, *not* the value of the variable x. To get the value of the variable x, just use x, not $x.

Quotes and escapes

Single quotes ('), double quotes (") and backticks (`) can be used as quotes. Quotes are "smart"--you can use one type of quotes within another type of quotes, e.g. v = "someone's value". If you need to use the same types of quotes, you can use a backslash, e.g. v = 'someone\'s value'. Backslashes can be used to escape other characters, including dollars signs in values, to prevent them from being treated as variable evaluation operators.

The Configuration Hierarchy, and Configuration Nodes

Variables and other data are stored in the configuration hierarchy, a collection of data which exists on disk and partially in memory. The configuration hierarchy is a group of "nodes", each with a single value. Any node can also contain other nodes (subnodes). Each node has a type ("bool", "int", "float", "string", or "node"), and a value. Salang expressions can get values from the hierarchy (e.g. "internal.verbose" is an expression which gets the value of the "verbose" subnode of the "internal" node), or set them (e.g. "internal.verbose = 1;").

Configuration nodes are referenced using a dot-separated (.) sequence of names, which functions like a pathname. Configuration nodes are grouped hierchically either using subfolders, or by using configuration group files (.cfg files). The .cfg files have additional hierarchical groupings within them, specified by curly-bracketed groups. For instance, the node name "rewrite_rules" refers to the rewrite_rules folder of LogAnalysisInfo, and the node name "rewrite_rules.server_responses" refers to the server_responses.cfg file, in the rewrite_rules folder of LogAnalysisInfo. The server_responses.cfg file looks like this:

```
server_responses = {
  100 = {
    regexp = "100"
    result = "Continue"
  }
  ...
  200 = {
    regexp = "200"
    result = "Successful Transfer"
  }
  ...
} # server_response
```

The series of three of dots (...) as shown above are ellipses and they indicate places where part of the file has been omitted for brevity, so there is a "100" node inside the server_responses node, and there is also a "200" node there. Therefore, the node name "rewrite_rules.server_responses.100" refers to the 100 node, and "rewrite_rules.server_responses.200" refers to the 200 node. Within the 100 node, there are two nodes, "regexp" and "result"; so "rewrite_rules.server_responses.100.regexp" refers to the regexp node in 100, and "rewrite_rules.server_responses.200.regexp" refers to the regexp node in 200.

Directories are treated equivalently to .cfg files in the configuration hierarchy. In both cases, they represent nodes, and can contain other nodes. In the case of directories, subnodes appear as subdirectories, or as .cfg files in the directory, or as configuration value files (.cfv files) in the directory; in the case of .cfg files, subnodes appear within the file using curly-bracket syntax, as above.

As mentioned above, every node has a value. In the example above, the value of rewrite_rules.server_responses.200.regexp is "100", and the value of rewrite_rules.server_responses.100.result is "Continue". Even group nodes like rewrite_rules.server_responses.100, rewrite_rules.server_responses, and rewrite_rules can have values, but their values are typically empty (""); a node has *either* a value, or subnodes.

Nodes can be used like variables (they *are* variable). Nodes can be referred to by name, in log filters or other Salang code; for instance this:

```
echo(rewrite_rules.server_responses.100.result)
```

will print the value "Continue" to the standard output stream. They can also be assigned values:

```
rewrite_rules.server_responses.100.result = "Something Else"
```

Nodes are read from disk into memory automatically when they are accessed, and the in-memory version is used for the remainder of the process, and then discarded. Therefore, it is not generally necessary to read data from disk explicitly; instead, put it in a node and access it by name. In the example above, the value of rewrite_rules.server_responses.100.result will be *temporarily* changed to "Something Else"; only the in-memory version of rewrite_rules.server_responses.100.result will change, and the change will not be propagated to disk unless **`save_node()`** is used to write out the changes back out to disk.

Configuration value files (.cfv files) are straight text files whose entire content is considered to be the string value of the node. For example, the file LogAnalysisInfo/somedir/somefile.cfv whose *contents* is "HELLO" could be referred to as somedir.somefile; the value of that node is "HELLO".

Salang has a built-in type of `node` that refers to a configuration node; it is a pointer to that node. You can get the value of the node using **`node_value()`**. You can also get a particular subnode using **`subnode_by_name()`**. You can check for the existence of a subnode using **`subnode_exists()`** or use **`node_exists()`** with the full nodename as the parameter.

In addition to the LogAnalysisInfo directory, Salang also looks in other "search paths" to find nodes specified by name:

- If the node name begins with **lang**_, it will search for it in LogAnalysisInfo/language/{current} where {current} is the current language. For instance, if the current language is "english", then the node "lang_stats.menu.groups.date_time_group" refers to the date_time_group node, in the groups node, in the menu node, in the LogAnalysisInfo/languages/english/lang_stats.cfg.
- If a profile is active (e.g. was specified with **-p** on the command line), it will search for the node in the profile .cfg file. For instance, if Sawmill is called with -p myprofile, then the node "database.fields.date_time" refers to the date_time database field, and "database_fields.date_time.label" refers to the label for the date_time database field.

- In log filters, if the node name is exactly the name of a log field, then the log field value is used. This is not actually a node, but functions as one for the purpose of reading and writing its value; you can refer to it like a normal Salang variable in most cases (e.g., "file_type = 'GIF'").
- If a local variable has been declared in Salang, it can be referred to by name. For instance, you can write:

  ```
  int i;
  i = 3;
  echo(i);
  ```

  to define and refer to a local variable node i. Local variables do not have full nodenames; they are free-floating in memory and cannot be saved to disk, they are refered to as not being "rooted" in LogAnalysisInfo like most other nodes. They can only be referred to by their short nodenames (e.g. "i").
- Inside **subroutines**, the subroutine parameters can be referred to by name. For instance, the following subroutine, which adds two numbers, refers to its parameters x and y directly:

  ```
  subroutine(add(int x, int y), (
    x + y
  ));
  ```

  Subroutine parameters are similar to local variables (above) in that they are not rooted, and cannot be written to disk.

- The special variables $0, $1, $2, etc. refer to the matched subexpressions of the last call to **`matches_regular_expression()`**.

Report Filter Syntax

Filters used in reports take a special variant syntax that allows only certain operations. Subroutines (described below) are not allowed, and only database field names are allowed as variables. Only strings are allowed as constants. The <, >, <=, and => operators are permitted for the date_time field only. The "within", "matches", and "matches_regexp" operators are permitted for any field. Expressions can be combined using "and", "or", and "not"; arbitrary parentheses are permitted to allow any combinations. **No other syntax is permitted.**

Log Filter Syntax

Log filters can use all syntax described on this page, and also support a few extra variables. Specifically, log filters can refer to log fields by name, so a reference to date_time in a log filter is a reference to the value of the date_time field in the log entry that is currently being processed. This can be used either to get or set values; e.g. "if (page eq '/index.html') then 'reject'" checks the current log entry's page field to see if it's "/index.html", and rejects the log entry if it is; and "page = '/index.html'" sets the page field of the current log entry to "/index.html".

Log filters can also use the special function current_log_line(), whose value is the entire current line of log data.

Types

Each configuration node has a type, which specifies how its data is stored internally. Possible types are:

- `string`: The data is an arbitrary string value like "hello" or "Bob".
- `int`: The data is an integer value like 15 or 49.
- `float`: The data is a floating point value like 1.4 or 18874.46567.
- `bool`: The data is a boolean value (true or false).
- `node`: The data is a reference (pointer) to a node in the configuration hierarchy.

Types are primarily useful for performance optimization. Salang is a weakly typed language, so it will allow any value to be assigned to any variable without warning or complaint; for instance assigning 134 to a `string` variable will result in "134" in the `string` variable, or assigning "134.5" to a `float` variable will result in the floating point value of 134.5 in the `float` variable. However, these types of conversions can be slow if they are performed many times, so if you know a variable is only going to hold and manipulate floating point values, you should use a `float` type rather than a `string` type.

The `node` type is particularly useful for performance optimization, since it prevents the need for node lookups; e.g. setting the value of a node explicitly with "`a.b.c.d = 0;`" requires a series of expensive lookups, as node "a" is looked up in the configuration root, and then "b" inside that, and then "c", and then "d"-- but if a `node` variable N already points to a.b.c.d, then "N = 0;" is a very fast operation that does the same thing. `node` variables are particularly useful for `foreach` loops, and functions like subnode_value(), where they can be used to iterate over all subnodes without requiring any performance intensive string-to-node lookups.

Statements

if A then B else C
This statement evaluates the A section; if the value of A is true, then the value of the entire statement is the B section; otherwise, the value of the statement is the C section.

subroutine(A(param1, param2, ..., paramN), B)
This statement defines a subroutine A with N parameters. The subroutine can be called with the statement "A(p1, p2, ..., pN)". The value of the subroutine call will be the value of B, with the parameters p1...pN plugged into the variables $param1.. $paramN before B is evaluated. The value of a subroutine declaration is empty.

foreach I N B
This statement is used to iterate over the subnodes of a particular node in the Salang hierarchy. It iterates over all values of node N, setting I to the full nodepath each iteration, and then evaluating expression B. The value of this expression is the concatenation of the values of all the evaluations of B.

for (I; C; E)
This repeats an expression 0 or more times. The expression I is evaluated to initialize the loop. The expression C is evaluated, before each iteration, and if the result is false, then the iteration does not complete. If C is true, the E is evaluated. The value of this expression is the concatenation of the E values.

while (C) E;
This repeats an expression 0 or more times. The expression C is evaluated before each iteration, and if the result is true, then E is evaluated. This continues until C is false. The value of this expression is the concatenation of the E values.

next
This statement goes immediately to the next iteration of the immediately enclosing loop.

last
This statement immediately terminates execution of the immediately enclosing loop, and continues execution after the end of the loop.

Built-in subroutines

get_file_type_from_url(string U)
The value of this statement is the file extension of the URL U; e.g., get_file_type_from_url("http://something/file.gif") is "GIF".

get_log_field(string N)
Returns the value of the log field N, of the current log entry.

node_exists(string N)
Returns true if the node specified by the nodename N exists; false if the node does not exist.

node_name(node N)
Returns the name of the node N.

num_subnodes(node N)
Returns the number of subnodes of the node specified by nodename N.

subnode_by_number(node N, int I)
Returns the Ith subnode of node N.

subnode_by_name(node N, string M)
Returns the subnode of node N whose name is M. If there is no subnode by that name, it creates one.

subnode_exists(node N, string M)
Returns true if there is a subnode named M in node N, false otherwise.

set_subnode_value(node N, string M, anytype V)
Sets the subnode named M of node N to the value V. Creates the subnode if it does not exist.

node_value(node N)
Returns the value of node N.

set_node_value(node N, anytype V)
Sets the value of node N to V; no value is returned.

node_type(node N)
Returns the type of node N as a string : e.g., "int", "float", "bool", "string", or "node".

set_node_type(node N, string T)
Sets the type of node N to T: e.g., "int", "float", "bool", "string", or "node".

delete_node(node N)
Deletes the node specified by nodename N. If the node is a profile, this also deletes the profile file.

insert_node(node P, string N, int I)
Inserts the node specified by nodename N into the node specified by nodename P, so that N ends up as the Ith subnode of P (i.e., it inserts N into P at position I). N is removed from its current location, and moved to the new one, so for instance if N is a subnode of O before this is called, it will no longer be a subnode of O afterwards ; i.e., it moves N, rather than copying it.

clone_node(node original_node, string clone_nodepath)
This makes a clone of the node *original_node*, and puts it at the node specified by *clone_nodepath* (which is created if it doesn't exist). The original node is unchanged, and after completion, the clone is an exact replicate of the original, except the name (which is computed from *clone_nodepath*).

sort(node N, string M, bool E)
Sorts the subnodes of the node specified by nodename N. The subnodes are sorted in an order specified by method M. M is a string of the format "field:F,T,A", where F is a field name (the name of a subnode found in each subnode of N; use "value" to use the value of each subnode of N directly), T is "integer", "float", "alphabetical", or "chronological" (which determines the sort type), and A is "ascending" or "descending" (which determine the direction of sort). If E is true, then variable are expanded before sorting; if E is false, then the sort is done on the literal values, without variable expansion.

create_profile(string N)
This creates a new profile with name N. It pulls various variables from the node hierarchy to set up the profile, the nodes are set by the "create profile" interview.

format(anytype V, string T)
This formats the value V according to the format type T. T can be "integer", "page", "float", "two_digit_fixed", "bandwidth", "date_time", "hostname", "duration", "duration_compact", or "duration_milliseconds". If the first character of T is a % character, the result will be formatted as a double-precision floating point number using printf-style formatting, e.g. '%.1f' will print the value in floating point format with one digit after the decimal place.

image(string N, int W, int H)
The value of this subroutine is an HTML image tag which displays the image specified by filename N, with width W and height H. The image file should be in the temporary directory in CGI mode, or in the picts folder, which is in the WebServeRoot folder of the LogAnalysisInfo folder, in web server mode.

create_image(int *width*, int *height*)
This creates a new image canvas of the specified image and height, and returns the image ID for use in drawing functions. It returns the image ID for use in drawing functions.

allocate_image_color(string *imageid*, int *red*, int *green*, int *blue*)
This allocates a new color for use in drawing on image *imageid* (created by create_image()), with the specified red, green, and blue components (0 to 255). It returns the color value for use in drawing functions.

add_text_to_image(string *imageid*, string *color*, int *x*, int *y*, string *text*, string *direction*)
This draws text on image *imageid* (created by create_image()), with the specified color (created with allocate_image_color())

at the specified x and y position, in the directory specified by *direction* ("up" or "right"). The upper left corner of the text will be anchored at (*x*, *y*) when drawing horizontally; the lower left corner will be anchored there when drawing vertically.

add_line_to_image(string *imageid*, string *color*, int *x1*, int *y1*, int *x2*, int *y2*)
This draws a line on image *imageid* (created by create_image()), with the specified color (created with allocate_image_color()) from the point (*x1*, *y1*) to the point (*x2*, *y2*).

add_polygon_to_image(string *imageid*, string *color*, node *points*, bool *filled*)
This draws a polygon on image *imageid* (created by create_image()), with the specified color (created with allocate_image_color()). The vertices of the polygon are specified in order in the subnodes of *points*; each subnode must have an x and a y value specifying the location of that point. The polygon will be a single pixel border if *filled* is false, or filled if it is true.

add_rectangle_to_image(string *imageid*, string *color*, int *xmin*, int *ymin*, int *xmax*, int *ymax*, bool *filled*)
This draws a rectangle on image *imageid* (created by create_image()), with the specified color (created with allocate_image_color()) and minimum x/y dimensions. The rectangle will be a single pixel border if *filled* is false, or filled if it is true.

write_image_to_disk(string *imageid*, string *pathname*)
This writes the image *imageid* (created by create_image()) to disk as a GIF file, to the location specified by *pathname*.

img_src(string N)
The value of this subroutine is the value of an HTML src section of an image tag; i.e. it's intended to be used right after src= in an image tag. The image file should be in the temporary directory in CGI mode, or in the picts folder, which is in the WebServeRoot folder of the LogAnalysisInfo folder, in web server mode.

fileref(string F)
This converts a local filename from the WebServerRoot directory to a URL that refers to that file. The filename should be in partial URL syntax, e.g. "picts/somefile.gif" to refer to the file somefile.gif in the picts directory of WebServerRoot. If necessary, it may copy the file; e.g., in CGI mode to the temporary directory. All references to files in WebServerRoot should be done through this function to ensure correct functioning in CGI mode.

save_changes()
This causes any changes made to the configuration hierarchy since the last save to be saved to the disk version of the hierarchy.

save_node(node N)
This saves node N to disk. For instance, if a N is "somenode.somechild.somevar", this will be saved to a file called somevar.cfg, in the somechild folder of the somenode folder of the LogAnalysisInfo folder. The format used is normal configuration format. This makes the node *persistent*, because any future access to that node; e.g., referring to somenode.somechild.somevar in an expression, even from a later process after the current one has exited, will automatically load the node value from disk. So by using save_node(), you can ensure that any changes made to that node will be a permanent part of the configuration hierarchy, and any values you have set will be available for use to all future processes and threads.

logout()
This logs the current user out (clearing the cookies that maintained their login), and takes them back to the login page (or the logout URL, if one is specified).

expand(string M)
This expands variables and expressions in M, and returns the expanded value. For instance, expand("$x") will return the value of the variable x, and expand("$x > $y") will return "12 > 10", if x is 12 and y is 10.

convert_escapes(string M)
This converts percent-sign escape sequences in M (e.g. converting %20 to a space), and returns the converted value. For instance, expand("some%20string") will return "some string".

hex_escape(string *str*)
This returns an escaped version of *str* where all "unusual" characters have been converted to "__HexEsc__XX", where XX is the hexadecimal code of the character. "Unusual" characters are all characters which are not letters, numbers, underbar (_), dash (-), period (.), forward slash (/), or colon (:).

set_active_profile(string *profile_name*)
This sets the active profile to *profile_name*. The specified profile will be searched when looking up variables; e.g. the variable database.fields.date_time.label (which would normally be invalid, if no profile were active) will be treated as a local nodename within the specified profile, and be resolved as though it were profiles. *profile_name*. database.fields.date_time.label.

debug_message(string M)
This converts variables and expressions in M, and displays the resulting value to the standard debug stream; i.e., to the console. This is useful for debugging Salang code.

echo(string M)

This outputs the string M (without any conversion) directly to the standard output stream. This is useful for debugging Salang code.

error(string M)
This throws an error exception, with error message M. The error will be reported through the standard Sawmill error reporting mechanism.

node_as_string(node N)
This converts the node N as a string value (like you would see it in a profile file). The value of this expression is the string value of N.

node **string_to_node(*string* s)**
This converts the string *s* to a node, and returns the node. This is similar to writing the string to a .cfg file, and reading it in as a node. For instance, string_to_node("x = { a = 1 b = 2 }") will return a node whose name is x, which has two subnodes a and b, with values 1 and 2.

autodetect_log_format(node *log_source*, string *result*, string *id*)
This autodetects the log format from the log source *log_source*, and puts the result in the node specified by *result. id* is an identifier which appears in the task info node for this process, allowing another process to tap into the progress for this process. This is used internally by the profile creation wizard.

get_directory_contents(string P, string R)
This gets the contents of the directory at pathname P, and puts the result in the node specified by R. R will contain a subnode for each file or subdirectory, and within each subnode there will be a `name` node listing the name of the file or directory and an `is_directory` node which is true for a directory or false for a file; there will also be a `size` node if it's a file, listing the file size in bytes.

get_file_info(string P, string R)
This gets information about the file or directory at pathname P, and puts the result in the node specified by R. R.exists will be true or false depending on whether P exists; R.parent will be the full pathname of the parent of P; R.type will be "file" or "directory" depending on whether P is a file or directory; R.filename will be the filename or directory name of P.

file_exists(string P)
The value of this is true or false, depending on whether the file or directory at pathname P exists.

delete_file(string *pathname*)
This deletes the file whose location is specified by *pathname*.

delete_directory(string *pathname*)
This deletes the directory whose location is specified by *pathname*, including all subdirectories and files.

get_files_matching_log_source(node L, string R)
This gets a list of files matching the log source in node L, and puts the result in the node specified by R. R will contain a subnode for each matching file; the value of the subnode will be the filename. This is used by the interface to implement Show Matching Files.

length(string S)
The value of this expression is the length of the string S.

substr(string V, int S, int L)
The value of this expression is the substring of the string V, starting at index S and of length L. The L parameter is optional, and if it is omitted, the value of the expression is the substring of V starting at S and continuing to the end of V.

split(string s, string divider, string resultnode)
This splits the string s on the divider specified in *divider*, and puts the resulting sections into the node specified by *resultnode*. For instance, split("Hello,you,there", ",", "volatile.splitresult") will set volatile.splitresult.0 to "Hello", volatile.splitresult.1 to "you", and volatile.splitresult.2 to "there".

starts_with(string S, string T)
The value of this expression is true if the string S starts with the value of the string T.

ends_with(string S, string T)
The value of this expression is true if the string S ends with the value of the string T.

contains(string S, string T)
The value of this expression is true if the string S contains the value of the string T.

replace_all(string S, string T, string R)
The value of this expression is the value of S after all occurrences of T have been replaced with R.

replace_first(string S, string T, string R)
The value of this expression is the value of S after the first occurrence of T has been replaced with R. If T does not occur in S, the value of this expression is S.

replace_last(string S, string T, string R)
The value of this expression is the value of S after the last occurrence of T has been replaced with R. If T does not occur in S, the value of this expression is S.

lowercase(string S)
The value of this expression is the value of S after all uppercase letters have been converted to lowercase.

uppercase(string S)
The value of this expression is the value of S after all lowercase letters have been converted to uppercase.

convert_charset(string *value*, string *fromcharset*, string *tocharset*)
This converts the string *value* from the charset *fromcharset* to the charset *tocharset*. It returns the converted version of *value*. Charset names are as documented for the GNU iconv conversion utility.

convert_local_code_page_to_utf8(string *value*)
This converts the string *value* from the local code page (local operating system charset) to the UTF-8 charset. It returns the converted version of *value*.

convert_utf8_to_local_code_page(string *value*)
This converts the string *value* from the UTF-8 charset to the local code page (local operating system charset). It returns the converted version of *value*.

get_search_engine_info(string *url*)
This examines *url* to see if it matches any of the search engine rules in the search_engines.cfg file in LogAnalysisInfo. If it does, it sets volatile.search_engine and volatile.search_phrase to the matching search engine and search phrase.

matches_regular_expression(string S, string R)
The value of this expression is true if the string S matches the regular expression R. If it matches, the variables $0, $1, $2, ... are set to the substrings of S which match the parenthesized subexpressions RE.

matches_wildcard_expression(string *str*, string *wildcardexp*)
The value of this expression is true if the string *str* matches the wildcard expression *wildcardexp*.

index(string S, string T)
The value of this expression is the index (character position) of the substring T in the string S. If T is not a substring of S, the value of this expression is -1.

last_index(string S, string T)
The value of this expression is the index (character position) of the *final* occurrence of substring T in the string S. If T is not a substring of S, the value of this expression is -1.

md5_digest(string S)
The value of this expression is the MD5 digest of the string S, as a 32-digit hexadecimal number.

get_license_info(node *licensenode*, node *resultnode*)
This looks at the Sawmill license key contained in *licensenode*, and extracts information about it, which it puts in the node pointed to by *resultnode*. The subnodes of the result are type, valid, valid64, valid65, addon, expiration, profiles, and users.

string **create_trial_license()**
This creates and returns a 30-day trial license key. Only one trial key can be generated per installation; after the first time, it will return an empty string.

node **license_string_to_license_node(string *license_string*)**
This converts a string format license key (*license_string*) into a node format license key. It returns the node key is returned. If the string is not valid, the node will not contain a checksum, and will have valid=false as a subnode.

display_statistics_filters(node F, string D)
The value of this expression is HTML which describes the Filters in node F, for database field D. This is used internally by the statistics interface to display Filters.

query_one_db_item(node F, string R)
This queries the numerical database field totals for the Filters F, and puts the result in the node R.

query_db_for_view(node V, string R)
This queries the database for statistics view V, and puts the result in R. The format of R depends on the type of the view V. Also, if volatile.csv_export is true, this computes a CSV version of the query result, and puts it in volatile.csv_export_result.

query_db_calendar(string R)
This queries the database to compute calendar information (which days are in the database), and puts the result in R. R contains a numerical year node for each year, and within that a numerical month node for each month, and within that a numerical day node for each day in the database.

query_db_field_hierarchy(string dbfieldname, string nodename)
This queries the database to compute the hierarchy of the database field dbfieldname, and puts the result in nodename. The values of nodes in nodename match the values of items for that database field, and the hierarchical structure of nodename exactly matches the structure of the field.

dababase_itemnum_to_item(int *dbfieldnum*, int *itemnum*)
This returns the item value (item name) of item number *itemnum* in the database field *dbfieldnum*.

dababase_item_to_itemnum(int *dbfieldnum*, string *item*)
This returns the item number for the item *item* in the database field *dbfieldnum*.

db_item_has_subitems(string D, string I)
This checks whether item I of database field D has subitems in the current database. It returns true if it does, and false if it isn't (i.e. if it's a bottom-level item).

uniques(string *filter*, string *dbfieldname*, node *result*)
This gets the list of all values of the unique field specified by *dbfieldname*, for the filter set specified by the expression in *filter*, and sets a subnode of *result* to the value of the unique field, for each unique value. For instance, uniques("page within '/dir/'", "visitors", "v.uniques") will add a subnode of v.uniques for each visitor to the directory /dir/, and the value of the subnode is the IP or host of the visitor.

set_log_field(string N, string V)
This sets the value of the log field N of the current log entry to V.

include(node N)
This loads and processes the Salang code in the node specified by nodename N.

discard(anytype V)
The value of this expression is always empty (""). This is useful if you want to evaluate an expression V, but not use its value.

capitalize(string V)
This capitalizes the value V, using the capitalization rules in the language module.

pluralizes(string V)
This pluralizes the value V, using the pluralization rules in the language module.

char_to_ascii(string c)
This returns the integer ASCII code of the character c. c should be a one-character string.

ascii_to_char(int i)
This returns a one-character string containing the ASCII character whose code is i.

ip_address_to_int(string *ipaddr*)
This converts the IP address *ipaddr* to an integer (a 32-bit representation with 8 bits per octet).

convert_field_map(string F, string M)
This converts the value of the log field F in the current log line we're processing, using the map M. M is a |-divided list of A->B mappings; e.g., if M is "1->cat|2->dog|3->ferret", then a field value "2" will be converted to "dog".

collect_fields_using_regexp(string R, string F)
This matches the current log line we're processing against the regular expression R, and if it matches, it extracts the parenthesized values in R and puts them in the fields specified by F. F is a comma-separated list of fields, and should include *KEY*, which specifies the key of the collected log entry to modify. E.g., if F is "page,*KEY*,date_time,host", the second parenthesized subexpression will be used as the key, and that key will specify which collected entry to modify; then the page, date_time, and host fields of that collected entry will be set to the first, third, and fourth parenthesized sections.

collect_listed_fields_using_regexp(string *regexp*, string *divider*, string *separator*, string *field_names_map*)
This matches the current log line we're processing against the regular expression *regexp*, and if it matches, it uses the first parenthesized section in *regexp* and uses that as the key, and uses the second parenthesized section and uses it as the name/values list. Then it uses that, and its other parameters, to do the same as collected_listed_fields().

collect_listed_fields(string *key*, string *name_values_list*, string *divider*, string *separator*, string *field_names_map*)

This extracts log field values from *name_values_list*, which is a list of name/values pairs, and puts them in the collected entry

specified by *key*. Names and values are listed in *name_value_list* the format "name1*separator*value1*divider*name2*separator*value2*divider*name3*separator*value3", e.g. pairs are separate from each other by the value of *divider*, and each name is separate from its value by the value of *separator*. In addition, *field_names_map* can be used to convert field names; *field_names_map* is a pipe-separated (|-separated) list of values like "fieldname=newfieldname", and if any extracted field matches a fieldname value from *field_names_map*, it will be put in the newfieldname log field. If there is no map, or if nothing matches, then values will be put in the log field specified by the field name.

accept_collected_entry_using_regexp(string R)
This matches the current log line we're processing against the regular expression R, and if it matches, it extracts the first parenthesized value in R; using that value as a key field, it accepts the log entry corresponding to that key (as created by collect_fields_using_regexp()) into the database.

set_collected_field(string key, string log_field_name, string set_value)
This sets the collected field *log_field_name*, in the collected log entry specified by *key*, to the value *set_value*.

get_collected_field(string key, string log_field_name)
This returns the value of the collected field *log_field_name*, in the collected log entry specified by *key*.

rekey_collected_entry(string F, string T)
This changes the key of the collected entry with key F so its key is T instead.

generate_report_id(string P, string A)
This creates a new report ID, which can be used to generate a report with generate_report(). P is the profile name, and A is a list of any configuration options that need to be changed from the defaults in the report. The value of this function is the generated report ID.

generate_report(string P, string I)
This generates the report with id I, for profile P. Generation occurs in a separate task, so this returns immediately, while report generation continues in the background. Calls to the functions below can tap into the status of the report, and the final generated report. The value of this function is empty.

get_progress_info(string *taskid*, string *resultnode*)
This acquires progress information for a running task with task ID *taskid*. It populates the node specified by *resultnode* with detailed progress information. If *taskid* is not a valid task or there is no progress available for that task, *resultnode* will have a subnode exist with value false. The value of this function is empty.

cached_report_exists(string P, string I)
This checks if the report from profile P with id I has been completely generated, and is now cached. The value of this function is true if the report is cached, and false if it is not cached (never generated or generation is in progress).

display_cached_report(string P, string I)
This displays the cached report from profile P with id I. This will fail if the report is not cached-- call cached_report_exists() to check. The value of this function is the complete HTML page of the report.

delete_cached_report(string P, string I)
This deletes a cached report from profile P with id I. Future calls to cached_report_exists() will be false until this is regenerated with generate_report(). The value of this expression is empty.

verify_mysql_connection(string H, string U, string P)
This verifies that a connection can be made to the MySQL server with hostname H, username U, and password P.

get_database_info(string P, string N)
This gets various information about the database for profile P, and puts it in node N.

build_database(string P, string R)
This builds the database for profile P. If the database build is occurring as part of an attempt to view a report, the report ID should go in R; otherwise, R should be empty ("").

update_database(string P, string R)
This updates the database for profile P. If the database build is occurring as part of an attempt to view a report, the report ID should go in R; otherwise, R should be empty ("").

exec(string executable, node options, bool wait)
This runs the command-line program specified by executable, from the command line. If the *executable* option is an empty string, the main program executable is run (e.g. Sawmill runs itself from the command line). The options node contains the command line options; e.g., for each subnode of the options node, the *value* of that option (not the name, which is ignored) is a command line option. The *wait* option specified whether to wait for completion; if it is true, exec() will not return until the command is done; if it is false, exec() will return immediately, leaving the command running in the background. This returns the Process ID (PID) of the process if *wait* is false, or zero otherwise.

get_task_info(string N)
This gets various information about currently active tasks, and puts it in node N.

cancel_task(string *task_id*)
This cancels an active task with ID *task_id*. When this returns, the task has been cancelled.

sleep_milliseconds(int M)
This delays for M milliseconds before evaluating the next expression.

save_session_changes()
This saves any changes to the configuration hierarchy to the sessions file, so they will carry over to future pages.

write_file(P, S)
This writes string S to a file at pathname P. If the file does not exist, it is created; if it exists, it is overwritten. P is a local pathname in LogAnalysisInfo, using / as pathname dividers; for instance, to write to a file test.html in WebServerRoot in LogAnalysisInfo, use "WebServerRoot/test.html".

read_file(P)
This reads the contents of the file at pathname P. It returns the contents of the file as a string. P is either a full pathname, or a pathname local to the Sawmill executable location, using / as pathname dividers; for instance, if your LogAnalysisInfo folder is in the same folder as the Sawmill binary (which is typical), then to read from a file test.html in WebServerRoot in LogAnalysisInfo, use "LogAnalysisInfo/WebServerRoot/test.html".

send_email(string sender, string recipient, string message, string smtp_server)
This sends an email to sender from recipient, using SMTP server smtp_server. The "message" variable is the entire contents of the message, including mail headers.

now()
This returns the current time, as the number of seconds since January 1, 1970, Coordinated Universal Time, without including leap seconds.

localtime()
This returns the current time, as the number of seconds since January 1, 1970, in the GMT time zone.

normalize_date(string *date*, string *format*)
This computes the "normal" format (dd/mmm/yyyy) of the date in *date*, and returns the date in normal format. *date* is in the format specified by *format*, which may be any of the date formats in the list at **Date format**.

normalize_time(string *time*, string *format*)
This computes the "normal" format (dd/mmm/yyyy) of the time in *time*, and returns the time in normal format. *time* is in the format specified by *format*, which may be any of the time formats in the list at **Time format**.

date_time_to_epoc(string D)
This converts the string D, which is a date/time of the format 'dd/mmm/yyyy hh:mm:ss' (GMT) to an epoc time (seconds since January 1, 1970). It returns the epoc time.

epoc_to_date_time(int E)
This converts the integer E, which is a date/time value in EPOC format (seconds since January 1, 1970) to a string of the format 'dd/mmm/yyyy hh:mm:ss'. It returns string date/time.

date_time_duration(string D)
This computes the duration in minutes of the date_time unit value D, which is of the format 'dd/mmm/yyyy hh:mm:ss', where any value may be replaced by underbars to indicate the unit size, e.g. '01/Feb/2005 12:34:56' indicates a single second (so the value returned will be 1); '01/Feb/2005 12:34:__' indicates a single minute (so the value returned will be 60); '01/Feb/2005 12:__:__' indicates a single hour (so the value returned will be 3600); '01/Feb/2005 __:__:__' indicates a single day (so the value returned will be 86400); '__/Feb/2005 __:__:__' indicates a single month (so the value returned will be the duration of the month, in seconds); and '__/___/2005 __:__:__' indicates a single year (so the value returned will be the duration of the year, in seconds). The value of this format is used frequently in report filters; this function is useful for computing durations of filtered data.

clear_session_changes()
This clears any changes to the configuration hierarchy saved using save_session_changes(). I.e. this reverts to the last saved version of the configuration hierarchy saved using save_changes().

start_progress_meter_step(string O)
This starts a progress meter step with operation name O.

finish_progress_meter_step(string O)
This finishes a progress meter step with operation name O.

set_progress_meter_maximum(float M)
This sets the maximum progress meter value to M.

set_progress_meter_position(float M)
This sets the current position of the progress meter to P (out of a total value specified by set_progress_meter_maximum()). This also causes the progress meter to be updated if necessary; call this function regularly during long operations to ensure that progress occurs properly.

current_log_pathname()
Returns the pathname of the log file currently being processed (for log filters).

current_log_line()
Returns the entire current line of log data being processed (for log filters).

sin(float X)
This returns the sine of X.

cos(float X)
This returns the cosine of X.

tan(float X)
This returns the tangent of X.

asin(float X)
This returns the arc sine of X.

acos(float X)
This returns the arc cosine of X.

atan(float X)
This returns the arc tangent of X.

atan2(float X, float Y)
This returns the arc tangent of X/Y, using the signs of X and Y to compute the quadrant of the result.

compile(string *expression*)
This compiles the Salang expression *expression*, and returns a compiled version of it as a node.

evaluate(node *compiled_node*)
This evaluates a compiled node (return by compile()), and returns the value of the expression value.

Database Detail

When you first create a profile, Sawmill will ask you what kind of information you want to track in your profile. The values you choose determine what your initial profile settings are, including the database cross-references and the available views.

The options available depend on your log format, but may include some of the following:

- **Track all fields day-by-day**. Turning this option on cross-references the date/time field to all other fields. This improves performance when the Calendar or Date Range controls are used. Day-by-day information will still be available in the database if this option is not checked, but full table scans will be required to query it, which may make the queries much slower.

- **Track hosts individually**. Turning this option on structures the "host," "browsing host," or "source IP" field so that all IPs are tracked fully. If you leave this off, Sawmill will track only the top level domains (e.g. yahoo.com) and subnets (e.g. 127.128) of the IP's, and you will not be able to get information about the activities of a particular IP. If you turn track hosts individually on, every IP address in the log data will be tracked separately, making information available about every individual IP addresses and hostnames. Turning this on can significantly increase the size and memory usage of the database.

In addition, there will be a checkbox for each available numerical field in the log data. Checking one of these boxes will add another field to the database, providing information about that numerical field, and will add that numerical field to every report. This will slightly increase the size of the database for most fields, but tracking a "unique" field like visitors may be much more expensive. Turning on unique host (visitor) tracking will result in the visitor id information being tracked for all database items, which will significantly slow log processing and increase database size, but it is necessary if you need visitor information. For web and web proxy logs, you can greatly increase processing speed (as much as four times) by checking only the "page views" box (and not track hits or bandwidth).

Using Log Filters

Sawmill offers a wide range of log filters that lets you selectively eliminate portions of your log data from the statistics, or allows you to convert values in log fields. Log filters are written in the configuration language (see **The Configuration Language**). Log filters should not be confused with the "Filters" that appear in reports; log filters affect how the log data is processed, and report "Filters" affect which parts of the database data are displayed. There are many reasons you might want to filter the log data, including:

- You may not be interested in seeing the hits on files of a particular type (e.g. image files in web logs).
- You may not be interested in seeing the events from a particular host or domain (e.g. web log hits from your own domain or email from your own domain for mail logs).
- In web logs you may not be interested in seeing hits which did not result in separate page views like 404 errors (file not found) or redirects.

Sawmill's default filters automatically perform the most common filtering, like categorizing image files as hits but not page views, stripping off page parameters and more, but you will probably end up adding or removing filters as you fine-tune your statistics.

How Filters Work

Filters are arranged in a sequence, like a computer program, starting with the first filter and continuing up through the last filter. Each time Sawmill processes a log entry, it runs the filters in order, starting with the first one. Sawmill applies that filter to the log entry. The filter may accept the log entry by returning "done", in which case it is immediately selected for inclusion in the statistics. If a filter accepts an entry, the other filters are not run; once a filter accepts, the acceptance is final. Alternately, the filter may reject the entry by returning "reject", in which case it is immediately discarded, without consulting any filters farther down the line. Finally, the filter may neither accept nor reject, but instead pass the entry on to another filter (by returning nothing); in this case, and *only* in this case, another filter is run.

In other words, every filter has complete power to pass or reject entries, provided the entries make their way to that filter. The first filter that accepts or rejects the entry ends the process, and the filtering is done for that entry. A filter gets to see an entry *only* when every filter before it in the sequence has neither accepted nor rejected that entry. So the first filter in the sequence is the most powerful, in the sense that it can accept or reject without consulting the others; the second filter is used if the first has no opinion on whether the entry should be accepted or rejected, etc.

Web Logs: Hits vs. Page Views

For web logs, web server and HTTP proxy servers, Sawmill distinguishes between "hits" and "page views" for most types of logs. A "hit" is one access to the web server; i.e. one request for a file which may not actually result in the transfer of a file, as in the case of a redirect or an error. A "page view" is an access to a page rather than to an image or a support file like a style sheet. For some web sites and some types of analysis, image files, .class files, .css file, and other files are not as important as HTML pages--the important number is how many pages were accessed, not how many images were downloaded. For other sites and other types of analysis, all accesses are important. Sawmill tracks both types of accesses. When a filter accepts an entry, it decides whether it is a hit or a page view by setting the "hits" and "page_views" fields to 1 or 0. Hits are tallied separately, and the final statistics can show separate columns for hits and page views in tables, as well as separate pie charts and graphs. Both hits and page views contribute to bandwidth and visitor counts, but the page view count is not affected by hits on image files and other support files.

The Log Filter Editor

The easiest way to create log filters is in the Log Filter Editor, in the Log Filters section of the Config. To get to the Log Filters editor, click Show Config in the Profiles list (or click Config in the reports), then click Log Data down the left, then click Log Filters. The Log Filters Editor lets you create new filters using a user-friendly graphical interface, without having to write advanced filter expressions. However, some filtering operations cannot be performed without advanced filter expressions, so the Log Filter Editor also provides an option to enter an expression.

Advanced Expression Examples

In this section are some examples of how filter expressions can be used, and how they are put together.

Filtering out GIFs

The following filter rejects GIF files in web logs:

```
  if (file_type eq 'GIF') then "reject"
```

Filtering out domains or hosts

The following filter ignores hits from your own web log domain:

```
  if (ends_with(hostname, ".mydomain.com")) then "reject"
```

You can use a similar filter to filter out hits from a particular hostname:

```
  if (hostname eq "badhost.somedomain.com") then "reject"
```

This type of filter can be used on any field, to accept and reject based on any criteria you wish.

Field names that appear in filters (like file_type or hostname above) should be exactly the field names as they appear in the profile (not the field label, which is used for display purposes only and might be something like "file type"). Field names never contain spaces, and are always lowercase with underbars between words.

Filtering out pages or directories

The host filter above can be modified slightly to filter out entries based on any field. One common example is if you want to filter out hits on particular pages, for instance to discard hits from worm attacks. A filter like this:

```
  if (starts_with(page, "/default.ida?")) then "reject"
```

rejects all hits on /index.ida, which eliminates many of the hits from the Code Red worm.

A filter like this:

```
  if (!starts_with(page, "/directory1/")) then "reject"
```

then continue on to the next filter

rejects all hits *except* those on /directory1/, which can be useful if you want to create a database which focuses on only one directory (sometimes useful for ISPs).

Filtering out events before a particular date range

The following filter rejects entries before 2004:

```
  if (date_time_to_epoc(date_time) < date_time_to_epoc('01/Jan/2004 00:00:00')) then "reject"
```

Filtering out events older than a particular age

The following filter rejects entries older than 30 days (60*60*24*30 is the number of seconds in 30 days):

```
  if (date_time_to_epoc(date_time) < (now() - 60*60*24*30)) then "reject"
```

Filtering out events outside a particular date range

The following filter rejects all entries except those in 2003:

```
  if ((date_time < '01/Jan/2003 00:00:00') or (date_time >= '01/Jan/2004 00:00:00')) then
"reject"
```

Advanced Example: Converting the page field to strip off parameters

The parameters on the page field (the part after the ?) are often of little value, and increase the size of the database substantially. Because of that, Sawmill includes a default filter that strips off everything after the ? in a page field (hint: if you need the parameters, delete the filter). Sawmill uses a special "replace everything after" filter for this use, but for the purpose of this example, here's another filter that does the same thing (but slower, because pattern matching is a fairly slow operation):

```
  if (contains(page, "?")) then
    if (matches_regular_expression(page, "^(.*?).*$")) then
      page = $1 . "(parameters)"
```

This checks if the page contains a question mark; if it does, it matches the page to a regular expression with a parenthesized subexpression which is set to just the part before and including the question mark. The variable __HexEsc__241 is set automatically to the first parenthesized section, and this variable is used to set the page field to the part before and including the question mark, with "(parameters)" appended. For example, if the original value was /index.html?param1+param2, the result will be /index.html?(parameters). That is exactly what we wanted--the parameters have been stripped off, so all hits on index.html with parameters will have the same value, regardless of the parameters--and that will reduce the size of the database.

The filters look the same in profile files, so you can also edit a filter in the profile file using a text editor. You will need to use a backslash (\\) to escape quotes, dollar signs, backslashes, and other special characters if you edit the profile file directly.

Advanced Example: Rewriting usernames to full names using a CFG file

If your log data contains usernames like bob and jill, and you want to list their full names in reports instead, you can use it by creating a CFG file which maps usernames to full names, and referring to it from the log filter. For instance, you could call the file usernames_to_full_names.cfg, and put it in LogAnalysisInfo, and it could contain this:

```
usernames_to_full_names = {
  bob = "Bob Smith"
  jill = "Jill Teti"
  george = "George Jones"
} # usernames_to_full_names
```

The first line *must* match the filename (without the .cfg). The final line ends with a # followed by a comment; this is optional, but it is a good idea to use it so you always know what opening curly bracket you are closing with a closing curly.

This file can be as long as you wish; it can have one entry, or millions.

Now the log filter can be this:

```
if (subnode_exists("usernames_to_full_names", username)) then
  username = node_value(subnode_by_name("usernames_to_full_names", username));
```

This filter uses **`subnode_exists()`** to check if there is an entry in usernames_to_full_names for the current username (the value of the "username" log field; this assumes that the field is called "username"). If there is, it uses **`subnode_by_name()`** to get the node corresponding to that user, in usernames_to_full_names, and then uses **`node_value()`** to get the value of the node (the full name). It assigns that value to the username field, overwriting it with the full name.

Advanced Example: Adding external metadata to a profile

An extension of the above technique can be used to integrate external metadata with the log data in reports. For example, if you have a database which maps usernames to full names, departments, and states, and if you can convert that database to CFG format, then you can create a profile in Sawmill which reads log data containing only usernames, but reports (and aggregated data by) full names, departments, and states. The CFG file might be called user_records.cfg (in LogAnalysisInfo), and could look like this:

```
user_records = {
  bob = {
    full_name = "Bob Smith"
    department = "Mathematics"
    state = "Texas"
  } # bob
  jill = {
    full_name = "Jill Teti"
    department = "Art"
    state = "Ohio"
  } # jill
  george =  = {
    full_name = "George Jones"
    department = "Literature"
    state = "Ohio"
  } # george
} # user_records
```

And the log filter could look like this:

```
if (subnode_exists("user_records", username)) then (
  node user_record = subnode_by_name("user_records", username);
  full_name = node_value(subnode_by_name(user_record, "full_name"));
  department = node_value(subnode_by_name(user_record, "department"));
  state = node_value(subnode_by_name(user_record, "state"));
```

```
);
```

This filter uses **subnode_exists()** to check for the existence of a user record for the username; if it exists, it uses **subnode_by_name()** to get a pointer to that record (node). Then it uses **subnode_by_name()** and **node_value()** on the user record node, to extract the full_name, department, and state values. It puts them in custom fields (full_name, department, and state), so these fields must also be created (see **Creating Custom Fields**).

This technique is extremely powerful; it can be used to effectively "join" log data to an existing database table, by exporting that table to a CFG file, and then look up values from that table using a common key found both in the table and the log data.

This method is quite efficient; it does not iterate through the entire list, but looks it up using a "binary search" approach which is usually fast enough that the performance of the lookup is not a significant factor in database build time.

Advanced Example: Tracking new sessions in a web log

This is an example which uses an in-memory **configuration node** to store a list of all IP addresses seen so far, so when a new one is encountered, the filter can set a custom new_sessions field to 1. This causes the new_sessions field to be 1 for all new-IP events, and 0 otherwise, so date/time reports will show how many new visitors are coming each year/month/day/hour/etc.

```
v.c_ip = replace_all(c_ip, ".", "_");
if (!node_exists("previously_seen_ips")) then
  previously_seen_ips = "";
if (!subnode_exists("previously_seen_ips", v.c_ip)) then (
  new_sessions = 1;
  set_subnode_value("previously_seen_ips", v.c_ip, true);
);
```

The first line uses **replace_all()**, to compute a variable v.c_ip from the c_ip field, by replacing dots (.) with underbars (_). This is necessary because configuration nodes use dots as the separator, so dots cannot be used in configuration node names; we're about to add nodes with names like v.previously_seen_ips.{c_ip}, so we need to make sure c_ip does not contains dots.

The next two line uses **node_exists()** to check to see if the node v.previous_seen_ips exists; if it doesn't, it creates it (assigning a value to an undefined node defines it). Without this step, the next line, which checks for subnodes of v.previously_seen_ips, would fail with an error that v.previously_seen_ips does not exist.

The next part uses **subnode_exists()** to check if there is a subnode of v.previously_seen_ips which matches the c_ip field (with dots converted to underbars). For instance, if the log data contains 12.34.56.78 as the IP, this will check for the existence of a node v.previously_seen_ips.12_34_56_78. If this node exists, then we know that this IP has appeared previously, and this is not a new session. If this node does not exist, then we create it using **set_subnode_value()**, and then set new_sessions to 1, so this event will appear as a new session event in the reports.

The code above uses an in-memory node for previously_seen_ips, so it won't remember which IPs it has seen for the next database update; that's fine as long as you rebuild, but if you want to update you'll want to save the node to disk, and reload it for each update. This can be done using the log.filter_finalization node in the profile (or plug-in), which specifies code to run at the end of log processing. The following example saves previously_seen_ips to disk:

```
log.filter_finalization = `save_node('previously_seen_ips')`
```

The node will be saved to the file previously_seen_ips.cfg, in the LogAnalysisInfo folder . Since nodes are automatically loaded from disk when they are accessed, future updates will automatically load this node into memory when the log filter first accesses it, so there is no need for a "load_node()" operation.

Advanced Example: Rejecting spiders based on JS and /robots.txt access

Sawmill detects spiders automatically based on the spiders.cfg file in LogAnalysisInfo, but not all spiders are in that file, and some spiders deliberately hide themselves (they do not declare themselves as spiders in their user-agent field). This section describes a more sophisticated way of identifying spiders, based on whether they hits JavaScript pages, or /robots.txt.

The /robots.txt file is a file which describes what a spider may do on the site; all well-behaved spiders must access this file before accessing anything else. Therefore, a straightforward way of rejecting spiders is to collect a list of all accesses to /robots.txt, and to reject all IPs which accessed /robots.txt. But that assumes the spider is well-behaved, which may not be the case. So another level of detection is useful, if your site uses JavaScript or CSS: look for all visitors who *never* accessed a JavaScript or CSS file (.js or .css file). Spiders typically do not access JavaScript or CSS files, so if you see an IP which accessed the site, but never accessed any .js file or .css file, they are likely a spider (if your site uses JavaScript or CSS!). The following algorithm rejects all hits from all IPs which either *never* hit a JS or CSS file, or *do* hit the /robots.txt file.

First, we added filter initializion and finalization to the profile, to log.filter_initialization and log.filter_finalization:

```
    # Initialize the filters
```

```
    filter_initialization = `
spider_rejection.accessed_js_css_file = '';
spider_rejection.accessed_robots_txt = '';
`
    # Finalize the filters
    filter_finalization = `
save_node('spider_rejection.accessed_js_css_file');
save_node('spider_rejection.accessed_robots_txt');
`
```

The initialization code makes sure the spider_rejection.accessed_js_css_file and spider_rejection.accessed_robots_txt nodes exist. Note: you will need to create a directory called spider_rejection, in the LogAnalysisInfo directory, to hold these nodes. The finalization step saves these nodes, to LogAnalysisInfo/spider_rejection/accessed_js_css_file.cfg and LogAnalysisInfo/spider_rejection/accessed_robots_txt.cfg.

Now add a filter (to log.filters in the profile .cfg file) to detect CSS/JS hits, and to reject spiders.

```
reject_spiders = `

# Convert . to _ in the IP
v.converted_ip = replace_all(c_ip, '.', '_');

# If this is a JS file, remember that this IP accessed it.
if ((file_type eq 'JS') or (file_type eq 'CSS')) then (
  set_node_value(subnode_by_name('spider_rejection.accessed_js_css_file', v.converted_ip),
true);
);

# If this is /robots.txt, remember that this IP accessed it.
if (cs_uri_stem eq '/robots.txt') then (
  set_node_value(subnode_by_name('spider_rejection.accessed_robots_txt', v.converted_ip), true);
);

# Reject as spiders any hit which did not access JS/CSS files, or did access /robots.txt
if (subnode_exists('spider_rejection.accessed_robots_txt', v.converted_ip) or
    !subnode_exists('spider_rejection.accessed_js_css_file', v.converted_ip)) then (
  'reject';
);

`
```

This filter does the following:

1. It creates and sets a variable v.converted_ip, to the IP address (which is called c_ip in this IIS profile; it may be called something else for other log formats), with dots converted to underbars. This is necessary because node names cannot contain dots.
2. It checks if the hit is a JS or CSS hit; if it is, it sets spider_rejection.accessed_js_css_file.*ip* to true, where *ip* is the converted IP address.
3. It checks if the hit is on /robots.txt (again using IIS field names; this would be called "page" for Apache); if it is, it sets spider_rejection.accessed_robots_txt.*ip* to true, where *ip* is the converted IP address.
4. It checks if the current IP address is in the accessed_robots_txt list, or is *not* in the accessed_js_css_file list; if so, it rejects the hit.

The first time a dataset is processed with these log filters, this will cause the accessed_robots_txt and accessed_js_css_file lists to be populated. The data then must be processed a second time, since it is not possible to know the first time an IP address is encountered, whether that IP will ever hit a JS file. The second time it is processed, all data has been processed once, and the lists are complete, so the spider rejection will work properly. So the database needs to be built twice to remove all spiders.

Pathnames

A pathname is a value which fully describes the location of a file or folder on your computer. Pathnames are used in Sawmill to describe the locations of the log data, the server folder, the **Database folder**, and other files and folders.

The leftmost part of a pathname generally describes which hard drive or partition the file or folder is on, and as you move from left to right along the pathname, each successive part narrows the location further by providing the name of an additional subfolder.

It is not generally necessary to type pathnames if you are using the Sawmill graphical web browser interface; the Browse button next to each pathname field provides an easier way to specify a pathname, using a familiar folder browsing mechanism. This button is available everywhere except when choosing the server folder in CGI mode, where you must manually enter the pathname.

Pathnames use different formats on different platforms. On Windows, the format is

```
driveletter:\folder1\folder2 ... foldern\filename
```

for files, and the same for folders, except that the final `filename` is omitted (but not the final `\`). For instance, a file `my.conf` inside the folder `configs`, which is inside the folder `sawmill`, which is inside the folder `web` on the C: drive, is represented by `C:\web\sawmill\configs\my.conf` . The folder containing `my.conf` is represented by `C:\web\sawmill\configs\` .

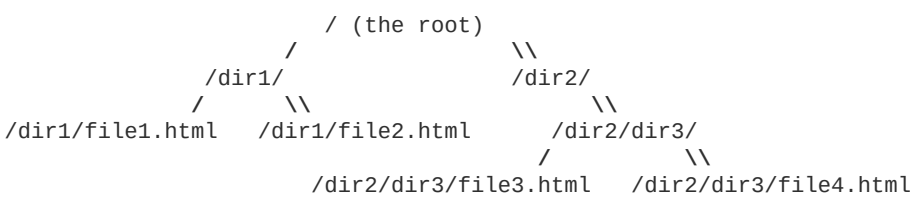
On MacOS X, Linux, or other UNIX-type systems, the format is

```
/folder1/folder2 ... foldern/filename
```

for files, and the same for folders, except that the final `filename` is omitted (but not the final `/`). For instance, a file `my.conf` inside the folder `configs`, which is inside the folder `sawmill`, which is inside the folder `web` (a subfolder of the root `/` folder), is represented by `/web/sawmill/configs/my.conf` . The folder containing `my.conf` is represented by `/web/sawmill/configs/` .

Hierarchies and Fields

Sawmill can break down log information on many fields. For instance, if it is processing a web server access log, it can break it down by page, date, host, and many others. Each field type is organized hierarchically, in a tree-like structure of items and subitems. For example, a section of a typical "page" hierarchy might look like this:

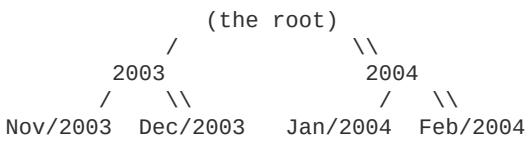


This is hierarchical; some pages are *above* others because they are *within* them. For instance, a folder is above all of its subfolders and all of the files in it.

This hierarchical structure allows you to "zoom" into your log data one level at a time. Every report you see in a web log analysis corresponds to one of the items above. Initially, Sawmill shows the page corresponding to /, which is the top-level folder (the "root" of the hierarchical "tree"). This page will show you all subpages of /; you will see /dir1/ and /dir2/. Clicking on either of those items ("subitems" of the current page) will show you a new page corresponding to the subitem you clicked. For instance, clicking on /dir2/ will show you a new page corresponding to /dir2/; it will contain /dir2/dir3/, the single subitem of /dir2/. Clicking on /dir2/dir3/ will show a page corresponding to /dir2/dir3/, containing the subitems of /dir2/dir3/: /dir2/dir3/file3.html and /dir2/dir3/file4.html.

Sawmill shows the number of page views (and/or bytes transferred, and/or visitors) for each item it displays. For instance, next to /dir2/ on a statistics page, you might see 1500, indicating that there were 1500 page views on /dir2/ or its subitems. That is, the sum of the number of page accesses on /dir2/, /dir2/dir3/, /dir2/dir3/file4.html, /dir2/dir3/file3.html is 1500. That could be caused by 1500 page views on /dir2/dir3/file4.html and no page views anywhere else, or by 1000 page views on /dir2/dir3/file3.html and 500 page views directly on /dir2/, or some other combination. To see exactly which pages were hits to create those 1500 page views, you can zoom in by clicking /dir2/.

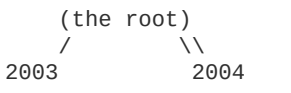
There are many other hierarchies besides the page hierarchy described above. For instance, there is the date/time hierarchy, which might look like this:



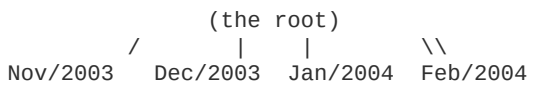
The date/time hierarchy continues downward similarly, with days as subitems of months, hours of days, minutes of hours, and seconds of minutes. Other hierarchies include the URL hierarchy (similar to the page hierarchy, with http:// below the root, http://www.flowerfire.com/ below http://, etc.) the hostname hierarchy (.com below the root, flowerfire.com below .com, www.flowerfire.com below flowerfire.com, etc.) and the location hierarchy (countries/regions/cities).

Some terminology: The very top of a hierarchy is called the "root"; e.g., "(the root)" in the date/time hierarchy above, or "/" in the page hierarchy. An item below another item in the hierarchy is called a "subitem" of that item; e.g., 2004 is a subitem of the root, and /dir2/dir3/file4.html is a subitem of /dir2/dir3/. Any item at the very bottom of a hierarchy, an item with no subitems, is called a "leaf" or a "bottom level item"; e.g., Jan/2003 and /dir1/file1.html are leaves in the above hierarchies. An item which has subitems, but is not the root, is called an "interior node"; e.g., 2004 and /dir2/dir3/ are interior nodes in the above hierarchies.

To save database size, processing time, and browsing time, it is often useful to "prune" the hierarchies. This is done using the Suppress Levels Below and Suppress Levels Above parameters of the database field options. Levels are numbered from 0 (the root) downward; for instance 2003 is at level 1 above, and /dir2/dir3/file4.html is at level 4. Sawmill omits all items from the database hierarchy whose level number is greater than the Suppress value. For instance, with a Suppress value of 1 for the date/time hierarchy above, Sawmill would omit all items at levels 2 and below, resulting in this simplified hierarchy in the database:



Using this hierarchy instead of the original saves space and time, but makes it impossible to get date/time information at the month level; you won't be able to click on 2003 to get the month information. Sawmill also omits all items from the hierarchy whose level number is less than or equal to the Collapse value (except the root, which is always present). For instance, with a Collapse value of 1, and Suppress of 2, Sawmill would omit all level 1 items, resulting in this hierarchy:



All four of the level 2 items are now direct subitems of the root, so the statistics page for this hierarchy will show all four months. This is useful not just because it saves time and space, but also because it combines information on a single page that otherwise would have taken several clicks to access.

Here's an example of "Suppress Levels Above" and "Suppress Levels Below," based on the page field value /dir1/dir2/dir3/page.html. With above=0 and below=999999, this will be marked as a single hit on the following items:

level 0: /
level 1: /dir1/
level 2: /dir1/dir2/
level 3: /dir1/dir2/dir3/
level 4: /dir1/dir2/dir3/page.html

With above=3, below=999999, all levels above 2 are omitted, the root level, 0, is always included:

level 0: /
level 2: /dir1/dir2/
level 4: /dir1/dir2/dir3/page.html

With above=0, below=2, all levels below 2 are omitted:

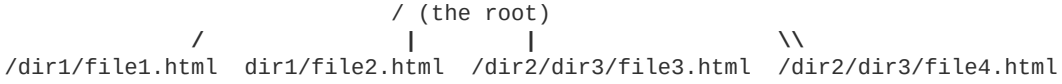
level 0: /
level 1: /dir1/
level 2: /dir1/dir2/

above=2, below=3: all levels above 2 are omitted (except 0), and all levels below 3 are omitted:

level 0: /
level 2: /dir1/dir2/
level 3: /dir1/dir2/dir3/

In the last example, zooming in on / will show you /dir1/dir2/ (you will never see /dir1/ in the statistics, because level 1 has been omitted); zooming in on that will show you /dir1/dir2/dir3/, and you will not be able to zoom any further, because level 4 has been omitted.

On a side note, the "show only bottom level items" option in the Table Options menu provides a dynamic way of doing roughly the same thing as using a high value of Collapse. Using the Options menu to show only bottom level items will dynamically restructure the hierarchy to omit *all* interior nodes. In the case of the page hierarchy above, that would result in the following hierarchy:



This hierarchy has all leaf items as direct subitems of the root, so the statistics page for this hierarchy will show all four pages. This is much faster than using Suppress because after Suppress has been modified, the entire database must be rebuilt to reflect the change.

The Database Structure Options provide a couple of other ways of pruning your hierarchies. The "Always include bottom-level items" option, forces Sawmill to include the bottom-level items in the hierarchy, regardless of the setting of the Suppress value. It is useful to include the bottom-level items if you need them for some feature of Sawmill. For instance, visitor information requires that all the bottom-level hosts be present, and session information is most meaningful if all the bottom-level date/times are present, but you want to prune some of the interior of the hierarchy to save space.

Hierarchies can be either left-to-right, with the left part of the item "enclosing" the right part, as in /dir1/dir2/file.html, where /dir1/ encloses /dir1/dir2/, which encloses /dir1/dir2/file.html, or right-to-left with the right part of the item enclosing the left part, as in www.flowerfire.com, where .com encloses flowerfire.com, which encloses www.flowerfire.com. Hierarchies use one or more special characters to divide levels (e.g. / in the page hierarchy or . in the host hierarchy). These options and more can

be set in the log field options.

Some log fields are not hierarchical, for instance the operation field of a web log (which can be GET, POST, or others), or integer fields like the size or response fields. These fields are specified in the log field options as non-hierarchical, or "flat", and all items in those hierarchies appear directly below the root.

Cross-Referencing and Simultaneous Filters

Sawmill lets you "zoom in" using complex filters, for instance to track the events on any particular day by page; e.g., in a web log to see which pages were hit on that day, or to break down the events on any page by day, to see which days the page was accessed. Sawmill can be configured to allow this sort of cross-referencing between any or all fields in the database. The ability to zoom is always available, but without cross-reference tables it must scan the entire main table to compute results, which can be slow for large datasets. Cross-reference tables provide records of common queries, so they can be computed quickly without reference to the main log data table.

Cross-references are not an *enabling* or activating feature, as they were in earlier versions of Sawmill -- all reports are available, even if no cross-reference tables are defined. Cross-reference tables are an *optimizing* feature, which increases the speed of certain queries.

Another way of looking at this feature is in terms of filters; when two fields are cross-referenced against each other, Sawmill is able to apply filters quickly to both fields at the same time, without needing to access the main table.

If two fields are *not* cross-referenced against each other, Sawmill can apply filters to one field or the other quickly, but filtering both simultaneously will require a full table scan. If the page field is not cross-referenced against the date/time field, for instance, Sawmill can quickly show the number of hits on a /myfile.html, or the number of hits on Jun/2004, but not the number of hits on /myfile.html which occurred during Jun/2004 (which will require a full table scan). This means not only that Sawmill can't quickly show a page with filters applied to both fields in the Filters section, but also that Sawmill cannot quickly show "pages" report when there is a filter on the date/time field, or a "years/months/days" or "days" report when there is a filter on the page field, since the individual items in these views effectively use simultaneous filters to compute the number of hits.

On the other hand, cross-reference tables use space in the database. The more fields you cross-reference, the larger and slower your database gets. Restricting cross-referencing only to those fields that you really need cross-referenced is a good way to limit the size of your database, and speed browsing of the statistics.

When you create your profile, you will be given several options in the amount of detail you want to track, and some of these options affect the default cross-references. Turning on day-by-day (calendar) tracking cross-references the date/time field to every other field. Other than that, cross-references are set by default for each field, but no two fields are included in the same cross-reference group. This is a fairly minimal use of cross-references, but for faster database builds, you can delete those as well. However when you query this may be slower since the database main table needs to be queried because there is no cross-reference group available.

Generally, you should start out with few cross-references; a default analysis is a good starting point. If you need a type of information not available (for instance, if you want to know the browser versions that are accessing a particular page), and the report generates too slowly, try adding the necessary cross-references. See **Memory, Disk, and Time Usage** for more information on optimizing your memory, disk space, and processing time.

Again, cross-references are never *necessary* to generate a particular report -- they just make processing reports faster.

Regular Expressions

Regular Expressions

A regular expression, often called a *pattern*, is simply a text string that describes or matches a set of strings, according to certain syntax rules. Strings are sequences of characters; for instance, a filename is a string, and so is a log entry. The pattern is made up of the string to match, plus special characters which match classes of string, plus operators to combine them. Sawmill uses regular expressions in many places, including:

- You can specify the log files to process using a regular expression.
- You can specify the log file format using **Log data format regular expression**.
- You can filter log entries based on a regular expression using log filters.

You can also use wildcard expressions in these cases, using * to match any string of characters, or ? to match any single character (for instance, *.gif to match anything ending with .gif, or Jan/??/2000 to match any date in January, 2000). Wildcard expressions are easier to use than regular expressions, but are not nearly as powerful.

Regular expressions can be extremely complex, and it is beyond the scope of this manual to describe them in full detail. See **Yahoo's Regular Expression Category** for more in-depth information; this link leads to Yahoo's web pages, and so requires a live internet connection.

Here are the simplest rules:

- A letter or digit matches itself (most other characters do as well).
- The . character (a period) matches any character.
- The * character matches zero or more repetitions of the expression before it.
- The + character matches one or more repetitions of the expression before it.
- The ^ character matches the beginning of the string.
- The $ character matches the ending of the string.
- A square-bracketed series of characters matches any of those characters. Adding a ^ after the opening bracket matches any character *except* those in the brackets.
- Two regular expressions in a row match any combination where the first half matches the first expression, and the second half matches the second expression.
- The \ character followed by any other character matches that character. For example, * matches the * character.
- A regular expression matches if it matches any part of the string; i.e. unless you explicitly include ^ and/or $, the regular expression will match if it matches something in the middle of the string. For example, "access\.log" matches not only `access.log` but also `old_access.log` and `access.logs`.
- A parenthesized regular expression matches the same thing as it does without parentheses, but is considered a single expression (for instance by a trailing *). Parentheses can be used to group consecutive expressions into a single expression. Each field should be parenthesized when using **Log data format regular expression**; that's how Sawmill knows where each field is.
- An expression of the form (A|B) matches either expression A or expression B. There can also be more than two expressions in the list.

The list goes on, but is too large to include here in complete form. See the Yahoo link above. Some examples:

- **a** matches any value containing the letter a.

- **ac** matches any value containing the letter a followed by the letter c.
- **word** matches any value containing the sequence "word".
- **worda*** matches any value containing the sequence "word" followed by zero or more a's.
- **(word)*a** matches any value containing zero or more consecutive repetitions of "word", where the last repetition followed by an a.
- **\.log$** matches any value ending with .log (good for matching all files in a directory ending with .log).
- **^ex.*\.log$** matches any value starting with ex and ending with .log.
- **^access_log.*1** matches any value starting with "access_log", and containing a 1 somewhere after the leading "access_log" (note that the 1 does *not* have to be at the end of the string for this to match; if you want to require that the 1 be at the end, add a $ to the end of the expression).
- **^access_log_jan....2004$** matches any value starting with "access_log_jan", followed by four characters (any four characters), followed by "2004", followed immediately by the end of the value.

As you can see, regular expressions are extremely powerful; a pattern can be devised to match almost any conceivable need.

NOTE ABOUT FILTERS

Both regular expression pattern filters and DOS-style pattern filters are necessary in some cases, but they should be avoided when possible because pattern filters can be considerably slower than the simpler filter types like "ends with" or "contains". If you can create a filter without patterns, do--your log processing will be faster.

Security

Since Sawmill runs as a CGI program or as a web browser, it publishes its interface to any web browser which can reach its server. This is a powerful feature, but also introduces security issues. Sawmill has a number of features which address these issues:

1. Non-administrative users can access Sawmill through the profilelist (same as administrative users). When a non-administrator is logged in, the profile list only allows users to view reports of profiles; users cannot create, edit, or delete profiles, and they cannot build, update, or modify the database of any profile. The profile list is available at:

   ```
   http://www.myhost.com:8987/
   ```

   in web server mode, or

   ```
   http://www.myhost.com/cgi-bin/sawmill
   ```

   in CGI mode.

2. If you wish to take it a step further, and not even present the profiles list to the users, you can refer users to the reports for a particular profile:

   ```
   http://www.myhost.com/cgi-bin/sawmill.cgi?dp+templates.profile.index+
   p+profilename+webvars.username+john+webvars.password+johnspassword
   ```

   (this should all be on one line). Accessing this URL will show the reports for the profile "profilename", after logging in as user john with password "johnspassword".

3. Only authorized administrators, meaning users who know the username and password of a Sawmill administrator, chosen at install time, may create new profiles, and only authorized administrators may modify profiles. Without administrator access, a user cannot create a new profile, modify an existing profile in any way, or perform any of the other tasks available on the administrative interface.

Sawmill also provides detailed control over the file and folder permissions of the files and folders it creates; see **File/Folder Permissions**.

File/Folder Permissions

Sawmill lets you control very precisely the permissions on the files and folders it creates. This is particularly useful on multi-user environments, where file permissions are a fundamental part of file system security.

Permissions can be set independently for a number of categories of files and folders Sawmill creates.

For each category, you can specify the value of the permissions number. The permissions number is a three-digit octal (base-8) number, as accepted by the UNIX chmod command. The first digit controls the permissions for the user running Sawmill, the second digit controls the permissions for the group of the user running Sawmill, and the third digit controls the permissions for all other users. Each digit is the sum of: 4, if read permission should be granted; 2, for write permissions; and 1, for execute or folder search permission. These values can be added in any combination to provide any mixture of permissions. For example, to grant only read permission, use a 4. To grant both read and write permission, use the sum of 4 and 2, or 6. To grant read and execute permission, use the sum of 4 and 1, or 5. You should give execute permissions for folders if you want users to be able to view their contents.

A complete example of a permissions option value is 754, this gives the Sawmill user read, write, and execute permission, gives the group read and execute permission, and gives all other users read permission.

See **Security** for more information on Sawmill's security features.

Users

The first time you run Sawmill through a web browser, it will prompt you to choose an administrative username and password. This information specifies the first user and is stored in the users.cfg file in the LogAnalysisInfo folder. After that you can create additional users by editing the users.cfg file.

The administrative users(s) must be logged in to perform any tasks where security is an issue, for instance, when specifying which log data to process, or when creating a profile. If your administrator changes or you forget the name and password, you can reset it by removing the users.cfg file.

You can check the "save my password" box when you enter the password. If you do that, Sawmill will save your username and password on the machine running the web browser, and will not ask you for the password if you are using that machine. Be careful though -- anyone else using that machine will also be able to get into Sawmill without a password.

For more information on security in Sawmill, see **Security**.

Memory, Disk, and Time Usage

Manage your Memory, Disk and Time Resources

Sawmill processes a huge amount of data while building a database or displaying statistics. Because of this, it uses a lot of resources: disk space, memory, and processing time. If you are considering running Sawmill on a public or shared server, you may want to investigate their resource policy to see if they allow high-CPU programs to be run there.

However, you can customize Sawmill to use less of some of these resources, by using more of others. You can also customize Sawmill to use less of *all* resources by reducing the amount of data in your database. This section describes the options that let you manage your memory, disk, and time resources.

Speeding up the Building of the Database

A database is built (or updated) in three stages: The main table is created from the processing of the log data, then the cross-reference tables are built from the main table and then the main table indices are created.

One way to speed up all of these processes is to use multiple processors. Depending on the version of Sawmill you're using, it may have the ability to split database builds across multiple processes, building a separate database with each processor from part of the dataset, and then merging the results. This can provide a significant speed up -- 65% speed up using two processors is fairly typical.

Increasing the speed of your processor is also a very good way to increase the speed of database building -- database builds are primarily CPU-bound, so disk speed, memory speed, and other factors are not as important as the speed of the processor.

If you've configured Sawmill to look up your IP numbers (using **Look up IP numbers using domain nameserver (DNS)**), the database building process will be slower than usual, as Sawmill looks up all the IP numbers in your log file. You can speed things up by not using **Look up IP numbers using domain nameserver (DNS)**, by decreasing the **DNS timeout (seconds)**, and/or by improving Sawmill's bandwidth to the DNS server.

You can also speed up all three stages by simplifying the database structure, by eliminating database fields or by using log filters to simplify them. For instance, if you add a log filter which converts all IP addresses to just the first two octets, it will be a much simpler field than if you use full IPs.

Cross-reference tables can be eliminated entirely to improve database build performance; however by eliminating them, you will slow query performance for those queries which would have used a cross-reference table. See **Cross-Referencing and Simultaneous Filters** for more details.

Using less memory during database builds

For most large datasets, the major factor in memory usage during builds are the item lists. There is one list per field, and each list includes every value for that field. For large fields, these lists can be huge -- for instance, if there are 100 million unique IPs, and each IP is 10 bytes long (e.g. "123.124.125.126" is 15 bytes long), then the total memory required for that list will be 100M * 10, or 1G of memory. Sawmill uses memory-mapped files for these lists, so depending on the operating system's implementation of memory mapped files, these may appear to be normal memory usage, virtual memory usage, or something else. However, most 32-bit operating systems restrict each process to 2G of memory space, including mapped files, so it doesn't take too much to exceed that.

The most complete solution is to get more memory, and to use a 64-bit operating system, this lifts the 2G limit. Even large datasets seldom actually reach 1G for the largest item list, and it is usually only a handful of fields which are large, so 2G is usually enough. But if you're running out of memory with 2G of RAM, it may be time to go to a 64-bit OS.

If a 64-bit OS isn't an option, you will need to simplify your database fields using log filters. For example, a filter which chops off the last octet of the IP will greatly reduce the number of unique IPs, probably dropping a huge 1G item list under 100M. Also, you may want to simply eliminate the troublesome field, if there is no need for it -- for instance, the uri-query field in web logs is sometime not needed, but tends to be very large. To determine which field is the problem, build the database until it runs out of memory, and then look at the database directory (typically in LogAnalysisInfo/Databases) to see which files are large. Pay particular attention to the 'items' folder -- if files in the xyz folder are particularly huge, then the xyz field is a problem.

Finally, if you need to use less disk space or memory due to a quota on your web server, you may be able to get around this problem by running Sawmill on a local machine, where *you* dictate disk space constraints, and setting it to fetch the log data by FTP.

Creating Log Format Plug-ins (Custom Log Formats)

Sawmill supports most common log formats, but if you're using a log format which is not recognized by Sawmill, it is possible for you to custom create a log format for it. To create a custom log format is a fairly technical task in most cases, and it may prove challenging for a novice user. A technical user, especially one with programming and regular expression experience, should be able to create a plug-in by following the instructions below. It is particularly important to be at least familiar with regular expressions; several aspects of log format plug-ins involve creating regular expressions to match the log data.

The log formats supported by Sawmill are described in small text files found in the log_formats subfolder of the LogAnalysisInfo folder. Each file in the log_formats folder defines a separate log format. To add a new log format to the list of formats supported by Sawmill, you need to create your own format file, and put in the log_formats folder.

To simplify the process, use the existing format files as templates. Look over a few of the built-in formats before you start creating your own. Most of the files look similar, and you will probably want to copy one of them and modify it to create your own format. However, as you compare the existing plug-ins to the examples below, you will often find that the format is not quite the same; in particular, you may find that the existing plug-ins are much more verbose that the syntax described below. You will also notice the difference in the log.fields, database.fields, and parsing filters, which have many unnecessary and redundant parameters in some plug-ins. We have been improving the syntax over the past few years to make it easier to create plug-ins, however many of the old plug-ins have not been updated to the new syntax. The old syntax is still supported but you should use the simpler syntax, as shown below, when creating new plug-ins.

Intellectual Property Rights	We Create the Plug-in
If you create a plug-in then you own all the rights to that plug-in and can decide if you want to release it to us for inclusion in the standard Sawmill distribution. If you release it to us, then you agree to give us unlimited rights to reproduce and distribute at no cost. If we create the plug-in (for free or for a fee) then we own all the intellectual rights to the plug-in.	If you would like us to create your plug-in then submit your request to us at support@flowerfire.com. All public formats will be accepted for free implementation, at no charge. They will be put into our log format queue, with no guarantee of when or if they are implemented. There is a continuous demand for new log file formats so there may be a significant delay before the plug-in is complete. For a faster response you may pay us to create the plug-in, which can be returned quickly.

Steps to Create a Log Format Plug-in

The easiest way to create a plug-in is to start from a template, and edit each of the sections in order. Log format plug-ins contain the following major sections:

1. **The log format name and label**
2. **The autodetection regular expression**
3. **Parsing the log data**
4. **Add the log fields**
5. **Add the database fields**
6. **Add the numerical database fields**
7. **Add the log filters**
8. **Add the database field associations**
9. Add the report groups (**automatic** or **manual** approach)
10. **Debug the plug-in**

Additional topics:

1. **Creating plug-ins for syslog servers or sysloggging devices**

The Log Format Name and Label

The log format name and the label are closely related. The log format label is the "human readable" name of the format. For instance, it might be "Common Access Log Format." The log format name is the "internal" version of that name, which is used internally to refer to it; it is typically a simpler version of the label, and it contains only letters, numbers, and underbars. For instance, if the format label is "Common Access Log Format," the name might be common_access. The name appears on the first line of the file, and the label appears in the log.format.format_label line of the file:

```
common_access = {

  # The name of the log format
  log.format.format_label = "Common Access Log Format"
  ...
```

The filename of the plug-in is also the name, with a ".cfg" extension. So the first thing you need to do when you create a plug-in is decide the label and name, so you can also name the file appropriately. Then copy another plug-in from the log_formats directory, naming the new one after your new plug-in. For instance, you might copy the common_access.cfg file, and rename the copy to my_log_format.cfg .

The Autodetection Regular Expression

The autodetection regular expression is used by the Create Profile Wizard to determine if a particular log format plug-in matches the log data being analyzed. The expression might look something like this:

```
  log.format.autodetect_regular_expression =
    "^[0-9]+/[0-9]+/[0-9]+ [0-9]+:[0-9]+:[0-9]+,[0-9.]+,[^,]*,[^,]*,[^,]*"
```

The regular expression specified here will be matched against the first few ten lines of log data (or more, if you also specify the number to match in a log.format.autodetect_lines option). If any of those lines matches this expression, this log format will be shown as one of the matching formats in the Create Profile Wizard.

When creating a log format plug-in, you'll typically want to set up this expression first, before you do anything else (other than create the plug-in file and choose the name and label). Once you've created the plug-in file, and set the expression, you can do the first testing of the plug-in by starting to create a new profile in the Create Profile Wizard, and seeing if the plug-in is listed as a matching format. If it's not listed, then the autodetect expression isn't right. It is best to get the autodetection working before you continue to the next step.

Because the expression appears in double-quotes (") in the plug-in, you need to escape any literal double-quotes you use in the regular expression, by entering them as \" . Because \ is used as an escape character, you also need to escape any \ you use as \\. For instance, if you want to match a literal [, that would be \[in the regular expression, so you should escape the \ and enter it as \\[in the quoted string.

Parsing the Log Data: Parsing with the Parsing Regular Expression

There are three ways of extracting field values from the log data: a parsing regular expression, a delimited approach (a.k.a., "index/subindex"), and parsing filters. Only one of these three approaches is typically used in a plug-in. This section describes parsing using a parsing regular expression. For a description of parsing with delimiters, see "Parsing With Delimited Fields". For a description of parsing with parsing filters, see "Parsing With Parsing Filters: Lines With Variable Layout" and "Parsing With Parsing Filters: Events Spanning Multiple Lines".

The parsing regular expression is matched against each line of log data, as the data is processed. This is easily confused with the autodetection regular expression, but the purpose of the two are very different. The autodetection regular expression is used *only* during the Create Profile Wizard, to show a list of formats which match the log data. The parsing regular expression is not used at all during profile creation; it is used *only* when log data is processed (e.g., during a database build) to extract field values from each line of the log data.

The expression should include subexpressions in parentheses to specify the field values. The subexpressions will be extracted from each matching line of data, and put into the log fields. For instance, this expression:

```
  log.format.parsing_regular_expression =
    "^([0-9]+/[0-9]+/[0-9]+) ([0-9]+:[0-9]+:[0-9]+),([0-9.]+),([^,]*),([^,]*),([^,]*)"
```

extracts a date field (three integers separated by slashes), then after a space, a time field (three integers separated by colons), then a series of four comma-separated fields; the first of the comma-separated fields is an IP address (any series of digits and dots), and the rest of the fields can be anything, as long as they don't contain commas.

Notice how similar the layouts of the parsing regular expression is to the autodetection regular expression. It is often possible to derive the parsing regular expression just by copying the autodetection expression and adding parentheses.

As each line of log data is processed it is matched to this expression and if the expression matches, the fields are populated from the subexpressions, in order. So in this case, the log fields must be date, time, ip_address, fieldx, fieldy, and duration in that order.

If the parsing regular expression is the one listed above, and this line of log data is processed:

```
  2005/03/22 01:23:45,127.0.0.1,A,B,30
```

Then the log fields will be populated as follows:

```
date: 2005/03/22
time: 01:23:45
ip_address: 127.0.0.1
fieldx: A
fieldy: B
duration: 30
```

Parsing With Delimited Fields

If your log data is extremely simple in its layout, you can use a faster parsing method, sometimes called delimited parsing or "index/subindex" parsing. In order to use this method, the log data must have the following characteristics:

- Each event must appear on a separate line; events may not span multiple lines.
- Each line must contain a list of fields separated by a delimiter. For instance, it might be a list of fields separated from each other by commas, or by spaces, or by tabs.

If the log data matches these criteria, you can use delimited parsing. The example we've been using above:

```
2005/03/22 01:23:45,127.0.0.1,A,B,30
```

meets the first condition, but not the second, because the date and time fields are separated from each other by a space, but other fields are separated from each other by commas. But let's suppose the format was this:

```
2005/03/22,01:23:45,127.0.0.1,A,B,30
```

This is now a format which can be handed by delimited parsing. To use delimited parsing, you must omit the log.format. parsing_regular_expression option from the plug-in; if this option is present, regular expression parsing will be used instead. You must also specify the delimiter:

```
log.format.field_separator = ","
```

If the delimiter is not specified, the comma in this case, whitespace will delimit fields; any space or tab will be considered the end of one field and the beginning of the next.

When using delimited parsing, you also need to set the "index" value of each log field. The index value of the field tells the parse where that field appears in the line. Contrast this to regular expression parsing, where the position of the field in the log. fields like specifies the position in the line; delimited parsing pays no attention to where the field appears in the log.fields list, and populates fields based solely on their index (and subindex; see below). So for the comma-separated format described above, the log fields would look like this:

```
log.fields = {
  date = {
    index = 1
  }
  time = {
    index = 2
  }
  ip_address = {
    type = "host"
    index = 3
  }
  fieldx = {
    index = 4
  }
  fieldy = {
    index = 5
  }
  duration = {
    index = 6
  }
}
```

For brevity, this is usually represented with the following equivalent syntax; any group G which has only one parameter p, with value v, can be represented as "G.p = v":

```
log.fields = {
  date.index = 1
  time.index = 2
```

```
    ip_address = {
      type = "host"
      index = 3
    }
    fieldx.index = 4
    fieldy.index = 5
    duration.index = 6
  }
```

For instance, field4 will be populated from the fourth comma-separated field, since its index is 4.

When using whitespace as the delimiter, quotation marks can be used to allow whitespace in fields. For instance, this line:

```
  2005/03/22 01:23:45 127.0.0.1 A "B C" 30
```

would extract the value "B C" into the fieldy field, even though there is a space between B and C, because the quote set it off as a single field value. It is in this case that the "subindex" parameter can be used; subindex tells the parser to split the quoted field one level deeper, splitting B and C apart into subindex 1 and subindex 2. So for the line above, if the fieldy field also had a "subindex = 1" parameter, fieldy would be set to "B"; if instead it had "subindex = 2", fieldy would be set to "C". If it has no subindex, or if subindex is 0, it will be set to the whole quoted field, "B C". This is used in Common Log Format (a web log format) to split apart the query string:

```
 ... [timestamp] "GET /some_page.html HTTP/1.0" 200 123 ...
```

In this case, the operation (GET) is extracted as subindex 1 of the quoted field, and the page (/some_page.html) is extracted as subindex 2, and the protocol (HTTP/1.0) is extracted as subindex 3.

Parsing With Parsing Filters: Lines With Variable Layout

Delimited parsing and regular expression parsing are both limited to log formats where each line has the same layout. But some log formats have a variable layout; some lines have one format, and some have another. Consider this bit of mail log data:

```
  2005/03/22,01:23:45,ICMP,127.0.0.1,12.34.56.78
  2005/03/22,01:23:46,TCP,127.0.0.1,3416,23.45.67.78,80
```

This firewall log shows two events. The first is a ICMP packet sent from 127.0.0.1 to 12.34.56.77; the second is a TCP packet sent from port 3416 of 127.0.0.1 to port 80 of 23.45.67.89. Because ICMP does not use ports, the first line is missing the port numbers which appear in the second line. This means that the format of the second line is different from the format of the first line, and the index positions of the fields vary from line to line; on the first line, the destination_ip field is at index 5, and on the second line, that same field is at position 6. This makes it impossible to parse this log format using delimited parsing. Similarly, it is difficult to create a single regular expression which can handle both formats although it's possible by making the port fields and one of the commas optional in the regular expression, but for the sake of the example, let's assume there is no regular expression which can handle both formats. Certainly, there are *some* formats which have such different line layouts that a single regular expression cannot handle all of them.

The solution is to use parsing filters. Parsing filters are code written in **The Configuration Language** which extract data from the log line, and put it in the fields. To write parsing filters is similar to writing a script or a program, it does help if you have some programming experience, but simple parsing filters can be written by anyone.

Here is an example of a parsing filter which can parse this format:

```
  log.parsing_filters.parse = `
if (matches_regular_expression(current_log_line(),
      '^([0-9/]+),([0-9:]+),([A-Z]+),([0-9.]+),([0-9.]+)$')) then (
  date = $1;
  time = $2;
  protocol = $3;
  source_ip = $4;
  destination_ip = $5;
)
else if (matches_regular_expression(current_log_line(),
   '^([0-9/]+),([0-9:]+),([A-Z]+),([0-9.]+),([0-9]+),([0-9.]+),([0-9]+),$')) then (
  date = $1;
  time = $2;
  protocol = $3;
  source_ip = $4;
  source_port = $5;
  destination_ip = $6;
  destination_port = $7;
)
```

```

Note the backtick quotes (`) surrounding the entire expression; a log parsing filter is a single string value within backtick quotes.

This filter uses the matches_regular_expression() function to match the current line of log data (returned by the current_log_line() function) against the regular expression in single quotes. The first long line checks for the ICMP line layout with just two IPs addresses; the second long line checks for the TCP/UDP layout with ports. If the current line matches the ICMP layout, then this filter populates the date, time, protocol, source_ip, and destination_ip fields. It does this using the $N variables, which are automatically set by the matches_regular_expression() function when it matches; $1 is the first parenthesized expression (date), $2 is the second (time), etc. This parsing filter handles either log format, and populates the correct parts of each line into the appropriate fields.

The use of escaping backslashes in parsing filters is greatly multiplied than using backslashes in regular expressions. You may recall that a literal backslash in a regular expression had to be represented as \\ in the parsing regular expression, because it was inside double quotes. Parsing filters are within *nested* quotes. Typically, the entire parsing filter is in backtick (`) quotes, and the regular expression is within that in single or double quotes. Because of this, if you want to use a backslash to escape something in a regular expression, it has to be *quadrupled*, e.g., you need to use \\\\[ to get \[ (a literal left square bracket) in a regular expression. The final extreme is if you need a literal backslash in a regular expression; that is respresented as \\ in regular expression syntax, and *each* of those has to be quadrupled, so to match a literal backslash in a regular expression in a parsing filter, you need to use \\\\\\\\. For instance, to match this line:

```
  [12/Jan/2005 00:00:00] Group\User
```

You would need to use this parsing filter:

```
  if (matches_regular_expression(current_log_line(),
     '^\\\\[([0-9]+/[A-Z][a-z]+/[0-9]+) ([0-9:]+)\\\\] ([^\\\\]+)\\\\\\\\(.*)')) then
    ...
```

Parsing With Parsing Filters: Events Spanning Multiple Lines

There is one major limitation to all types of parsing discussed so far, all of these methods assume that there is one event per line. However many log formats split events across lines, so one field of the event may appear on the first line, and other fields may appear on later lines. To parse this type of log you need to use the collect/accept method with parsing filters. In this method fields are "collected" into virtual log entries and when all fields have been collected, the virtual log entry is "accepted" into the database.

Consider the following mail log:

```
  2005/03/22 01:23:45: connection from 12.34.56.78; sessionid=<156>
  2005/03/22 01:23:46: <156> sender: <bob@somewhere.com>
  2005/03/22 01:23:46: <156> recipient: <sue@elsewhere.com>
  2005/03/22 01:23:50: <156> mail delivered
```

This represents a single event; the mail client at 12.34.56.78 connected to the server, and sent mail from bob@somewhere.com to sue@elsewhere.com. Note the sessionid <156>, which appears on every line; log data where events span lines almost always has this type of "key" field, so you can tell which lines belong together. This is essential because otherwise the information from two simultaneous connections, with their lines interleaved, cannot be separated.

A parsing filter to handle this format could look like this:

```
  log.parsing_filters.parse = `
if (matches_regular_expression(current_log_line(),
      '^[0-9/]+ [0-9:]+: connection from ([0-9.]+); sessionid=<([0-9]+)>')) then
  set_collected_field($2, 'source_ip', $1);
else if (matches_regular_expression(current_log_line(),
           '^[0-9/]+ [0-9:]+: <([0-9]+)> sender: <([^>]*)>')) then
  set_collected_field($1, 'sender', $2);
else if (matches_regular_expression(current_log_line(),
           '^[0-9/]+ [0-9:]+: <([0-9]+)> recipient: <([^>]*)>')) then
  set_collected_field($1, 'recipient', $2);
else if (matches_regular_expression(current_log_line(),
           '^([0-9/]+) ([0-9:]+): <([0-9]+)> mail delivered')) then (
  set_collected_field($3, 'date', $1);
  set_collected_field($3, 'time', $2);
  accept_collected_entry($3, false);
)
`
```

This works similarly to the variable-layout example above, in that it checks the current line of log data against four regular
```

expressions to check which of the line types it is. But instead of simply assigning 12.34.56.78 to source_ip, it uses the function set_collected_field() with the session_id as the "key" parameter, to assign the source_ip field of the collected entry with key 156 to 12.34.56.78. Effectively, there is now a virtual log entry defined, which can be referenced by the key 156, and which has a source_ip field of 12.34.56.78. If another interleaved connection immediately occurs, a second "connection from" line could be next, with a different key (because it's a different connection), and that would result in another virtual log entry, with a different key and a different source_ip field. This allows both events to be built, a field at a time, without there being any conflict between them.

Nothing is added to the database when a "connection from" line is seen; it just files away the source_ip for later use, in log entry 156 (contrast this to all other parsing methods discussed so far, where every line results in an entry added to the database). Now it continues to the next line, which is the "sender" line. The "connection from" regular expression doesn't match, so it checks the "sender" regular expression, which does match; so it sets the value of the sender field to bob@somewhere.com, for log entry 156. On the next line, the third regular expression matches, and it sets the value of the recipient field to sue@elsewhere.com, for log entry 156. On the fourth line, the fourth regular expression matches, and it sets the date and time field for log entry 156.

At this point, log entry 156 looks like this:

source_ip: 12.34.56.67
sender: bob@somewhere.com
recipient: sue@elsewhere.com
date: 2005/03/22
time: 01:23:50

We have non-populated all fields, so it's time to put this entry in the database. That's what accept_collected_entry() does; it puts entry 156 into the database. From there, things proceed just as they would have if this had been a log format with all five fields on one line, extracted with a regular expression or with delimited parsing. So by using five "collect" operations, we have effectively put together a single virtual log entry which can now be put into the database in the usual way.

In order to use parsing filters with accept/collect, you must also set this option in the plug-in:

```
log.format.parse_only_with_filters = "true"
```

If you don't include this line, the parser will use delimited or regular expression parsing first, and then run the parsing filters; typically, you want the parsing filters solely to extract the data from the line.

The parsing filter language is a fully general language; it supports variables, nested if/then/else constructs, loops, subroutines, recursion, and anything else you would expect to find in a language. Therefore there is no limit to what you can do with parsing filters; any log format can be parsed with a properly constructed parsing filter.

The Log Fields

The log fields are variables which are populated by the parsing regular expression; they are used to hold values which are extracted from the log data. They will be used later as the main variables which are manipulated by log filters, and if a log entry is not rejected by the log filters, the log field values will be copied to the database fields to be inserted into the database.

The log fields are listed in the log.fields section of the profile. For the example above, the log fields would look like this:

```
log.fields = {

  date = ""
  time = ""
  ip_address = ""
  fieldx = ""
  fieldy = ""
  duration = ""

} # log.fields
```

The name of the date field must be "date"; the name of the time field must be "time."

The log.format.date_format and log.format.time_format options describe the format of the date and time fields in the log data. For instance, a value of dd/mmm/yyyy for log.format.date_format means that the date will look something like this: 01/Jan/2006. If these options are omitted from the plug-in, they will default to "auto", which will do its best to determine the format automatically. This almost always works for time, but some date formats cannot be automatically determined; for instance, there is no way to guess whether 3/4/2005 means March 4, 2005 or April 3, 2005 (in this case, auto will assume the month is first). In cases where "auto" cannot determine the format, it is necessary to specify the format in the log.format.date_format option in the plug-in. Available formats are listed in **Date format** and **Time format**.

Usually, the date and time fields are listed separately, but sometimes they cannot be separated. For instance, if the date/time

format is "seconds since January 1, 1970" (a fairly common format), then the date and time information is integrated into a single integer, and the date and time fields cannot be extracted separately. In this case, you will need to use a single date_time log field:

```
  log.fields = {

  date_time = ""
  ip_address = ""
  ...

} # log.fields
```

and both the log.format.date_format option and the log.format.time_format options should be set to the same value:

```
  log.format.date_format = "seconds_since_jan1_1970"
  log.format.time_format = "seconds_since_jan1_1970"
```

Fields which are listed without any parameters, like the ones above, will have default values of various parameters assigned to them. Most importantly, the label of the field will be set to "$lang_stats.field_labels.*fieldname*" where *fieldname* is the name of the field. For instance the label of the fieldx field will be set to "$lang_stats.field_labels.fieldx". This allows you to create plug-ins which are easily translated into other languages, but it also means that when you create a plug-in like this with no label value specified, you also need to edit the file LogAnalysisInfo/languages/english/lang_stats.cfg to add field labels for any new fields you have created. In that file, there is a large section called field_labels where you can add your own values to that section. For instance, in the case above, you would need to add these:

```
  fieldx = "field x"
  fieldy = "field y"
```

The other fields (date, time, ip_address, and duration) are already in the standard field_labels list. The standard list is very large, so you may find that *all* your fields are already there. If they aren't, you either need to add them, or explicitly override the default label by defining the log field like this:

```
  log.fields = {
    ...
    fieldx = {
      label = "field x"
    }
    ...
  } # log.fields
```

This extended syntax for the log field specifies the label in the plug-in, which means you don't need to edit lang_stats.cfg, but also means that the plug-in will not be translated into other languages. Plug-ins created for distribution with Sawmill should always use default field labels, and the field names should always be added to lang_stats, to allow for localization (translation to the local language).

The Database Fields

Sawmill stores information primarily in a single large table of the database called the Main Table. This table has one row per accepted log entry which usually means one row per line of log data, and one column per database field. The database.fields section of the plug-in lists these database fields.

The list of database fields is often similar to the list of log fields, but they aren't usually quite the same. Log fields list the fields which are actually present in the log data; database fields list the fields which are put into the database. These fields may be different if: 1) Some of the log fields are not interesting, and are therefore not tracked in the database or 2) A database field is based on a derived field (see below), which is not actually in the log data, or 3) The log field is numerical, and needs to be aggregated using a numerical database field (see next section).

Derived fields are "virtual" log fields which are not actually in the log data, but which are computed from fields which *are* in the log data. The following derived fields are available:

Log Field	Derived Log Field	Notes
date and time (both fields)	date_time	date_time is a field of the format dd/mmm/yyyy hh:mm:ss, e.g., 12/Feb/2006 12:34:50, which is computed from the date and time values in the log data.
date_time	hour_of_day	The hour of the day, e.g., 2AM-3AM.
date_time	day_of_week	The day of the week, e.g., Tuesday.
date_time	day_of_year	The day of the year, e.g., 1 for January 1, through 365 for December 31.
date_time	week_of_year	The week of the year, e.g., 1 for January 1 through January 8.
host	domain_description	A description of the domain for the host, e.g. "Commercial" for .com addresses. See the "host" comment below.

host	location	The geographic location of the IP address, computed by GeoIP database. See the "host" comment below.
agent	web_browser	The web browser type, e.g. "Internet Explorer/6.0". See the "agent" comment below.
agent	operating_system	The operating system, e.g. "Windows 2003". See the "agent" comment below.
agent	spider	The spider name, or "(not a spider)" if it's not a spider. See the "agent" comment below.
page	file_type	The file type, e.g., "GIF". See the "page" comment below.
page	worm	The worm name, or "(not a worm)" if it's not a worm. See the "page" comment below.

Note: "Host" derived fields are not necessarily derived from fields called "host"--it can be called anything. These fields are derived from the log field whose "type" parameter is "host". So to derive a "location" field from the ip_address field in the example above, the log field would have to look like this:

```
ip_address = {
  type = "host"
} # ip_address
```

You will sometimes see this shortened to this equivalent syntax:

```
ip_address.type = "host"
```

Note: "Agent" derived fields are similar to "host", the "agent" field can have any name as long as the type is "agent". However, if you name the field "agent", it will *automatically* have type "agent", so it is not necessary to list it explicitly unless you use a field name other than "agent".

Note: "Page" derived fields: similar to "host", the "page" field can have any name as long as the type is "page". However, if you name the field "page" or "url", it will *automatically* have type "page", so it is not necessary to list it explicitly unless you use a field name other than "page" or "url".

Never include a date or time field in the database fields list! The database field should always be the date_time field, even if the log fields are separate date and time fields.

When creating the database fields list, it is often convenient to start from the log fields list. Then remove any log fields you don't want to track, and add any derived fields you do want to track, and remove any numerical fields (like bandwidth, duration, or other "counting" fields), which will be tracked in the numerical fields (next section). For the example above, a reasonable set of database fields is:

```
database.fields = {

  date_time = ""
  ip_address = ""
  location = ""
  fieldx = ""
  fieldy = ""

} # database.fields
```

Numerical Database Fields

Normal database fields cannot be summed or aggregated numerically; values like "Monday" and "Tuesday" cannot be combined into an aggregate value; nor can values like "/index.html" and "/pricing.html". Fields like this are tracked by normal, non-numerical database fields (previous section). However some fields lend themselves to aggregation; if you have a bytes field with a value of 5, and another bytes field with a value of 6, it is reasonable to add them to get total bytes of 11. Similarly, if you have a 30 second duration, and another 15 second duration, you can compute a total duration by totaling them up to get 45 seconds. Log fields which can be summed to get total values are listed in the numerical fields section of the plug-in.

For the example above, the numerical fields could be this:

```
database.numerical_fields = {

  events = {
    label = "$lang_stats.field_labels.events"
    default = true
    requires_log_field = false
    type = "int"
    display_format_type = "integer"
    entries_field = true
  } # events

  duration = {
```

```
      label = "$lang_stats.field_labels.duration"
      default = true
      requires_log_field = true
      type = "int"
      display_format_type = "duration_compact"
    } # duration

  } # database.numerical_fields
```

This example lists two numerical fields, "events" and "duration". The duration field comes straight from the log data; it is just tracking and summing the duration values in the log data. The "events" field is a special field whose purpose is to count the total number of events, which typically means the total number of lines of log data; see below for details.

Most log formats have an "events" field which counts the number of events. Here is a list of names for events, depending on the log format:

"Hits" for web log formats.
"Accesses" for firewall/proxy/cache formats.
"Messages" for mail log formats.
"Packets" for packet tracking formats.

There are other names for other formats. The name should be the best name for one "event" in the log. When in doubt, just use "events". This field functions like any other numerical field, but its value is set to 1 for every line, using a log filter. Therefore, almost all plug-ins have at least one log filter (see below for more about log filters):

```
  log.filters = {

    mark_entry = {
      label = '$lang_admin.log_filters.mark_entry_label'
      comment = '$lang_admin.log_filters.mark_entry_comment'
      value = 'events = 1'
    } # mark_entry

  } # log.filters
```

This log filter has a label and a comment so it will appear nicely in the log filter editor, but the real value of the filter is 'events = 1'; all the filter really does is set the events field to 1. Many plug-ins do not require any other log filters, but this one is almost always present. **Make sure you always set the events field to 1!** If you omit it, some or all entries will be rejected because they have no non-zero field values.

Since the events field is always 1, when it is summed, it counts the number of events. So if you have 5,000 lines in your dataset, the Overview will show 5,000 for events, or the sum of events=1 over all log entries.

The parameters for numerical fields are:

<table>
<tr><th>Name</th><th>Purpose</th></tr>
<tr><td>label</td><td>This is how the fields will appear in the reports, e.g., this will be the name of the columns in tables. This is typically $lang_stats.field_labels.fieldname; if it is, this must be defined in the field_labels section of LogAnalysisInfo/languages/english/lang_stats.cfg, or it will cause an error when creating a profile.</td></tr>
<tr><td>default</td><td>"true" if this should be checked in the Numerical Fields page of the Create Profile Wizard.</td></tr>
<tr><td>requires_log_field</td><td>"true" if this should only be included in the database if the corresponding log field exists. If this is "false", the log field does not have to exist in the log.fields list; it will be automatically added. If this is "true", and the field does not exist in log.fields, it will not be automatically added; instead, the numerical field will be deleted, and will not appear in the database or in reports.</td></tr>
<tr><td>type</td><td>"int" if this is an integer field (signed, maximum value of about 2 billion on 32-bit systems); "float" if this is a floating point field (fractional values permitted; effectively no limit to size)</td></tr>
<tr><td>display_format_type</td><td>This specifies how a numerical quantity should be formatted. Options are:

<table>
<tr><td><code>integer</code></td><td>display as an integer, e.g., "14526554"</td></tr>
<tr><td><code>duration_compact</code></td><td>display as a compact duration (e.g., 1y11m5d 12:34:56)</td></tr>
<tr><td><code>duration_milliseconds</code></td><td>display as a duration in milliseconds (e.g., 1y11m5d 12:34:56.789)</td></tr>
<tr><td><code>duration_microseconds</code></td><td>display as a duration in microseconds (e.g., 1y11m5d 12:34:56.789012)</td></tr>
<tr><td><code>duration_hhmmss</code></td><td>display as a duration in h:m:s format (e.g., "134:56:12" for 134 hours, 56 minutes, 12 seconds)</td></tr>
<tr><td><code>duration</code></td><td>display as a fully expanded duration (e.g., "1 year, 11 months, 5 days, 12:34:56")</td></tr>
</table>
</td></tr>
</table>

	`bandwidth`	display as bandwidth (e.g., 5.4MB, or 22kb)
aggregation_method	"sum" if this field should be aggregated by summing all values; "average" if it should be aggregated by averaging all values; "max" if it should be aggregated by computing the maximum of all values; "min" if it should be aggregated by computing the minimum of all values. If not specified, this defaults to "sum". See below for more about aggregation.	
average_denominator_field	The name of the numerical database field to use as the denominator when performing the "average" calculation, when aggregation_method is average. This is typically the "entries" (events) field. See below for more information about aggregation.	
entries_field	"true" if this is the "entries" (events) field; omit if it is not	

The numbers which appear in Sawmill reports are usually aggregated. For instance, in the Overview of a firewall analysis, you may see the number of bytes outbound, which is an aggregation of the "outbound bytes" fields in every log entry. Similarly, if you look at the Weekdays report, you will see a number of outbound bytes for each day of the week; each of these numbers is an aggregation of the "outbound bytes" field of each log entry for that day. Usually, aggregation is done by summing the values, and that's what happens in the "outbound bytes" example. Suppose you have 5 lines of log data with the following values for the outbound bytes: 0, 5, 7, 9, 20. In this case, if the aggregation method is "sum" (or if it's not specified), the Overview will sum the outbound bytes to show 41 as the total bytes.

In some cases it is useful to do other types of aggregation. The aggregation_method parameter provides three other types, in addition to "sum": "min", "max", and "average". "Min" and "max" aggregate by computing the minimum or maximum single value across all field values. In the example above, the Overview would show 0 as the value if aggregation method was "min", because 0 is the minimum value of the five. Similarly, it would show 20 as the value if the aggregation method was "max". If the aggregation method was "average", it would sum them to get 41, and then divide by the average_denominator_field value; typically this would be an "entries" (events) field which counts log entries, so its value would be 5, and the average value shown in the Overview would be 41/5, or 8.2 (or just 8, if the type is "int").

Log Filters

In the log filters section, you can include one or more log filters. Log filters are extremely powerful, and can be used to convert field values (e.g., convert destination port numbers to service names), or to clean up values (e.g. to truncate the pathname portion of a URL to keep the field simple), or to reject values (e.g. to discard error entries if they are not interesting), or just about anything else. There aren't many general rules about when to use log filters, but in general, you should create the "events" filter (see above), and leave it at that unless you see something in the reports that doesn't look quite right. If the reports look fine without log filters, you can just use the events filter; if they don't, a log filter may be able to modify the data to fix the problem.

Database Field Associations

Database fields in most log formats can have associations, a numerical field is relevant to a non-numerical field; for instance, the number of hits is a relevant number for the day of the week, URL, file type, worm, and all other non-numerical fields. For log formats like that (including the example above), there is no need for database field associations, and you can skip this section. For some plug-ins, especially plug-ins which combine multiple disparate formats into a single set of reports, there may be some numerical fields which are not relevant to some non-numerical fields. For instance, if there is a single plug-in which analyzes both error logs (with date, time, and error_message fields, and a number_of_errors numerical fields), and access logs (with date, time, hostname, and page fields, and hits and page_views numerical fields), it does not make sense to ask how many errors there were for a particular hostname, or a particular page, because the error entries only contain date, time and error_message fields. In this case, you can use field associations to associate non-numerical fields with the numerical fields that track them. This makes reports more compact and more relevant, and makes cross-reference tables smaller, and eliminates all-zero columns from reports.

The database_field_associations node, like the report_groups node, goes inside create_profile_wizard_options. Here is an example, based on the integrated access/error log plug-in described above:

```
  create_profile_wizard_options = {

    # This shows which numerical fields are related to which non-numerical fields.
    database_field_associations = {

      hostname = {
        hits = true
        page_views = true
      }
      page = {
        hits = true
        page_views = true
      }
      error_message = {
        number_of_errors = true
      }

    } # database_field_associations
```

```
    report_groups = {
      ...
    }

  } # create_profile_wizard_options
```

Field associations affect two parts of the final profile: the reports, and the cross-reference groups. In the example above, the default hostname and page reports will include only hits and page_views columns, and not a number_of_errors column, and the default error_message report will include only a number_of_errors column, and not a hits or page_views column. Since cross-reference groups are tables which optimize the generation of particular reports, the associated cross-reference groups for those reports will also not have the non-associated fields. Fields which are missing from database_field_associations are assumed to be associated with all numerical fields, so the date_time reports would show all three numerical fields in this case (which is correct, because all log entries, from both formats, include date_time information). If the database_field_associations node itself is missing, all non-numerical fields are assumed to be associated with all numerical fields.

Report Groups: Automatic Report Creation

The report_groups section of a plug-in lists the reports and groups them. This can be done in a simple format, described in this section, or in a more complex and much more flexible format, described in the next section. In the simple approach, Sawmill automatically creates an Overview report, a Log Detail report, session reports if session information is available, a Log Detail report, and a Single-page Summary report which contains all other reports; in this approach, the purpose of the report groups description is to specify where these reports appear in the reports menu. In the complex (manual) approach, the report_groups specifies not just the position of reports, but also which reports are created, and what options they use.

Using the simpler automatic format, here is a good choice for the example discussed above:

```
  create_profile_wizard_options = {

    # How the reports should be grouped in the report menu
    report_groups = {
      date_time_group = ""
      ip_address = true
      location = true
      fieldx = true
      fieldy = true
    } # report_groups

  } # create_profile_wizard_options
```

The outer group is called create_profile_wizard_options, and it has only one node in it: report_groups, which describe reports and where they should appear in the left menu. In general, the list of groups will start with date_time_group = "", which specifies that there should be a group in the menu with date/time reports like years/months/days, days, day of week, and hour of day; this is a special option which is automatically expanded to this:

```
  date_time_group = {
    date_time = true
    day_of_week = true
    hour_of_day = true
  }
```

The rest of the database fields can be listed below the date_time_group; you can just grab them from the database.fields section, and change "" to "true". That will put all the reports at the top level of the menu, below the date/time group.

If there are too many reports to fit in a simple flat menu, you can add groups. For instance, you could group fieldx and fieldy like this:

```
  report_groups = {
    date_time_group = ""
    ip_address = true
    location = true
    fieldxy_group = {
      fieldx = true
      fieldy = true
    }
  } # report_groups
```

This will put the fieldx and fieldy reports in separate menu group called fieldxy_group. When you create a new group, you *must* also define it in LogAnalysisInfo/languages/english/lang_stats.cfg, in the menu.groups section (search for "menu = {"), so the web interface knows what to call the menu group, e.g. :

```
menu = {
  groups = {
    fieldxy_group = "Field X/Y"
    ...
```

Report Groups: Manual Report Creation

The automatic report creation approach described above is sufficient for simple plug-ins, but it has a number of significant drawbacks:

- There is always one report per database field; it is not possible to omit the report for any field.
- There is only one report per database field; it is not possible to create several reports for a field, with different options.
- You can not add filters to reports.
- It is not possible to customize which columns appear in reports; reports always contain one non-numerical field, and all associated numerical fields (see **Database Field Associations**).
- Reports cannot be created with graphs, except date/time reports; it is not possible to create date/time reports *without* graphs.
- It is not possible to override the number of rows, sort order, or any other report options.

The manual report creation approach described in this section overcomes all these limitations, because all reports, and their options, are specified manually. However, reports use default values for many options, so it is not necessary to specify very much information per report; in general, you only need to specify the non-default options for reports.

Here's an example of a very basic manual report creation, for the example above:

```
create_profile_wizard_options = {

  # How the reports should be grouped in the report menu
  manual_reports_menu = true
  report_groups = {

    overview.type = "overview"

    date_time_group = {
      items = {
        date_time = {
          label = "Years/months/days"
          graph_field = "events"
          only_bottom_level_items = false
        }
        days = {
          label = "Days"
          database_field_name = "date_time"
          graph_field = "events"
        }
        day_of_week = {
          graph_field = "events"
        }
        hour_of_day = {
          graph_field = "events"
        }
      }
    } # date_time_group

    ip_address = true
    location = true
    fieldxy_group = {
      items = {
        fieldx = true
        fieldy = true
      }
    }

    log_detail = true
    single_page_summary = true

  } # report_groups

} # create_profile_wizard_options
```

This has the same effect as the automatic report grouping described above.

Note:

1. The option "manual_reports_menu = true" specifies that manual report generation is being used.
2. The date_time group has been fully specified, as a four-report group.
3. The first report in the date_time group is the "Years/months/days" report, with label specified by lang_stats.miscellaneous.years_months_days (i.e., in LogAnalysisInfo/language/english/lang_stats.cfg, in the miscellaneous group, the parameter years_months_days), which graphs the "events" field and shows a hierarchical report (in other words, a normal "Years/months/days" report).
4. The second report in the date_time group is the "Days" report, with label specified by lang_stats.miscellaneous.days, which graphs the "events" field and shows a hierarchical report (in other words, a normal "Days" report).
5. The third report in the date_time group is a day_of_week report, which graphs the "event" field.
6. The fourth report in the date_time group is the hour_of_day report, which graphs the "event" field.
7. The ip_address, location, fieldx, and fieldy reports are specified the same as in automatic report creation, except for the addition of an "items" group within each group, which contains the reports in the group. Nothing is specified within the reports, so all values are default.
8. The log_detail and single_page_summary reports are specified manually (they will not be included if they are not specified here).

To simplify manual report creation, there are many default values selected when nothing is specified:

1. If no label is specified for a group, the label in lang_stats.menu.group.*groupname* is used (i.e., the value of the *groupname* node in the "group" node in the "menu" node of the lang_stats.cfg file, which is in LogAnalysisInfo/language/english). If no label is specified and the group name does not exist in lang_stats, the group name is used as the label.
2. If no label is specified for a report, and the report name matches a database field name, then the database field label is used as the report label. Otherwise, if lang_stats.menu.reports.*reportname* exists, that is used as the label. Otherwise, if lang_stats.field_labels.*reportname* exists, that is used as the label. Otherwise, *reportname* is used as the label.
3. If a "columns" group is specified in the report, that is used to determine the columns; the column field names are taken from the field_name value in each listed column. If it contains both numerical and non-numerical columns, it completely determines the columns in the report. If it contains only non-numerical columns, it determines the non-numerical columns, and the numerical columns are those associated with the database_field_name parameter (which must be specified explicitly in the report, unless the report name matches a database field name, in which case that database field is used as the database_field_name); see **Database Field Associations**. If no columns node is specified, the database_field_parameter is used as the only non-numerical column, and all associated numerical fields are used as the numerical columns.
4. If a report_menu_label is specified for a report, that value is used as the label in the reports menu; otherwise, the report label is used as the label in the reports menu.
5. If a "filter" is specified for a report, that filter expression is used as the report filter; otherwise, the report filter is used.
6. If only_bottom_level_items is specified for a report, the report shows only bottom level items if the value is true, or a hierarchical report if it is false. If it is not specified, the report shows only bottom level items.
7. If graph_field is specified for a report, a graph is included which graphs that field.
8. Any other options specified for a report will be copied over to the final report. For instance, graphs.graph_type can be set to "pie" to make the graph a pie chart (instead of the default bar chart), or "ending_row" can be set to change the number of rows from the default 20.

For instance, here is an advanced example of manual report grouping, again using the example above:

```
create_profile_wizard_options = {

  # How the reports should be grouped in the report menu
  manual_reports_menu = true
  report_groups = {

    overview.type = "overview"

    date_time_group = {
      items = {
        days = {
          label = "Days"
          database_field_name = "date_time"
          graph_field = "duration"
        }
        day_of_week = {
          graph_field = "duration"
        }
      }
    } # date_time_group

    ip_address = true
    location = true
    fieldxy_group = {
      label = "XY"
```

```
      items = {

        fieldx = {
          label = "Field X Report (XY Group)"
          report_menu_label = "Field X Report"
        }

        fieldy = {
          sort_by = "events"
          sort_direction = "ascending"
          graph_field = "events"
          graphs.graph_type = "pie"
        } # fieldy

        fieldx_by_fieldy = {
          label = "FieldX by FieldY"
          ending_row = 50
          columns = {
            0.field_name = "fieldx"
            1.field_name = "fieldy"
            2.field_name = "events"
            3.field_name = "duration"
          } # columns
        } # fieldx_by_fieldy

      } # items

    } # fieldxy_group

    log_detail = true

  } # report_groups

} # create_profile_wizard_options
```

Notes About the Example Above :

- The date_time group has been simplified; the hierarchical years/months/days report has been removed, as has the hours of day report.
- The label of the fieldxy group has been overriden to "XY"
- The fieldx report has a custom label, "Field X Report (XY Group)", but it has a different label, "Field X Report" for its entry in the reports menu.
- The fieldy report is sorted ascending by events, and includes an "events" pie chart.
- A new report has been added, fieldx_by_fieldy (with label "FieldX by FieldY"), which is a 50-row table showing both fieldx and fieldy. This report will aggregate totals for each fieldx/fieldy pair, and will show the number of events and total duration for each pair, in an indented two-column table.
- The single-page summary has been omitted.

Debugging

Log format plug-ins are almost always too complex to get right the first time; there is almost always a period of debugging after you've created one, where you fix the errors. By far the most useful debugging tool available is the command-line database build with the -v option. Once you've created a profile from the plug-in, build the database from the command line like this:

```
sawmill -p profilename -a bd -v egblpfdD | more
```

(use SawmillCL.exe on Windows). That will build the database, and while it's building, it will give *detailed* information about what it's doing. Look for the lines that start with "Processing" to see Sawmill looking at each line of log data. Look for the lines that start with "Marking" to see where it's putting data into the database. Look at the values it's putting in to the database to see if they look right. In between, look for the values it's extracting from the log data into the log fields, to be sure the fields values are what they should be. If you're using regular expressions to parse, Sawmill will show you what the expression is, and what it's matching it against, and whether it actually matched, and if it did, what the subexpressions were. Careful examination of this output will turn up any problems in the plug-in.

When you've found a problem, fix it in the plug-in, then run this to recreate your profile:

```
sawmill -p profilename -a rp
```

Then rerun the database build with the command above, and repeat until everything seems to be going smoothly. If the data seems to be populating into the database properly, switch to a normal database build, without debugging output:

```
sawmill -p profilename -a bd
```

When the build is done, look at the reports in the web interface; if you see any problems, you can return to the debugging output build to see how the data got in there.

When you have your log format plug-in complete, please send it to us! We'd be delighted to include your plug-in as part of Sawmill.

Creating Plug-ins for Syslog Servers and Syslogging Devices

To analyze log data logged to a syslog server in Sawmill, you must have two plug-ins: one to analyze the syslog header, and one to analyze the device's message. Since Sawmill supports so many syslog formats and syslogging devices already, it is not usually necessary to add both; usually, you'll find that the syslog format is already supported, so you only need to create the logging device plug-in, or vice versa. But if neither is supported already, you will need to create both plug-ins to process the data.

A syslog plug-in is slightly different from a normal plug-in. First, the log.miscellaneous.log_data_type option is set to "syslog":

```
log.miscellaneous.log_data_type = "syslog"
```

Secondly, the plug-in should only define log fields and database fields which are in the syslog header. These always include date and time; often they include the logging device IP address, and sometimes they include the syslog priority or other fields. Syslog plug-ins must *always* use log parsing filters to collect field values into the collected entry with the empty key, for example:

```
set_collected_field('', 'date', $1)
```

See above for information on using set_collected_field() to collect fields into entries. Syslog plug-ins must set the variable volatile.syslog_message to the message field. Syslog plug-ins should *not* accept entries; that is the responsibility of the syslogging device plug-in. Syslog plug-ins should always use "auto" as the date and time format; if the actual format is something else, they must use normalize_date() and normalize_time() to normalize the date and time into a format accepted by "auto". Other than that, syslog plug-ins are the same as other plug-ins.

A syslogging device plug-in (a.k.a., a syslog_required plug-in) is also slightly different from a normal plug-in. First, the log.miscellaneous.log_data_type option is set to "syslog_required":

```
log.miscellaneous.log_data_type = "syslog_required"
```

Secondly, the plug-in should only define log fields and database fields which are in the syslog message. These vary by format, but do *not* include date, time, logging device IP, or syslog priority. Syslogging device plug-ins must *always* use log parsing filters. Since the syslog plug-in collected date and time into the empty key entry, syslogging device plug-ins must copy those over to another key if they use keyed collected entries. If they do not use keys, they can just collect all their fields into the collected entry. The syslogging device should accept collected entries.

Getting Help

If you have any problems creating a custom format, please contact **support@flowerfire.com** -- we've created a lot of formats, and we can help you create yours. If you create a log format file for a popular format, we would appreciate it if you could email it to us, for inclusion in a later version of Sawmill.

Using the Sawmill Scheduler

Scheduled Items - Using the Scheduler

Sawmill includes a built-in scheduler which can be used to schedule regular database builds, database updates, database expirations, off-line HTML page generation, emailed reports, and more. For instance, you can use it to schedule a profile's database to be updated every day at midnight, so the statistics are never more than a day old. You could also schedule Sawmill to generate HTML pages from your statistics every week, so you have an off-line browseable HTML snapshot of your weekly statistics. The Scheduler can also update the database every day, and after that, send you the statistics by email.

Scheduled items can only be run when Sawmill is running. The easiest way to do this is to run Sawmill in web server mode. If you need to use Sawmill's scheduler features with CGI mode, you can do it by using an external scheduling program like cron or NT Scheduler to run Sawmill every minute in scheduler mode, i.e. `sawmill -scheduler`. Sawmill can be called directly from cron or NT Scheduler as well, using a command line to describe the operation to be performed.

A scheduled item consists of a date and time, a profile name (or matching pattern), an operation, and possibly extra options. Sawmill will perform the operation on the profile whenever the date and time match the current date and time. For instance if you choose January, Monday, day 12 of the month, 06:09 as your date and time, and Build Database as the operation, Sawmill will build the database at 6:09 AM on any Monday which is the 12th of January. The current date and time must match *all* the selections, so it won't run at 6:09 AM on Monday the 13th of January, or at 6:09 AM on Tuesday the 12th of January. Usually, you will select "any month," "any day," or * (which means any hour or any minute) for some of the fields. For instance, to update a database every hour, you might choose any month, any day, any day, *:00, Update Database.

Sawmill calls itself from the command line to perform scheduled tasks. The extra options, if specified, are added to the end of the command line. This makes it possible to perform complex scheduled actions by overriding the default options on the command line. For instance without any extra options, sending email will send the default view by email to the default address, without any filters. But if the options are set to `-ss smtp.somewhere.com -rea fromsomeone@somewhere.com -rca someone@somewhere.com -rn single_page_summary -f recentdays:30`, the Single-page summary of the past 30 days will be sent to someone@somewhere.com, from fromsomeone@somewhere.com, using the SMTP server at smtp.somewhere.com. A list of available reports can be displayed by running Sawmill from the command line with the "-p profilename -a lr" options.

Sawmill creates a file called TaskLog, in the LogAnalysisInfo directory, which contains a log of all scheduled tasks that have run, as well as logs of all other actions Sawmill has taken (e.g. view statistics, build database from the web interface or command line, etc.). This log can be very helpful if you're trying to debug a problem with the Scheduler.

Language Modules--Localization and Text Customization

All text displayed by Sawmill can be modified, customized, or translated by modifying Sawmill's Language Module files. This includes the links at the top of the interface, the name of table headers in the reports and this documentation. Language modules are text files, located in the languages subfolder of the LogAnalysisInfo subfolder. Documentation is separate, in the docs subfolder, which contains all the text Sawmill ever displays. The documentation files can be duplicated, renamed, and modified to provide translations into any language. By modifying the language modules, or creating your own, you can translate Sawmill's interface into any language, or you can customize any text that Sawmill generates. Some language translations already exist; visit the Sawmill web site for more information. If you need translations into other languages, that will be under your own discretion.

There are four language module files which together define every bit of text Sawmill ever generates (plus, there are the documentation files).

They are:

- **The statistics reports - lang_stats.cfg**: This includes the names of the reports, the names of the fields, the instructions, the names of the options, the table headings and more. This small language module is all that must be translated to translate Sawmill's statistics into any language.
- **The administrative interface - lang_admin.cfg**:. This includes the text of the profiles list, the configuration page, the Scheduler and other administrative pages.
- **The options interface - lang_options.cfg**: The names and descriptions of all of Sawmill's options.
- **The error messages interface - lang_messages.cfg**: The error messages and debugging output.

For example, if all you need is a translation of the statistics you can use the English administrative interface, but you need to provide statistics in the native language of your clients, then you only need a translation of the lang_stats module. If you need to be able to use the profile list, configuration page, Scheduler, or other features in a language other than English, you will need translations of the lang_admin, lang_options, and lang_messages modules.

Creating a New Language Module

To begin translating, duplicate the "english" folder in the languages folder of the LogAnalysisInfo folder, and change its name to the new language and then edit the files inside it. To switch languages, change the language specified in the preferences.cfg file of the LogAnalysisInfo folder to match the name of the folder you created.

Language Module Syntax

Language modules are configuration files, and their syntax is described in **Configuration Files**.

When translating or customizing a language module, you should not change the names of variables--those must remain exactly as they are. Only the values should be changed.

Debugging Language Modules

When you write a language module, you must be precise in the syntax and even a small error, like an unclosed quote, can make the entire file unreadable by Sawmill. This may result in a situation where Sawmill can't even report the error properly because the error message is in the corrupted language module.

This sort of a problem is difficult to track down, so a special feature of Sawmill lets you debug language modules easily. It requires a command-line version of Sawmill, and you must change the name of the Sawmill executable so that it contains the characters "lmd" (that stands for Language Module Debug) -- for instance, you might change the name of the program to `sawmill_lmd`. Once you've done that, run Sawmill. When it notices that its name contains "lmd", it will start generating information about the language modules it's reading, including the tokens it's processing and the name/value pairs it's finding. It is usually a simple matter to examine the output and see where the erroneous quote or bracket is.

Special Language Module Variables

Language modules are configuration files, and can contain references to any other configuration variables, using the standard dollar-sign syntax. Any variable in the configuration hierarchy can be referenced in this way.

Some special variables are also available:

- RUNNING_USERNAME: The name of user Sawmill is running as.
- PRODUCT_NAME: The name of Sawmill ("Sawmill" by default, but different if the product has been relabelled or white-labelled).
- PRODUCT_EXECUTABLE: The name of the Sawmill executable file ("SawmillCL.exe" in this case).
- PRODUCT_EXECUTABLE_DOCS: The name of the Sawmill executable file, as used in documentation where a generic executable name is more useful than the actual one ("sawmill" in this case).
- PRODUCT_URL: A link to the Sawmill home page ("http://www.sawmill.net/" in this case).
- COPYRIGHT_HOLDER: The holder of the Sawmill copyright ("Flowerfire" in this case).

Supported Log Formats

Sawmill supports many different log formats using an extensible plug-in architecture. This version of Sawmill supports the following formats. Click a category below to jump to that format category.

Web Server

- Apache/NCSA Combined Log Format
- Apache/NCSA Combined Format With Server Domain After Agent
- Apache/NCSA Combined Format With Server Domain After Date
- Apache/NCSA Combined Format With Server Domain After Host
- Apache/NCSA Combined Format With Server Domain After Size (e.g. 1&1, Puretec)
- Apache/NCSA Combined Format With Server Domain Before Host
- Apache/NCSA Combined Format With Cookie Last
- Apache/NCSA Combined Format With Visitor Cookie
- Apache/NCSA Combined Format With WebTrends Cookie
- Apache Custom Log Format
- Apache Error Log Format
- Apache SSL Request Log Format
- BeatBox Hits Log Format (default)
- BEA WebLogic
- Apache/NCSA Combined Log Format (BETA)
- Flex/JRun Log Format
- msieser HTTP Log Format (BETA)
- Blue Coat W3C Log Format (ELFF)
- ColdFusion Web Server Log Format
- Common Access Log Format
- Common Access Log Format (Claranet)
- Common Access Log Format (WebSTAR)
- Common Access Log Format, with full URLs
- Apache/NCSA Common Agent Log Format
- Common Error Log Format
- Common Referrer Log Format
- Domino Access Log Format
- Domino Agent Log Format
- Domino Error Log Format
- Flash Media Server Log Format
- W3C Log Format
- IBM Tivoli Access Manager Log Format
- IIS Log Format
- IIS Log Format (dd/mm/yy dates)
- IIS Log Format (dd/mm/yyyy dates)
- IIS Extended Log Format
- IIS Log Format (mm/dd/yyyy dates)
- IIS Extended (W3C) Web Server Log Format
- IIS Log Format (yy/mm/dd dates)
- Miva Access Log Format
- Miva Combined Access Log Format
- Netscape Extended Log Format
- NetPresenz Log Format
- NetPresenz Log Format (d/m/y dates)
- NetPresenz Log Format (24-hour times, d/m/y dates)
- PeopleSoft AppServer Log Format
- Sambar Server Log Format
- SecureIIS Log Format
- SecureIIS Binary Log Format (SUPPORTED ONLY AFTER TEXT EXPORT)
- Symantec Mail Security Log Format
- Tomcat Log Format
- TomcatAlt

- URLScan Log Format
- URL-Scan (W3C) Log Format
- Web Logic 8.1 Log Format
- WebSTAR Log Format
- WebSTAR W3C Web Server Log Format
- Zeus Log Format (Alternate Dates)
- Zeus Extended Log Format

Uncategorized

- Aventail Client/server Access Log Format
- Metavante Log Format

Syslog Server

- Snare Log Format (BETA)
- GNAT Box Syslogger (v1.3) Syslog
- Imail Header
- IPCop Syslog
- Kiwi CatTools CatOS Port Usage Format
- Kiwi (dd-mm-yyyy dates)
- Kiwi Syslog (ISO/Sawmill)
- Kiwi (mm-dd-yy dates, with type and protocol)
- Kiwi (mm-dd-yyyy dates)
- Kiwi (mmm/dd dates, hh:hh:ss.mmm UTC times)
- Kiwi Syslog (UTC)
- Kiwi YYYYMMDD Comma Syslog
- Minirsyslogd Log Format
- MM/DD-HH:MM:SS Timestamp
- Network Syslog Format
- No Syslog Header (use today's date, or use date/time from message)
- NTsyslog Log Format
- Passlogd Syslog Format
- Passlogd Syslog (Full Messages)
- PIX Firewall Syslog Server Format
- PIX Firewall Syslog Server (no year) (EMBLEM)
- Seconds since Jan 1 1970 Timestamp Syslog
- SL4NT Log Format
- SL4NT (dd/mm/yyyy)
- SL4NT (dd.mm.yyyy, commas without spaces)
- SLNT4 Log Format
- Snare Log Format
- Complete Syslog Messages (report full syslog message in one field)
- Syslog NG Log Format
- Syslog NG Messages Log Format
- Syslog NG Log Format (no timezone)
- Syslog NG Log Format (no date in log data; yyyymmdd date in filename)
- Syslog (yyyymmdd hhmmss)
- Timestamp (mm dd hh:mm:ss)
- Unix Syslog
- Unix Syslog With Year
- Wall Watcher Log Format
- Windows NT Syslog
- Windows Syslog Format
- WinSyslog

Proxy Server

- Blue Coat Log Format
- Blue Coat Log Format (Alternate)
- Blue Coat Custom Log Format
- Blue Coat Squid Log Format
- Combined Proxy Log Format
- Common Proxy Log Format
- EZProxy Log Format
- Microsoft Port Reporter Log Format
- Microsoft Proxy Log Format
- Microsoft Proxy Log Format (d/m/yy dates)
- Microsoft Proxy Log Format (d/m/yyyy dates)
- Microsoft Proxy Log Format (m/d/yyyy dates)
- Microsoft Proxy Packet Filtering Log Format
- Microsoft Proxy Log Format (Bytes Received Field Before Bytes Sent)
- ProxyPlus Log Format

- Proxy-Pro GateKeeper Log Format
- Squid Common Log Format
- Squid Event Log
- Squid Log Format With Full Headers
- Squid Guard Log Format
- Squid Log Format
- Useful Utilities EZproxy Log Format
- VICOM Gateway Log Format
- Vicomsoft Internet Gateway Log Format
- Winproxy Log Format
- Winproxy Log Format (2-digit years)
- Winproxy Common Log Format
- Kerio Winroute Firewall Log Format
- WinGate Log Format (no Traffic lines, dd/mm/yy dates)
- WinGate Log Format (no Traffic lines, mm/dd/yy dates)
- WinGate Log Format (with Traffic lines)
- Winproxy 5.1 Log Format (yyyy-mm-dd dates)
- WinProxy Alternate Log Format
- Youngzsoft CCProxy

Other

- Array 500 Combined Log Format
- Ascend Log Format
- Atom Log Format
- du Disk Usage Tracking Format (find updatetest -type f | xargs du) (BETA)
- Java Administration MBEAN Log Format (BETA)
- Nessus Log Format (BETA)
- praudit Log Format (BETA)
- Snare for AIX Log Format (BETA)
- Sourcefire IDS (BETA)
- Symantec Antivirus Log Format (BETA)
- Symantec System Console Log Format (BETA)
- Trend Micro Control Manager (BETA)
- BitBlock Log Format
- Blue Coat Instant Messenger Log Format
- Borderware runstats Log Format
- Click To Meet Log Format
- Dade Behring User Account Format (With Duration)
- Dade Behring User Log Format
- Digital Insight Magnet Log Format
- Eventlog to Syslog Format
- Event Reporter Logs (version 7)
- Event Reporter v6
- FastHosts Log Format
- FedEx Tracking Log Format
- CSV (Generic Comma-Separated Values) Log Format
- Google Log Format
- GroupWise Post Office Agent Log Format
- Groupwise Web Access Log Format (dd/mm/yy)
- Groupwise Web Access Log Format (mm/dd/yy)
- GroupWise Internet Agent Accounting Log Format (2-digit years)
- GroupWise Internet Agent Accounting Log Format (4-digit years)
- Hosting.com Log Format
- htdig Log Format
- InfiNet Log Format
- INN News Log Format
- INN News Log Format (Alternate)
- IOS Debug IP Packet Detailed (Using Syslog Server)
- ipchains Log Format
- IPEnforcer
- IPMon Log Format (Using Syslog Server)
- IST Log Format
- Novell iChain Extended (W3C) Web Server Log Format
- iPlanet Error Log Format
- Lava2 Log Format
- Sawmill Task Log Format
- Microsoft Elogdmp (CSV) Log Format (CSV)
- Netscape Messenger Log Format
- NetKey Log Format
- nmap Log Format
- Optima Log Format
- Order Log Format
- O'Reilly Log Format

- Planet-Share InterFax Log Format
- PsLogList Log Format
- RACF Security Log Format
- RAIDiator Error Log Format
- Redcreek System Message Viewer Format
- Servers Alive Log Format
- Servers Alive (Statistics) Log Format
- SIMS Log Format
- Sysreset Mirc Log Format
- tcpdump Log Format (-tt)
- tcpdump Log Format
- tcpdump Log Format (-tt, with interface)
- tcpdump Log Format (-tt, with interface) Alternate
- Tellique Log Format
- Unicomp Guinevere Log Format
- Unicomp Guinevere Virus Log Format
- WAP
- WebSphere Business Integration Message Brokers User Trace Log Format
- WebSEAL Audit Log Format
- WebSEAL Authorization (XLM) Log Format
- WebSEAL Error Log Format
- WebSEAL Security Manager Log Format
- WebSEAL Wand Audit Log Format
- WebSEAL Warning Log Format
- WebSEAL CDAS Log Format
- Welcome Log Format
- Whatsup Syslog
- WhistleBlower (Sawmill 6.4)
- Whistle Blower Performance Metrics Log
- Win2K Performance Monitor
- Windows 2000/XP Event Log Format (export list-CSV) ddmmyyyy
- Windows 2000/XP Event Log Format (save as-CSV) dd/mm/yyyy
- Windows Event Log Format (24 hour times, d/m/yyyy dates)
- Windows Event Log Format (ALTools export)
- Windows Event (Comma Delimited, m/d/yyyy days, h:mm:ss AM/PM times) Log Format
- Windows Event (Comma Delimited) Log Format
- Windows Event Log Format (dumpel.exe export)
- Windows Event Log Format (dumpevt.exe export)
- Windows Event .evt Log Format (SUPPORTED ONLY AFTER CSV OR TEXT EXPORT)
- Windows XP Event Log (Microsoft LogParser CSV Export)
- Windows Event (Tab Delimited) Log Format
- Windows NT4 Event Log Format (save as-CSV)
- Windows NT Scheduler Log Format
- X-Stop Log Format
- Yamaha RTX Log Format

Network Device

- 3Com Office Connect / WinSyslog Log Format
- Apple File Service Log Format
- AppleShare IP Log Format
- Annex Term Server (BETA)
- Bintec VPN 25 or XL (BETA)
- Cisco As5300 Log Format (BETA)
- Firepass Log Format (BETA)
- Intermapper Event Log Format (BETA)
- Internet Security Systems Network Sensors (BETA)
- Lancom Router (BETA)
- Nortel Contivity Log Format (BETA)
- Bind 9 Query Log Format
- Bind 9 Log Format (Syslog required)
- Bind 9 Query Log Format (with timestamp)
- Bindview Reporting Log Format
- Bindview User Logins Log Format
- Bindview Windows Event Log Format
- Bind Query Log Format
- Bind Query Log Format With Timestamp
- Bind Response Checks Log Format
- Bind Security Log Format
- Bind 9 Update Log Format (with timestamp)
- bpft4 Log Format
- bpft4 Log Format (with interface)
- bpft traflog Log Format
- Cisco 827 Log Format (Kiwi, Full Dates, Tabs)

- CiscoWorks Syslog Server Format
- Cisco 3750 Log Format
- Cisco Access Control Server Log Format
- Cisco Access Register
- Cisco ACNS log w/ SmartFilter
- Cisco AS5300 - 2
- Cisco CE Log Format
- Cisco CE Common Log Format
- Cisco EMBLEM Log Format
- Cisco IDS Netranger Log Format
- Cisco NetFlow
- Cisco NetFlow (version 1)
- Cisco NetFlow Binary (DAT) Log Format (SUPPORTED ONLY AFTER ASCII EXPORT)
- Cisco NetFlow (FlowTools ASCII Export)
- Cisco NetFlow (flow-export)
- Cisco NetFlow (no dates)
- Cisco Router Log Format (Using Syslog Server)
- Cisco Router Log Format (no syslog)
- Cisco SCA Log Format
- Cisco Secure Server (RAS Access) Log Format
- Cisco SOHO77
- Cisco Voice Router
- Cisco VPN Concentrator
- CiscoVPNConcentratorAlt
- Cisco VPN Concentrator (Comma-delimited)
- Cisco VPN Concentrator (Comma separated - MMDDYYYY)
- Cisco VPN Concentrator Syslog Log Format
- Clavister Firewall Binary Log Format (SUPPORTED ONLY AFTER FWLoggqry.exe EXPORT)
- DLink DI-804HV Ethernet Broadband VPN Router Log Format
- Ethereal
- Ethereal/tcpdump Binary Log Format (SUPPORTED ONLY AFTER -r -tt CONVERSION)
- Foundry Networks BigIron
- Free Radius Log Format
- Intel NetStructure VPN Gateway Log Format
- Intermapper Outages Log Format (dd mmm yyyy dates, 24-hour times)
- Intermapper Outages Log Format (mmm dd yyyy dates, AM/PM times)
- Interscan E-mail Log Format
- Interscan E-mail Viruswall Log Format
- Interscan Proxy Log Format (dd/mm/yyyy dates)
- Interscan Proxy Log Format (mm/dd/yyyy dates)
- Interscan Viruswall Virus Log Format
- InterScan Viruswall Log Format
- iptables Log Format
- IPTraf Log Format
- IP Traffic LAN Statistics Log
- IPTraf TCP/UDP Services Log Format
- ISC DHCP Log Format
- ISC DHCP Leases Log Format
- Kerio Network Monitor Log Format
- Kerio Network Monitor HTTP Log Format
- LinkSys Router Log Format
- MonitorWare
- MonitorWare (Alternate)
- Nagios Log Format
- Neoteris Log Format
- Netgear FVS318
- Netgear Security Log Format
- Netgear Security Log Format (logging to syslog)
- Net-Acct
- NetForensics Syslog Format
- NetGear Log Format
- NetGear DG834G Log Format
- NetGear FR328S Log Format
- Nortel Contivity Log Format
- Nortel Networks RouterARN Format (SUPPORTED ONLY AFTER TEXT EXPORT)
- Radius Accounting Log Format
- Radius Accounting Log Format II
- Radius ACT Log Format
- Radware Load Balancing (Using Syslog Server)
- Simple DNS
- SiteMinder WebAgent Log Format
- SNMP Manager Log Format
- Snort Log Format (syslog required)
- Snort 2 Log Format (syslog required)
- SNORT Portscan Log Format

- Snort Log Format (standalone, mm/dd dates)
- Snort Log Format (standalone, mm/dd/yy dates)
- Socks 5 Log Format
- Steel Belted Radius ACT Log Format
- TrendMicro/eManager Spam Filter Log Format
- Trend Micro ScanMail For Exchange Log Format
- Trend ServerProtect CSV Admin Log Format
- Trend Webmanager Log Format
- Vidius Combined Log Format
- Watchguard Binary (WGL) Log Format (SUPPORTED ONLY AFTER TEXT EXPORT)
- Windows 2003 DNS Log Format
- ZyXEL Communications Log Format

Media Server

- Blue Coat RealMedia Log Format
- Blue Coat Windows Media Log Format
- Helix Universal Server Log Format
- Helix Universal Server (Style 5) Log Format
- IceCast Log Format
- IceCast Alternate Log Format
- Microsoft Media Server Log Format
- Quicktime/Darwin Streaming Server Log Format
- Quicktime Streaming Error Log Format
- RealProxy Log Format
- RealServer Log Format
- RealServer Log Format, Alternate
- RealServer Error Log Format
- Shoutcast 1.6 Log Format
- Shoutcast 1.8+ Log Format
- SHOUTcast W3C Log Format

Mail Server

- Aladdin Esafe Gateway Log Format
- Aladdin eSafe Mail Log Format
- Aladdin eSafe Sessions Log Format
- Aladdin eSafe Sessions Log Format v5
- Aladdin eSafe Sessions (with URL category) Log Format
- Amavis Log Format
- Anti-Spam SMTP Proxy (ASSP) Log Format
- Argosoft Mail Server Log Format
- Barracuda Spam Firewall
- Barracuda Spyware Firewall
- Amavis Log Format (BETA)
- Argosoft Mail Server Log Format (BETA)
- Communigate Pro Log Format (BETA)
- EIMS Error Log Format (BETA)
- Exim 4 Log Format (BETA)
- IIS SMTP Comma Separated Log Format
- Interscan Messaging Security Suite Integrated Log Format (BETA)
- IronPort Log Format (BETA)
- Kaspersky Log Format (BETA)
- Keria Mailserver Mail Log Format (BETA)
- Mailman Post Log Format (BETA)
- Mail Enable W3C Log Format (BETA)
- McAfee E1000 Mail Scanner (BETA)
- msieser SMTP Log Format (BETA)
- Postfix Log Format (BETA)
- Scanmail For Exchange Log Format (BETA)
- Sendmail Log Format (BETA)
- Sophos Antispam PMX Log Format
- Centrinity FirstClass Log Format
- Centrinity FirstClass (m/d/yyyy) Log Format
- ClamAV
- Communigate Log Format
- Communigate Pro Log Format
- Critical Path Mail Server POP/IMAP Log Format
- Critical Path Mail Server SMTP Log Format
- Declude SPAM
- Declude Virus
- EIMS SMTP (12 hour) Log Format
- EIMS SMTP (24 hour) Log Format
- EmailCatcher

- Microsoft Exchange Internet Mail Log Format
- Exim Log Format
- Exim 4 Log Format
- FirstClass Server Log Format
- GFI Attachment & Content Log Format
- GFI Spam Log Format
- GMS POP Log Format
- GMS POST Log Format
- GMS SMTP Log Format
- GW Guardian Antivirus Log Format
- GW Guardian Spam Log Format
- IIS SMTP Common Log Format
- IIS SMTP W3C Log Format
- IMail (7/8) Log Format
- Interscan Messaging Security Suite Log Format
- Iplanet Messenger Server 5 Log Format
- Ironmail AV Log Format (Sophos)
- Ironmail CSV Log Format
- Ironmail SMTPO Log Format
- Ironmail SMTP Proxy Log Format
- Ironmail Sophosq Log Format
- Ironmail Spam Log Format
- IronPort Log Format
- IronPort Bounce Log Format
- iMail Log Format
- iMail Log Format, Alternate
- iPlanet Messaging Server 5 MTA Log Format
- Kaspersky Labs for Mail Servers (linux) Log Format
- LISTSERV Log Format
- LogSat SpamFilterISP Log Format B500.9
- Lucent Brick (LSMS) Admin Log Format
- LSMTP Log Format
- LSMTP Access Log Format
- Lyris MailShield Log Format
- Mailer Daemon Log Format
- Mailman Subscribe Log Format
- mailscanner Log Format
- Mail Enable W3C Log Format
- Mail Essentials Log Format
- MailMax SE Mail POP Log Format
- MailMax SE SMTP Log Format
- MailScanner Log Format (testfase)
- MailScanner Virus Log Format (email messages sent)
- MailStripper Log Format
- MailSweeper (AM/PM) Log Format
- MailSweeper (24 Hour) Log Format
- MailSweeper (long) Log Format
- MDaemon 7 Log Format
- MDaemon 7 (All) Log Format
- MDaemon 8 (All) Log Format
- Merak POP/IMAP Log Format
- Merak SMTP Log Format
- Microsoft Exchange Server Log Format
- Microsoft Exchange Server 2000/2003 Log Format
- Microsoft Exchange Server 2000 Log Format (comma separated)
- Mirapoint SMTP Log Format
- MTS Professional Log Format
- NEMX PowerTools for Exchange
- Novell NetMail Log Format
- Novell NetMail 3.5 Log Format
- Openwave Intermail Log Format
- Postfix Log Format
- Postfix II Log Format
- Post Office Mail Server Log Format
- PostWorks IMAP Log Format
- PostWorks POP3 Log Format
- PostWorks SMTP Log Format
- qmail-scanner Log Format
- qmail (Syslog Required) Log Format
- qmail Log Format (TAI64N dates)
- SendmailNT Log Format
- SmartMaxPOP Log Format
- SmartMaxSMTP Log Format
- Sophos Antispam Message Log Format
- Sophos Mail Monitor for SMTP

- SpamAssassin Log Format
- Symantec Gateway Security 2 (CSV) Log Format
- UNIX Sendmail Log Format
- uw-imap Log Format
- Web Washer Log Format
- WinRoute Mail Log Format
- XMail SMTP Log Format
- XMail Spam Log Format

Internet Device

- McAfee Web Shield XML Log Format (BETA)
- DansGuardian 2.2 Log Format
- DansGuardian 2.4 Log Format
- DansGuardian 2.9 Log Format
- ISS Log Format
- iPrism Monitor Log Format
- iPrism-rt Log Format
- iPrism (with syslog)
- McAfee Web Shield Log Format
- Message Sniffer Log Format
- N2H2 Log Format
- N2H2 / Novell Border Manager Log Format
- N2H2 Sentian Log Format
- Netegrity SiteMinder Access Log Format
- Netegrity SiteMinder Event Log Format
- Netilla Log Format
- NetApp Filers Audit Log Format
- NetCache NetApp Log Format
- NetCache NetApp 5.5+ Log Format
- Packet Dynamics Log Format
- Privoxy Log Format
- Separ URL Filter Log Format
- Websweeper Log Format

FTP Server

- BDS FTP Log Format
- Bulletproof/G6 FTP Log Format (dd/mm/yy dates, 24-hour times)
- Bulletproof/G6 FTP Log Format (dd/mm/yyyy dates)
- Bulletproof/G6 FTP Log Format (dd/mm/yyyy dates, 24 hour times)
- Bulletproof/G6 FTP Log Format (mm/dd/yy dates)
- Bulletproof/G6 FTP Log Format (mm/dd/yyyy dates)
- Bulletproof/G6 FTP Sessions Log Format
- Bulletproof/G6 FTP Log Format (yyyy/mm/dd dates)
- FileZilla Server Log Format
- Flash FSP Log Format
- Gene6 FTP Log Format
- IIS FTP Server Log Format
- MacOS X FTP Log Format
- NcFTP Log Format (Alternate)
- NcFTP Xfer Log Format
- ProFTP Log Format
- PureFTP Log Format
- Raiden FTP Log Format
- Rumpus Log Format
- Serv-U FTP Log Format
- UNIX FTP Log Format
- War FTP Log Format
- War FTP Log Format (Alternate)
- WebSTAR FTP Log Format
- WS_FTP Log Format
- WU-FTP Log Format
- WU-FTP Log Format (yyyy-mm-dd Dates, Server Domain)

Firewall

- Applied Identity WELF Log Format
- Argus
- Astaro Log Format
- Barrier Group Log Format
- AscenLink Log Format (BETA)
- Cisco PIX/ASA/Router/Switch Log Format (BETA)

- FortiGate Log Format (BETA)
- IAS Comma-Separated Log Format (BETA)
- Interscan Web Security Suite (BETA)
- IPTables Config Log Format (BETA)
- Juniper Secure Access SSL VPN Log Format (BETA)
- Microsoft Windows Firewall Log Format (BETA)
- Netscreen SSL Gateway Log Format (BETA)
- Netscreen Web Client Export Log Format
- NetScreen Log Format (BETA)
- Symantec Security Gateways Log Format (SGS 2.0/3.0 & SEF 8.0) (BETA)
- Tipping Point IPS Log Format (BETA)
- Watchguard XML Log Format
- XWall Log Format (BETA)
- Borderware Log Format
- Check Point SNMP Log Format
- Cisco PIX/IOS Log Format
- Clavister Firewall Log Format
- Clavister Firewall Log Format (CSV)
- Clavister Firewall Syslog Log Format
- Coradiant Log Format (object tracking)
- Coradiant TrueSight Log Format (object tracking) v2.0
- Cyberguard WELF Log Format
- Firebox Log Format
- Firewall-1 (fw log export) Log Format
- Firewall-1 (fw logexport export) Log Format
- Firewall-1 (fw log -ftn export) Log Format
- Firewall-1 Log Viewer 4.1 Export Log Format
- Firewall-1 NG Log Format
- Firewall-1 Next Generation Full Log Format (text export)
- Firewall-1 Next Generation General Log Format (text export)
- Firewall-1 Text Export Log Format
- Firewall1 Webtrends Log Format
- Fortinet Log Format (syslog required)
- FortiGate Log Format
- FortiGate Comma Separated Log Format
- FortiGate Space Separated Log Format
- FortiGate Traffic Log Format
- Gauntlet Log Format
- Gauntlet Log Format (yyyy-mm-dd dates)
- GNAT Box Log Format (Syslog Required)
- GTA GBWare Log Format
- IAS Log Format
- IAS Alternate Log Format
- Ingate Firewall Log Format
- Instagate Log Format
- Interscan Web Security Suite
- ipfw Log Format
- IPTables Config Log Format
- Microsoft ISA WebProxy Log Format (CSV)
- Microsoft ISA Server Packet Logs
- Lucent Brick
- Microsoft ICF Log Format
- Microsoft ISA WebProxy Log Format (W3C)
- iPlanet/Netscape Log Format
- Netscreen IDP Log Format
- Neoteris/Netscreen SSL Web Client Export Log Format (BETA)
- Netwall Log Format
- NetScreen Log Format
- NetScreen Traffic Log Format (get log traffic)
- NetScreen Traffic Log Format
- Nokia IP350/Checkpoint NG (fw log export) Log Format
- Norton Personal Firewall 2003 Connection Log Format
- Novell Border Manager Log Format
- portsentry Log Format
- Rapid Firewall Log Format
- Raptor Log Format
- Raptor Log Format (Exception Reporting)
- SafeSquid Log Format (logging to syslog server)
- SafeSquid Standalone Log Format
- Symantec Gateway Security 400 Series Log Format
- Sharewall Log Format
- Sidewinder Log Format
- Sidewinder Firewall Log Format
- Sidewinder Raw Log Format (SUPPORTED ONLY AFTER acat -x EXPORT)
- Sidewinder Syslog Log Format

- SmoothWall Log Format
- SmoothWall SmoothGuardian 3.1 Log Format
- SonicWall or 3COM Firewall
- SonicWall 5
- Sonicwall TZ 170 Firewall
- Stonegate Log Format
- Symantec Enterprise Firewall Log Format
- Symantec Enterprise Firewall 8 Log Format
- Symantec Security Gateways Log Format (SGS 2.0/3.0 & SEF 8.0)
- Symantec Gateway Security Binary Log Format (SUPPORTED ONLY WITH TEXT EXPORT)
- Symantec Web Security Log Format
- Tiny Personal Firewall Log Format
- Watchguard Log Format
- Watchguard Firebox Export Log Format
- Watchguard Firebox Export Header
- Watchguard Firebox v60 Log Format
- Watchguard Firebox V60 Log Format
- Watchguard Historical Reports Export Log Format
- Watchguard SOHO Log Format
- Watchguard WELF Log Format
- Watchguard WSEP Text Exports Log Format (Firebox II & III & X)
- Webtrends Extended Log Format (Syslog)
- Webtrends Extended Log Format
- WELF Log Format (stand-alone; no syslog)
- WELF date/time extraction (no syslog header)
- Zone Alarm Log Format
- Zyxel Firewall Log Format
- Zyxel Firewall WELF Log Format

Application

- Active PDF Log Format
- Arcserve NT Log Format
- AutoAdmin Log Format (BETA)
- Backup Exec Log Format (BETA)
- MPS Log Format (BETA)
- Performance Monitor Log Format (BETA)
- BroadVision Error Log Format
- BroadVision Observation Log Format
- Cognos Powerplay Enterprise Server
- Cognos Ticket Server Log Format
- ColdFusion Application Log Format
- ColdFusion Application Log Format (CSV)
- Essbase Log Format
- Filemaker Log Format
- Filemaker 3 Log Format
- FusionBot Log Format
- Java Bean Application Server Log Format
- Microsoft SQL Profiler Export
- Microtech ImageMaker Error Log Format
- MicroTech ImageMaker Media Log Format
- Mod Gzip Log Format
- iPlanet/Netscape Directory Server Format
- NVDcms Log Format
- Oracle Listener Log Format
- Oracle Failed Login Attempts Log Format
- Plesk Server Administrator Web Log
- Policy Directory Audit Log Format
- Policy Directory Security Audit Trail Log Format
- PortalXPert Log Format
- Samba Server Log Format
- ShareWay IP Log Format
- SiteCAM Log Format
- SiteKiosk Log Format
- SiteKiosk 6 Log Format
- Software602 Log Format
- Symantec AntiVirus Corporate Edition
- Symantec AntiVirus Corporate Edition (VHIST Exporter)
- Tivoli Storage Manager TDP for SQL Server Format
- Vamsoft Open Relay Filter Enterprise Edition Log Format
- WebNibbler Log Format
- Wipro Websecure Audit Log Format
- Wipro Websecure Auth Log Format
- Wipro Websecure Auth (Alternate Dates)

- Wipro Websecure Debug Log Format

Sawmill automatically detects all of these formats, and arranges your profile options intelligently based on your log format. If your format is not on the list, but publicly available, support may be provided at no charge; and if so, it will be put into our log format queue. There is a continuous demand for new log file formats so there may be a significant delay before the plug-in is complete. For a faster response you may pay us to create the plug-in, which can be returned quickly. We can create one for you, just send a sample of your log data, 1 MB is ideal, but not more than 10 MB compressed to **support@flowerfire.com** and we will create a format description file. If your format is a private one, you can still process it with Sawmill by creating a custom log format description file (see **Creating Log Format Plug-ins (Custom Log Formats)**).

DOCUMENTATION

Getting Screen Dimensions and Depth Information

Custom Design your Web Pages for your Visitors

When analyzing the traffic to your web site, it is often useful to know the size of the monitors that your visitors are using. Knowing the screen dimensions, width and height as measured in pixels will allow you to design web pages that will fit on the monitors of those visiting you. Also knowing the screen depth, depth in bits; i.e. the number of available colors, will help you decide what colors to use in your images. This information is not generally available in web server log data, but you can add it by including the following small JavaScript program in one or more of your HTML pages:

```
<script language="JavaScript">
document.write('<img width="1" height="1" src="/log_analysis_screen_info.gif?' +
 'width=' + screen.width + '&height=' + screen.height + '&depth=' + screen.colorDepth + '">\n');
</script>
```

Just copy the script, paste it at the bottom of one of your web pages, and you're done. This script causes an extra entry to be included in the log data for each line processed. This entry, which appears as a hit on the file /log_analysis_screen_info.gif, is used automatically by Sawmill when analyzing web logs to compute the values of the derived "screen dimensions" and "screen depth" log fields, which are displayed in the statistics in the "Top screen dimensions" and "Top screen depths" views.

The image /log_analysis_screen_info.gif does not need to exist, but if it doesn't, you will see 404 errors (broken links) in your statistics for the file, and in some browsers you may see a small dot on your page where the JavaScript code appears. If you don't want to see these, you need to create an image file at the root of your site called /log_analysis_screen_info.gif. A blank or transparent image is a good choice. You can also put the image somewhere else on your site, and change the JavaScript to match, but the name of the image and its parameters must remain the same. However, it will actually slow your page down slightly if the image exists -- it is usually best *not* to create the image file.

Querying the SQL Database Directly

Sawmill can use a SQL database as its back-end database, storing the parsed data in that database, and querying the database to generate its reports. You can query that database directly too, from an external program or script. This chapter provides an introduction to the structure of the Sawmill SQL database to help you get started with your own queries.

The main table of the database is called `logfile`. This table contains one line per accepted database entry, and one column per database field. For instance, if you have 1 million lines of web log data, this table will have 1 million rows, with one column for each web log field (date/time, hostname, page, etc.).

Each value in the `logfile` table is a number. In the case of numerical fields (like hits, events, bandwidth, duration, etc.) the value will be the value of that field for the corresponding entry; for instance, a 1234-byte web log transfer would have 1234 in the bytes field.

The non-numerical fields of `logfile` are normalized, so for each field `fieldname`, there is a table called `fieldnameitemnum` (the field name, plus `itemnum`), which lists all the item numbers and item values for that field. The values in the `logfile` table correspond to the itemnums in the itemnums table. For instance, if "GIF" is itemnum 3 in the `file_typeitemnum` table, then a value of 3 in the `file_type` column of the `logfile` table indicates that the file type is "GIF".

By joining `logfile` to one or more itemnum tables, almost all standard queries can be performed. For instance, this query:

```
select count(page_views) as page_views, i.hostname
  from logfile l
  left join hostnameitemnum i on l.hostname = i.itemnum
  group by i.hostname
  order by page_views desc limit 10
```

generates a "top ten" report for the hostname field, showing page views per hostname, sorted descending. You can replace hostname with the internal name of any other field to get that top-ten report. You can change "limit 10" to "limit N" to show N rows, or remove it to show all rows. You can change page_views to the internal name of any other numerical database field to show that instead.

Any query can be executed from the mysql command line like this:

```
echo "query" | mysql -u username -h hostname -ppassword databasename > queryresult.tsv
```

This will run the query *query*, and generate the result of the query in the file queryresult.tsv.

The `fieldnamesubitem` tables describes the hierarchy of non-numerical fields by listing the direct descendents of each item. You might use this to get all months in a particular year, or all files and directories in a particular directory, or all cities in a particular state. Each row describes one item/subitem pair (using normalized numbers derived from the itemnum tables, as above). Item number 1 is the root (and the corresponding item is the empty string).

The `fieldnamebottomlevelitem` tables provide immediate mapping from an item to all its bottom-level items. For instance, you could use this to get all days in a particular year. Each row describes one item/bottomlevelitem pair (using normalized numbers derived from the itemnum tables, as above).

The `xref` tables provide fast access to common reports; they precompute the aggregated numerical values for field values. For instance, a cross-reference table for the file_type field would have one row for each file type, with the totals for that field; e.g., file_type=GIF, bytes=11235567. This table can be queried directly (much faster than querying `logfile`) to get the "top ten" reports for any field, or combination of fields, which have xref tables. When there are "unique" numerical fields; e.g., visitors, for web logs, they can be computed from the tables with names like `xref0visitors`; this table lists the unique IPs (visitor IPs) for each itemnum combination which appears in the corresponding `xref0` table.

Credits

Sawmill was created by Flowerfire, Inc.

Original inspiration for Sawmill came from **Kuck & Associates, Inc**, and from ***Seized by the Tale***, two sites which needed such a tool.

Thanks to the makers of **gd**, an excellent GIF creation library which is used to create all images displayed by Sawmill, including the pie charts, line graphs, table bars, legend boxes, and icons. gd was written by Thomas Boutell and is currently distributed by boutell.com, Inc. gd 1.2 is copyright 1994, 1995, Quest Protein Database Center, Cold Spring Harbor Labs.

Thanks to the makers of zlib, which Sawmill uses to process gzip and ZIP log data.

Thanks to Jason Simpson, Ken Brownfield, and Wayne Schroll for important feedback on the 1.0 alpha versions.

Thanks to Stephen Turner for his experienced input on the early 1.0 versions.

Thanks to the 1.0 beta testers for help on the beta versions, especially to Gary Parker, Glenn Little, and Phil Abercrombie.

Thanks to the 2.0 beta testers for help on the beta versions, especially to Gary Parker, Vincent Nonnenmach, and Glenn Little.

Thanks to all the 3.0 beta testers, especially Vincent Nonnenmach.

Thanks to all the 4.0 beta testers, especially (yet again) Vincent Nonnenmach, and many others.

Thanks to all the 5.0 beta testers, especially Fred Hicinbothem, Yuichiro Sugiura, and Peter Strunk.

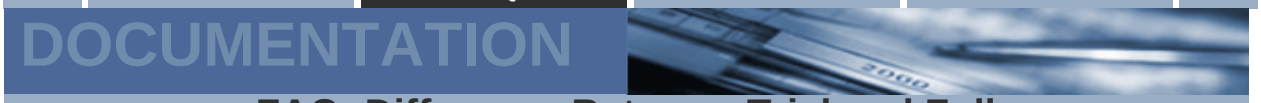
Thanks to all the 6.0 beta testers, especially Ed Kellerman, Noah Webster, Johnny Gisler, Morgan Small, Charlie Reitsma, James K. Hardy, Alexander Chang, Richard Keller, Glenn Little, Eric Luhrs, and Yann Debonne.

Thanks to all the 7.0 beta testers, too numerous to name here.

Thanks to all the 8.0 beta testers, who are helping us make it a great product.

Sawmill is a much better product thanks to the help of these and other beta testers.

Copyright



FAQ: Difference Between Trial and Full

Question: What's the difference between the full version of Sawmill and the Trial version?

Short Answer: The Trial version is identical to the full version, except that it expires after 30 days.

Long Answer

Sawmill Trial is a free trial version, intended to let you evaluate the program without having to buy it. It is identical to the full version, except that it expires 30 days after it is first used. After the trial period is over, the trial version will no longer work, but it can be unlocked by purchasing a license, and all settings, profiles, and databases will remain intact.

FAQ: Difference Between Enterprise and Professional

Question: What's the difference between Sawmill Enterprise and Sawmill Professional?

Short Answer: Enterprise supports MySQL, WebNibbler, multithreaded database builds, and full interface customization.

Long Answer

Sawmill Enterprise is intended for large organizations with very large datasets and advanced customization needs.

Sawmill Enterprise has all the features of Sawmill Professional, and the following additional features.

- **MySQL**. Support for MySQL as a back-end database. This allows the data collected by Sawmill to be queried externally, and provides much greater scalability through the use of multi-computer database clusters.
- **Multithreading**. Sawmill can use multiple processors to build a database, providing faster log processing than a single CPU can provide. With external scripting, it is also possible to yoke a cluster of computers into a log processing grid, splitting the dataset and building sections of the database on multiple systems, and merging them as a final step, for even higher scalability.
- **WebNibbler™**. WebNibbler™ provides advanced user tracking for web logs. Taking the user cookie concept to a new level, WebNibbler™ can categorize all events from a particular user, even if that user browses anonymously before logging on by name, or browses from multiple systems or is using multiple browsers.
- **Interface customization**. The web interface for Sawmill is written entirely in its internal language, called salang (somewhat similar to perl). With Enterprise licensing, these files can be edited, providing *complete* customization of the entire user interface, both administrative and non-administrative.

FAQ: Unlocking a Trial Installation

Question: When I purchase, do I have to download a new version of Sawmill, or can I "unlock" my existing trial installation?

Short Answer: You can unlock your trial installation by entering your license key in the Licensing page.

Long Answer

You don't have to download again. When you purchase, you get a license key by email. You can enter that key into the Licensing page (which you can get to by clicking Licensing on the Administrative menu) to unlock a trial installation, converting it into a fully licensed installation.

FAQ: Resetting the Trial Period

Question: My 30-day trial has expired, and I haven't finished evaluating Sawmill yet. How can I get a new trial?

Short Answer: Go to the Licensing page, delete your expired license, and click "Try Sawmill For 30 Days."

Long Answer

Sawmill's trial license allows you to use it for evaluation purposes only. However, if after 30 days you still have not had a chance to fully evaluate Sawmill, you can extend your trial for another 30 days by doing the following:

1. Go to the Licensing page.
2. Delete your current trial license.
3. Click the "Try Sawmill for 30 Days" button.

This will work only once -- after that, you will need to contact us at **support@flowerfire.com** if you want to extend your trial period further.

FAQ: Upgrading Without Losing Data

Question: How can I upgrade to a new version of Sawmill without losing my profiles, databases, and other data?

Short Answer: When upgrading from an older 7.x version to a newer 7.x version (except on Windows), start with the new LogAnalysisInfo and copy files as described in the long answer. On Windows simply install Sawmill over the existing installation. When upgrading from 6.x to 7, copy Configs and run from the command line with -a cc.

Long Answer

For Windows users:

Installing a newer 7.x version of Sawmill over your existing Sawmill 7 installation will not result in data loss. If you are currently using Sawmill on a Windows platform all that's required is downloading the Sawmill distribution package and then clicking install. The installer will simply install what's necessary and will not overwrite or remove your existing profiles, databases, or any user configuration data. Once the install is complete upgrading is complete, and you are now ready to continue using Sawmill.

If you're upgrading from version 6 to version 7, you can only transfer your configuration files -- databases cannot be transferred. Sawmill 7 stores much more information in its databases, and this information cannot be computed from a version 6 database. Copy the Configs folder from the old LogAnalysisInfo to the new and run Sawmill (SawmillCL.exe on Windows) from the command line with "-a cc" to convert the old configurations to new v7 profiles. Then build the databases from those profiles to process the log data and create v7 databases.

For non-Windows users:

Start by installing the new version, with its new LogAnalysisInfo and the new binary; e.g., the Sawmill application and Sawmill executable on MacOS, or the Sawmill executable on other platforms.

If you're upgrading from version 6 to version 7, you can only transfer your configuration files -- databases cannot be transferred. As mentioned above, Sawmill 7 stores much more information in its databases, and this information cannot be computed from a version 6 database. Copy the Configs folder from the old LogAnalysisInfo to the new and run Sawmill from the command line with "-a cc" to convert the old configurations to new v7 profiles. Then build the databases from those profiles to process the log data and create v7 databases.

If you're upgrading from an older 7.x to a newer 7.x, start with the new LogAnalysisInfo. In order to preserve profiles, settings, databases, and more, you need to copy them from the old LogAnalysisInfo folder. Here are the parts you may want to copy:

1. **Profiles**. Copy all files ***except*** default_profile.cfg, from your existing profiles folder in the LogAnalysisInfo folder, to the new profiles folder.
2. **Databases**. Copy folders from your existing LogAnalysisInfo folder to the new one.
3. **Schedules**. Copy the file schedules.cfg from your existing LogAnalysisInfo folder to the new one.
4. **Users**. Copy the file users.cfg from your existing LogAnalysisInfo folder to the new one.
5. **Licenses**. Copy the file licenses.cfg from your existing LogAnalysisInfo folder to the new one.

- **NOTE:** ***Default Profile***. Should not be copied! This file cannot be copied because the new file may have new options that the old one did not. If you have changed default_profile.cfg, and need to keep your changes, you will need to merge them by cut/pasting specific option changes over manually from the old file to the new file using a text editor.

In some cases, for simple upgrades, you can simply copy the entire LogAnalysisInfo folder to the new installation. But in general, there are changes to files in the LogAnalysisInfo folder from one version to the next, so you may not get the latest features and bug fixes, and Sawmill may not even work, if you use an older LogAnalysisInfo. Therefore, it is best to use the new LogAnalysisInfo, and copy just the pieces you need from the old one.

FAQ: Available Platforms

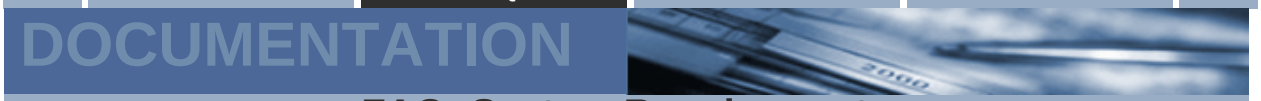
Question: What platforms does Sawmill run on?

Short Answer: Windows ME/NT/2000/XP/2003, MacOS, most versions and variants of UNIX.

Long Answer

Sawmill runs on Windows ME/NT/2000/XP/2003, MacOS X, and most popular flavors of UNIX (Linux, Solaris, FreeBSD, OpenBSD, NetBSD, BSD/OS, Tru64 UNIX (Digital Unix), IRIX, HP/UX, AIX, OS/2, and BeOS). Binary versions are available for the most popular platforms; on less common platforms, it may be necessary to build Sawmill yourself from the source code, which is available for download in encrypted/obfuscated format, and then build it with g++.

That's just the server; once you have the server running, you can configure Sawmill, generate reports, and browse reports from **any** computer, using a web browser.



FAQ: System Requirements

Question: How much memory, CPU power, and disk space do I need to run Sawmill?

Short Answer: At least 256 MB RAM, 1 GB preferred; 500 MB disk space for an average database; and as much CPU power as you can get.

Long Answer

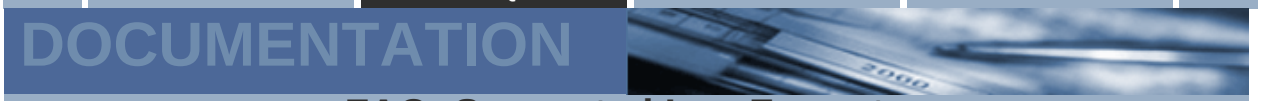
Sawmill is a heavy-duty number crunching program, and can use large amounts of memory, CPU, and disk. You have some control over how much it uses of each, but it still requires a reasonably powerful computer to operate properly.

Sawmill uses around 30 MB of memory when it processes a small to medium size log file, and it can use considerably more for very large log files. The main memory usage factors are the "item lists", which are tables containing all the values for a particular field. If you have a field in your data, which is very complex, and has many unique values (the URL query field for web log data is a common example of this), the item list can be very large, requiring hundreds of megabytes of memory. This memory is mapped to disk to minimize physical RAM usage, but still contributes to the total virtual memory usage by Sawmill. So for a database with very complex fields, large amounts of RAM will be required. For large datasets, it is possible for Sawmill to use more than 2 GB of address space, exceeding the capabilities of a 32-bit system; in this situation, it is necessary to use a 64-bit system, or a MySQL database, or both (see **Database Memory Usage** and **Sawmill uses too much memory for builds/updates, and is slow to view**). This typically will not occur with a dataset smaller than 10 GB, and if it often possible to process a much larger dataset on a 32-bit system with 2 GB. A dataset over 100 GB will often run across this issue, however, so a 64-bit system is recommended for very large datasets. If your system cannot support the RAM usage required by your dataset, you may need to use log filters to simplify the complex database fields.

The Sawmill installation itself takes less than 50 MB of disk space, but the database it creates can take much more. A small database may be only a couple megabytes, but if you process a large amount of log data, or turn on a lot of cross-references and ask for a lot of detail, there's no limit to how large the database can get. In general, the database will be somewhere on the order of 50% the size of the uncompressed log data in it, perhaps as much as 100% in some cases. So if you're processing 100 GB of log data, you should have 100 GB of disk space free on your reporting system to hold the database. If you use an external (e.g. SQL) database, the database information will take very little space on the reporting system, but will take a comparable amount of space on the database server.

Disk speed is something else to consider also when designing a system to run Sawmill. During log processing, Sawmill makes frequent use of the disk, and during statistics viewing it uses it even more. Many large memory buffers are mapped to disk, so a disk speed can have a very large impact on database performance, both for processing log data and querying the database. A fast disk will increase Sawmill's log processing time, and the responsiveness of the statistics. SCSI is better than IDE, and SCSI RAID is best of all.

During log processing, especially while building cross-reference tables, the CPU is usually the bottleneck -- Sawmill's number crunching takes more time than any other aspect of log processing, so the rest of the system ends up waiting on the CPU most of the time. This means that any improvement in CPU speed will result in a *direct* improvement in log processing speed. Sawmill can run on any system, but the more CPU power you can give it, the better. Large CPU caches also *significantly* boost Sawmill's performance, by a factor of 2x or 3x in some cases.



FAQ: Supported Log Formats

Question: What sorts of log files can Sawmill process?

Short Answer: Sawmill can handle all major log formats and many minor formats, and you can create your own custom formats.

Long Answer

Sawmill is not just for web server logs, though it's well suited to that task. Sawmill also supports firewall logs, proxy logs, mail logs, antivirus logs, network logs, FTP logs, and much more.

Click here for the full list of **Supported Log Formats**.

It automatically detects all the formats it supports, and chooses appropriate settings for the format.

We're continually adding new log formats, so the list above will keep growing. If the format is publicly available, support may be provided at no charge; and if so, it will be put into our log format queue. There is a continuous demand for new log file formats so there may be a significant delay before the plug-in is complete. For a faster response you may pay us to create the plug-in, which can be returned quickly. **support@flowerfire.com** for details.

If you want to analyze a log in a different format, Sawmill also lets you create your own format description file; once you've done that, your format becomes one of the supported ones--Sawmill will autodetect it and choose good options for it, just like any built-in format.

Sawmill's format description files are very flexible; almost any possible format can be described. If you have an unsupported format and you'd like help writing a format file, please contact **support@flowerfire.com**, and we'll write a format file for you, at no charge.

FAQ: Sawmill vs. The Competition

Question: How is Sawmill different from other log analysis tools?

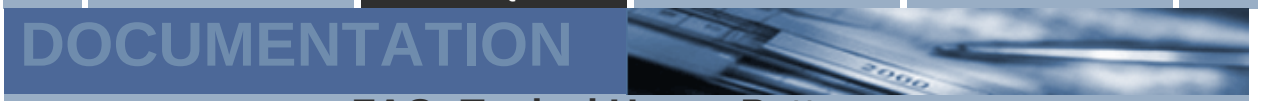
Short Answer: Among other things, Sawmill does not generate static reports -- it generates dynamic, interlined reports.

Long Answer

There are many areas in which Sawmill beats the competition, but one major one is that Sawmill's statistics are **dynamic**, and its statistics pages are **interlinked**. Most other log analysis programs are report-based -- you specify certain criteria; e.g., "give me all hits on my web site on January 14, broken down by page" and it generates a single report, and it's done. If you want more detail about something, it's not available, or it's only available if you reprocess the log data with different settings.

Sawmill generates HTML reports on the fly, and it supports zooming, filtering, and many other dynamic features. You can zoom in a certain directory, for instance, and then see the events for that directory broken down by date, by IP, by weekday, or in any other way you like. You can create arbitrary filters, for instance to zoom in on the events for a particular address on a particular day, or to see the search terms that were used from a particular search engine on a particular day, which found a particular page. Sawmill lets you navigate naturally and quickly through hierarchies like URLs, pages/directories, day/month/years, machine/subnets and others.

Of course, there are many other features that set Sawmill apart from the competition-- see our website for a complete list.



FAQ: Typical Usage Patterns

Question: How does a typical company use Sawmill; what does a typical Sawmill setup look like?

Short Answer: Installations vary from customer to customer--Sawmill provides enough flexibility to let you choose the model that works best for you.

Long Answer

There are quite a lot of different "models" that different customers use. For web server analysis, it is common to have Sawmill running on the active web server, either stand-alone or in web server mode, accessing the growing log files directly; this works well as long as the dataset is not too large and the server is not too heavily loaded. For very large datasets, however, many customers have dedicated Sawmill machines, which pull the logs over the network from the server(s). Databases are generally updated regularly; it's common to have them updated in the middle of the night, every night, using the Sawmill Scheduler or an external scheduler like cron.

In terms of the database layout, some common models include:

- **A single database**. Most customers use a single large database that contains all their data. This works well if you have a lot of disk space and a fast computer, or computers to process your logs with, or if your log data is not too large. You can use Sawmill's normal filtering features to zoom in on particular parts of the data, but it's all stored in a single database. Sawmill has other features that can be used to limit certain users to certain parts of the database; this is particularly useful for ISPs who want to store all their customers' statistics in a single large database, but only let each customer access their own statistics.

- **A "recent" database and a long-term database**. In cases where log data is fairly large (say, more than 10 Gigabytes), or where disk space and/or processing power is limited, some customers use two databases, one in detail for the recent data, updated and expired regularly to keep a moving 30-day data set, for instance, and the other less detailed for the long-term data, updated regularly but never expired. The two databases combined are much smaller than a single one would be because they use less overall information, so it takes less time to process the logs and to browse the database. This is often acceptable because fine detail is needed only for recent data.

- **A collection of specialized databases**. Some customers use a collection of databases, one for each section of their statistics. This is particularly useful for log data in the multi-Terabyte range; a tightly-focused database for instance, showing only hits on the past seven days on a particular directory of the site is *much* smaller and faster than a large all-encompassing database. This is also useful if several log files of different types are being analyzed for instance, an ISP might have one database to track bandwidth usage by its customers, another to track internal network traffic, another to track usage on its FTP site, and another to track hits on its own website.

There are a lot of options, and there's no single best solution. You can try out different methods, and change them if they're not working for you. Sawmill provides you the flexibility to choose whatever's best for you.

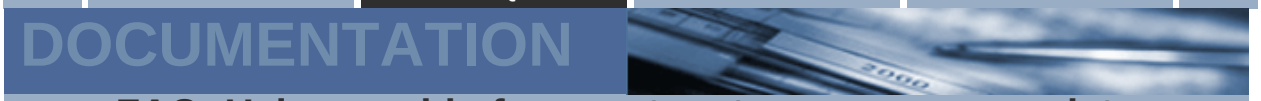
FAQ: Processing Large Log Files

Question: How large of a log file can Sawmill process?

Short Answer: There are no limits, except those imposed by the limitations of your server.

Long Answer

There is no fundamental limit -- given enough memory, disk space, and time, you can process the world. We've processed log files terabytes in size, billions of lines long, and been able to browse their statistics at **full** complexity in real time, with no troubles.



FAQ: Using a grid of computers to process more data

Question: How can I use a grid (cluster) of computers to process logs faster?

Short Answer: Use an internal database, build a separate database on each computer, and merge them.

Long Answer

Sawmill has several features which can be used to script this sort of approach. Specifically, if you are using the internal database, there is a "database merge" operation which can be performed from the command line, using "-a md". This merges two existing databases into a single database. Using this feature, it becomes possible to write a script to:

1) Split your dataset into 100 equal parts,

2) build 100 databases on 100 Sawmill installations on 100 computers and

3) run 99 merge operations in sequence on a single computer, e.g., "sawmill -p profilename -a md -mdd /database/from/installationN" to merge in the data from the Nth installation, to combine them all into a single database.

Since merge operations are generally much faster than log processing, this approach can be used to parallelize a huge log processing operation, greatly accelerating it.

DOCUMENTATION

FAQ: Log Entry Ordering

Question: Does the log data I feed to Sawmill need to be in chronological order?

Short Answer: No; your log entries can be in any order.

Long Answer

Unlike many other log analysis tools, Sawmill doesn't care what order your log data is in. It can be in any order you like. That means that if you have several log files from several different servers in a cluster, you can dump the data from all of them into the same database, without worrying that their date ranges overlap.

FAQ: What is a Log File?

Question: What is a log file?

Short Answer: Log files are text files created by your server, recording each hit on your site. Sawmill generates its statistics by analyzing log files.

Long Answer

Log files are large text files generated by web servers, proxy servers, ftp servers, and just about every other kind of server. Every time something happens on the server (it serves a file, or delivers a message, or someone logs in, or something else), the server logs that information to the file, which continues to grow as new events occur. Log files are not particularly human-readable, and do not generally contain summarizing information, which is why Sawmill exists -- Sawmill processes your log files, summarizes them and analyzes them in many ways, and reports it back to you in a much friendlier format-- graphs, tables, etc.

You need to have access to your log files to use Sawmill. If you don't have log files, Sawmill can't do anything for you. If you don't know where your log files are, ask your server administrator (hint: they are often stored in a directory called "logs"). In some cases, servers are configured so they do not keep log files, or the logs are hidden from users; in these situations, you will not be able to use Sawmill. Again, your server administrator can help you find your log files, or they can tell you why they're not available. If you're trying to analyze a web site, and your ISP does not provide logs for you, you may want to consider switching to one that does.

FAQ: Scheduling

Question: Can Sawmill be configured to automatically analyze the access log for my site on a shared server once a day at a given time?

Short Answer: Yes, if you run it stand-alone, or if your server has a scheduling program.

Long Answer

It depends on your web server. If you run Sawmill as a stand-alone program (rather than as a CGI program) on your server, then you can use Sawmill's built-in Scheduler to do this. If you can't run it stand-alone or don't want to, then you can still set up automatic database builds if your server has its own scheduling program (like cron or Windows Scheduler).

FAQ: Running on a Different IP

Question: I'm running Sawmill on Windows, and it automatically starts itself up on IP 127.0.0.1 and port 8987. How can I tell it to use another IP address and port?

Short Answer: Set the Server Hostname option and the Web Server Port option in the Network section of the Preferences.

Long Answer

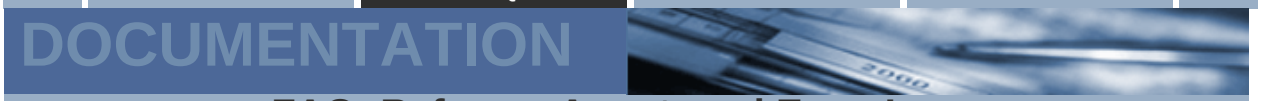
By default, Sawmill binds to all available IPs, so if there's an IP address where it is allowed to listen on port 8987, it already is, it's also listening on 127.0.0.1.

If you want it to listen *only* on the IP you specify you can do it from the Preferences. Go to the Preferences, click on the Network category, change the "Server hostname" option to the IP address you want to use, and change the "Web server port" option to the port number you want to use. The next time you start Sawmill, it will automatically bind to the IP address you specified.

If you're using the command-line version of Sawmill (sawmill), you can either do the same as above, or you can give Sawmill command line options to tell it which IP number and port to use:

```
sawmill -ws t -sh 128.129.130.131 -wsp 8888
```

When you use these options, Sawmill will immediately start up its web server on the port you specify.



FAQ: Referrer, Agent, and Error Logs

Question: How do I see referrer (referring URL, search engines, and search terms), agent (browser and OS), or error statistics?

Short Answer: Use "extended" or "combined" log format to see referrer and agent information, or analyze the log files with a separate profile. For error logs, analyze them with a separate profile.

Long Answer

Different log formats contain different types of information. All major web log formats include page, date/time, and browsing host information, but not all contain referrer and agent information. If your log format does not include referrer or agent information, Sawmill will not include that information in its database. The easiest way to get referrer or agent information is to change your web server's log format to an "extended," or "combined" format, which includes referrer and agent information; then Sawmill will automatically include referrer and agent information in the database and in the statistics.

If it's not possible to change your log format, and you have a separate referrer log (often called `referer_log`), then you can analyze that log directly with Sawmill. Just point Sawmill at the log, and Sawmill should recognize it as a referrer log. Sawmill will show statistics with referrer and page information. Host and date/time information are not available in a standard referrer log, so referrer and page is all Sawmill can extract. By using an extended or combined log format, you will be able to do more powerful queries, for instance to determine the referrers in the most recent week.

Similarly, if you can't configure your server to use extended or combined, but you have a separate agent log, you can analyze it with Sawmill by creating a separate profile that analyzes the agent (web browser and operating system) information only. Since an agent log contains only agent information, you won't be able to cross-reference the agent information with page, date/time, host, referrer, or anything else; to do that, you'll need an extended or combined format log.

To analyze error information, you'll need an error log (often called `error_log`). Just point Sawmill at your error log when you create the profile. Since the error log contains only error messages, you won't be able to cross-reference the errors against page, date/time, or any other fields; if you need cross-referencing of errors, you may be able to get what you need by cross-referencing the "server response" field of your normal web log to the fields you need cross-referenced; then apply "404" as a filter on the server response field and you'll see only those web site hits that generated 404 (file not found) errors.

FAQ: Language Modules--Localization and Customization

Question: Is Sawmill available in languages other than English? How can I change the output of Sawmill to be in a different language, or to use different wording?

Short Answer: Sawmill is currently available in English, German, and Japanese, and can be translated into any language fairly easily. Customization of output text is also easy.

Long Answer

Sawmill has a feature designed for just this purpose, called Language Modules. Language modules are text files which contain all of the text that Sawmill ever generates. You can translate part or all of Sawmill into any language by modifying the language modules. English, German, and Japanese translations already exist. Language modules can also be used to customize the output of Sawmill in almost any conceivable way. For full details, see **Language Modules--Localization and Text Customization** in the online manual.

FAQ: Running Sawmill at System Startup

Question: Can I set up Sawmill to start automatically when the computer starts up?

Short Answer: Yes; run it as a Service on Windows; use StartupItems under MacOS X; use the /etc/rc.d mechanism on UNIX systems that support it.

Long Answer

Sawmill can be configured to run at startup in the same way any other program can, and the exact method depends on your operating system. Here's how:

On Windows:

Sawmill is automatically installed as a Service, and will be running as soon as installation is complete. The Service is set to automatically start when the system starts up. You can edit Service parameters, for instance to have it run as a different user, or to have it start manually, in the Services control panel.

On MacOS X:

1. Install Sawmill in its default location at /Applications/Sawmill.
2. If the folder /Library/StartupItems does not exist, create it.
3. Copy the Sawmill folder from /Applications/Sawmill/Startup to /Library/StartupItems.

On RedHat 9 Linux or later:

Do the following as root:

1. Move the `sawmilld` file from the Extras/RH9 directory of your Sawmill installation, to /etc/rd.d/init.d. Type

   ```
   chkconfig --add sawmilld
   ```

   ```
   chkconfig --level 2345 sawmilld on
   ```

   to install it and turn it on.

2. Rename the Sawmill executable to `sawmill` (or change the name of the executable in the script) and put it in /etc/sawmill.
3. Put a symbolic link to LogAnalysisInfo in /etc/sawmill/LogAnalysisInfo (or you can put the actual directory there), using the `ln -s` command. You'll need to create the directory /etc/sawmill first. For example:

   ```
   ln -s/usr/home/sawmill/LogAnalysisInfo/etc/sawmill/LogAnalysisInfo
   ```

On Other Linux or Other UNIX-type Operating Systems:

1. Install a script to start Sawmill in /etc/rc.d or /etc/init.d, or however your UNIX variant does it. A sample script, based on the Apache script, is available **here**. The method varies from UNIX to UNIX, but to give one specific example, in RedHat Linux 9.0 you should call the script sawmilld and put it in /etc/rc.d/init.d, and then make symbolic links to it from the rc0.d - rc6.d directories, encoding into the name of the link whether to Start SawmillCL.exe at that runlevel, or to kill it. A good sequence of links is the following:

   ```
   ln -s /etc/rc.d/init.d/sawmilld /etc/rc.d/rc0.d/K15sawmilld
   ln -s /etc/rc.d/init.d/sawmilld /etc/rc.d/rc1.d/K15sawmilld
   ln -s /etc/rc.d/init.d/sawmilld /etc/rc.d/rc2.d/K15sawmilld
   ```

```
ln -s /etc/rc.d/init.d/sawmilld /etc/rc.d/rc3.d/S85sawmilld
ln -s /etc/rc.d/init.d/sawmilld /etc/rc.d/rc4.d/S85sawmilld
ln -s /etc/rc.d/init.d/sawmilld /etc/rc.d/rc5.d/S85sawmilld
ln -s /etc/rc.d/init.d/sawmilld /etc/rc.d/rc6.d/K15sawmilld
```

If you're not sure where to put the Sawmill links or what to call them, and you have Apache installed on your system, look for files with names containing `httpd` in /etc/rc.d or /etc/init.d, and use the same names and locations for Sawmill, replacing `httpd` with `sawmilld`.

2. Rename the Sawmill executable to `sawmilld` (or change the name of the executable in the script) and put it in /bin or somewhere else in your default path.

3. Put a symbolic link to LogAnalysisInfo in /etc/SawmillCL.exe/LogAnalysisInfo (or you can put the actual directory there), using the `ln -s` command (you'll need to create the directory /etc/SawmillCL.exe first), e.g.:

   ```
   ln -s /usr/home/SawmillCL.exe/LogAnalysisInfo /etc/SawmillCL.exe/LogAnalysisInfo
   ```

FAQ: Running Sawmill in the Background

Question: When I run Sawmill in a UNIX terminal window, and then close the window, Sawmill stops working. What can I do about that?

Short Answer: Add an ampersand (&) to the end of the command line to run it in the background.

Long Answer

When you run Sawmill from the command line in UNIX by just typing the name of the program, it runs in the foreground. That means that you don't get your prompt back until Sawmill exits, and it also means that if you close your terminal window, the Sawmill server will terminate and you will not be able to use it anymore until you open another terminal window and restart Sawmill. Often, that's not what you want--you want Sawmill to keep running after you close the window. You can do that by running Sawmill in the background.

To run Sawmill (or any other UNIX program) in the background, add an ampersand (a & character) to the end of the command line; for instance, you might use the following command line:

```
./SawmillCL.exe &
```

if the name of your Sawmill program is `SawmillCL.exe`. When you type this, you will see one line of output as Sawmill is backgrounded, and a few lines from Sawmill describing the running web server, and then you will have your shell prompt back, so you can type more commands. At this point, Sawmill is running in the background. You can type `exit` at the prompt to close the shell, or you can just close the window, and Sawmill will continue to run in the background.

On some rare occasions, Sawmill may generate output to the shell console. This is not usually a problem, but on some systems, background programs that generate output may be suspended, and that can make Sawmill inaccessible. To prevent this from happening, you may want to use this command line instead:

```
nohup ./SawmillCL.exe &
```

The "nohup" part of the command line stands for "no hang-up" and prevents this sort of output-related suspension problem. Unfortunately `nohup` doesn't exist on all systems. If you don't know if your system supports `nohup`, try including `nohup` on the command line--if it doesn't run that way, don't use it.

You can see current background jobs started from the current terminal using the `jobs` command (with most shells). You can terminate a background job by bringing it to the front using the `fg` command and then using control-C, or using the `kill` command together with its process ID. You can find the process ID (pid) of any background process, including ones started in other windows, using the `ps` command. For more information about any of these commands, use the `man` command (e.g. type `man ps`), or consult your UNIX documentation.

FAQ: Relocating LogAnalysisInfo

Question: How can I move the LogAnalysisInfo folder somewhere else?

Short Answer: Install Sawmill somewhere else, or make a symbolic link to LogAnalysisInfo, or put the pathname of the new location in the file LogAnalysisInfoDirLoc

---

Long Answer

Sawmill stores most of its data, including all internal databases, in a folder called LogAnalysisInfo. By default, this is in the same folder as the Sawmill binary. If you want it to be somewhere else, there are several options:

- **Relocate the databases.** Instead of relocating the whole LogAnalysisInfo folder, relocate only the databases, by changing the database folder location in the Database section of the Config section of the profile. Since most of the disk usage of the LogAnalysisInfo folder is due to databases, this provides most of the benefit of relocating LogAnalysisInfo, if disk usage is the issue.
- **Create a symbolic link (non-Windows only).** This works only on non-Windows systems. If Sawmill is installed in a directory called /some/dir/sawmill, and you want LogAnalysisInfo to be at /some/other/dir/LogAnalysisInfo, you can do this from the command line:

  ```
  mv /some/dir/sawmill/LogAnalysisInfo /some/other/dir
  ln -s /some/other/dir/LogAnalysisInfo /some/dir
  ```

  This creates a symbolic link from the installation location to the new location of LogAnalysisInfo, which Sawmill will automatically follow. By the way, you can also just move LogAnalysisInfo to /var/sawmill/LogAnalysisInfo, and Sawmill will look for it there.
- **Create a file LogAnalysisInfoDirLoc (for Windows).**This is a text file containing the location of the LogAnalysisInfo folder. This is most useful on Windows; on other platforms you can use a symbolic link, as described above. For instance, if you installed Sawmill in C:\Program Files\Sawmill 7, and want LogAnalysisInfo to be at E:\LogAnalysisInfo, you can move it from C:\Program Files\Sawmill 7\LogAnalysisInfo to the E: drive, and then create a text file (with Notepad) called LogAnalysisInfoDirLoc, **no .txt extension**, and type E:\LogAnalysisInfo as the contents of that text file. If Sawmill does not find a LogAnalysisInfo folder in the folder where it is running, it will look for that file, LogAnalysisInfoDirLoc, and it will look for LogAnalysisInfo in the location specified by the contents of that file.

FAQ: Using the Scheduler with CGI Mode

Question: How can I run Sawmill in CGI mode, and still use the Sawmill Scheduler?

Short Answer: Use an external Scheduler to run jobs or to call the Sawmill Scheduler, or run Sawmill in both CGI and web server modes.

Long Answer

Sawmill's built-in scheduler can only run scheduled jobs if Sawmill is actually running when the job's time comes. That's fine if you're running Sawmill in web server mode, where it runs all the time. But in CGI mode, Sawmill only runs when someone is actively using it, so scheduled tasks will not run. There are three main solutions to this problem: use an external scheduler to call Sawmill's scheduler, use an external scheduler to run the jobs directly, or run Sawmill on *both* CGI and web server modes, with the CGI mode doing everything but the scheduled jobs, and web server mode handling those.

UNIX

On UNIX, the most common scheduler is cron. You can set up cron to call Sawmill's scheduler by running the command (as root)

```
crontab -e
```

from the UNIX command line, and then adding

```
* * * * * sudo -u apache /full/path/to/SawmillCL.exe -scheduler
```

to the resulting file. You will need to replace "apache" with the name of your CGI user, and you will need to replace `/full/path/to/SawmillCL.exe` with the full pathname of your Sawmill executable. This tells cron tab to run Sawmill every minute, as the CGI user, with the `-scheduler` option, which tells Sawmill to run any scheduled jobs, and exit .

Another option is to run your Sawmill database builds and other jobs directly with cron; for instance you could add a line like this:

```
0 0 * * * sudo -u apache /full/path/to/SawmillCL.exe -rfcf configname -cm ud
```

(replace "apache" with the name of your CGI user) to update the profile specified by *configname* every night at midnight (the first number is the minute of the hour when the job should be run; the second number is the hour when the job should be run, and * * * means to run it every day).

Yet another option is to run Sawmill in web server mode as well as CGI mode, with the web server mode instance running only for the purpose of running jobs. The two will not interfere with each other; just start Sawmill from the command line using

```
/full/path/to/SawmillCL.exe &
```

and it will continue to run until the next reboot. If you want Sawmill to automatically restart itself at system startup, see **Running Sawmill at System Startup**.

Windows

Unfortunately, Windows Scheduler does not let you run jobs every minute (like UNIX cron does), so you cannot use it to call the Sawmill Scheduler directly. However, other options are available. You can use the Windows Scheduler to run your Sawmill jobs directly. For instance, to set Sawmill to update the database for a particular profile every night, do this:

- Open the Scheduled Tasks control panel.
- Double-click "Add Scheduled Task" and click Next.

- Choose "Sawmill (CGI)" from the list. If it does not appear, "Browse" and locate the {PRODUCT_NAME}CL.exe file, which is usually at c:\Program Files$PRODUCT_NAME 7${PRODUCT_NAME}CL.exe. Then click "Next".
- Click "Daily" and click "Next".
- Choose a time to run the build; sometime in the middle of the night (like midnight) is a good choice, and click "Next".
- Enter your username and password, and click Next.
- Click "Open advanced properties for this task when I click Finish," and click "Next".
- Add "-p *profilename* -a ud" to the end of the Run field, and click "OK".

Now Windows Scheduler is configured to update your database automatically every day.

Another option is to run Sawmill in web server mode as well as CGI mode, with the web server mode instance running only for the purpose of running jobs. The two will not interfere with each other; just start Sawmill by double-clicking its icon (you can also configure it to start whenever your computer restarts, using Windows Scheduler), and scheduled jobs will run as long as Sawmill is running. If you need Sawmill to be running while you are logged out, see **Running Sawmill as a Service**.

FAQ: Downloading Log Data by FTP

Question: Can Sawmill be configured to automatically FTP log files from multiple servers, and add them daily to a database?

Short Answer: Yes.

Long Answer

Yes; just select one of the FTP log sources when Sawmill asks you where your data is. Sawmill can FTP one or more log files from any FTP server, anonymously or with a username/password.

FAQ: Using a Command-line Log Source

Question: Can Sawmill use scp, or sftp, or ssh, or https, to download log data? Can I download a whole directory of files via HTTP? Can it uncompress tar, or arc, or sea, or hqx, etc.?

Short Answer: Not directly, but you can do it by using a command-line log source to run a command line, script, or program that does whatever is necessary to fetch the data, and prints it to Sawmill.

Long Answer

Sawmill supports many different methods of acquiring log data, including direct access to local files, and FTP or HTTP access to remote files; it can also decompress the major compression formats on the fly, including zip, gzip, and bzip2. If you need to use a different method to fetch the log data, like scp, sftp, or ssh, or if you need to read the log data from a database, or if you need to uncompress, decode, or decrypt a format that is not directly supported by Sawmill, you can do it by using a command-line log source.

Command-line log sources are very simple in concept. You give Sawmill a command line; it runs the command line whenever it needs to get the log data; the command, script or program you specify "prints: the log data generates it to stdout (standard command line output stream) and Sawmill reads the output of the command to get the log data. This provides you with unlimited flexibility in how you feed your data to Sawmill.

For instance, suppose Sawmill didn't support gzip for that (it does). Then you could use the following (UNIX) command log source: `/bin/gunzip -c /logs/mylog.gz`. Since the -c flag tells gunzip to write the output to stdout, Sawmill will read the log data directly from this command, without needing to use its built-in gunzipper. More usefully, any decompression utility with a similar flag can be used to allow Sawmill to read any compressed, archived, or encrypted log directly, even if it doesn't know anything about the format.

Even if you don't have a program that will dump the data to stdout, you can still use this approach by writing a tiny script. Consider the following (UNIX) shell script which scp'd files from a remote server and feeds them to Sawmill:

```
scp user@host:/logs/mylog.txt /tmp/templog
cat /tmp/templog
rm /tmp/templog
```

This script copies a log file from a remote machine (securely, using scp), prints it to stdout using "cat", and deletes it when it's done. The same script with slight modifications, could copy multiple files, or use a different method than scp to fetch the files (like sftp).

A simpler (and better) example which does the same thing is this command:

```
scp -qC user@host:/logs/mylog.txt > /dev/stdout
```

This explicitly scps the files to stdout, which sends them straight into Sawmill without the intermediate step of being stored on the disk or deleted. Since it's just one line, there's no need to use a script at all; this single line can be the command for the log source.

The HTTP client in Sawmill does not support "spidering"; it cannot follow links, so you can't point it at a page and have it download all log data referred from that page. You can use a command line log source to pull down the data into Sawmill, using a third-party command-line spider. One excellent choice is the free command-line spider wget, which is included with many operating systems, including most Linux and cygwin, and is available for download from most others. wget can write multiple spidered files directly to standard output, so it is necessary to write a small script which wgets the files locally, and then dumps them to standard output (this also allows incremental downloading of only changed files), like this:

```
wget -N -r -nH --cut-dirs=2 -q -A gz -np -P /var/mylogs http://somewhere/my/logs/
gunzip -c /var/stats/mylogs/*.gz
```

This is a very specific example, which you will need to change for your own purposes. wget has many, many options in addition to the ones used here; at a mimimum, you will want to use the -r option. The wget options on this example means:

- -N: use timestamping to download only new files.
- -r: Recursively follow links.
- -nH: Don't include the URL hostname in the directory name.
- --cut-dirs=2: chop off the top two levels of directories.
- -q: don't generate console output.
- -A gz: only download gz files.
- -np: don't follow links to the parent directory.
- -P /var/stats/mylogs: save the result in /var/stasts/mylogs.

DOCUMENTATION

FAQ: Running Sawmill as a Service

Question: Can I run Sawmill as a Service on Windows? Can I run Sawmill while I'm logged out?

Short Answer: As of version 7, Sawmill is installed as a service when you run the normal installer.

Long Answer

Earlier versions of Sawmill required extra steps to run them as a service, but this is no longer a problem-- the normal Windows installer automatically installs Sawmill as a service when you run it.

FAQ: Remote Administration

Question: My web site is hosted in another state. Does Sawmill provide browser based admin tools I can use to configure it and retrieve reports?

Short Answer: Yes, Sawmill's interface is entirely browser based.

Long Answer

Sawmill's interface is entirely web browser based. Sawmill runs either as a stand-alone program, in which case it uses its own built-in web server to serve its interface, or as a CGI program, in which case it uses the normal web server on the machine. In either case, Sawmill is configured by running a web browser on any machine you choose, and accessing Sawmill as though it were a website. Statistics are also served through a web browser interface. You do not need be physically present at the server to configure it or to view reports; all you need is a web browser. For more information about remote access see **Can't Access the Server**

FAQ: Statistics for Multiple Sites

Question: Can Sawmill generate separate analyses for all the websites hosted on my server?

Short Answer: Yes, Sawmill includes a number of features for just this purpose.

Long Answer

Absolutely. This is one of our core design goals -- to make Sawmill a good choice for web hosting providers, ISPs, and others who serve multiple sites from a single server. Sawmill's profiles provide an excellent mechanism for generating different statistics for each customer or website. If each site has its own log file(s), this is trivial; you can just make a profile that analyzes the appropriate log file. If all sites share a single log file, it's not much harder -- Sawmill's advanced filtering mechanism lets you easily ignore all log entries except those of interest to a particular website.

The technique you use depends on your situation. In general, you will need to have a separate profile for each user, you can quickly create all of your profiles using the Create/Update Many Profiles feature. For maximum flexibility, each profile can have its own database, and each profile can be password-protected or secured in some other way, to prevent unauthorized users from accessing it. See **Security** for a discussion of some of the ways profiles can be secured. If each profile has its own database, then the log filters can be used to filter out all statistics except those belonging to the user.

If you don't care if users can access each others' statistics, you can use a single profile with a single database, and give each user a bookmark URL pointing to their statistics in the database; this is the simplest approach, but it makes it possible for one user to see another's statistics, which is usually undesirable.

Advantages of using a single database:

- **Faster log processing -- log data is read only once.** This is particularly important when using an FTP log source with a log file containing the data for all profiles, because the log data will be fetched once per profile, so if you have 1000 profiles, this will use 1000 times more bandwidth. For local log files, this is not much of an issue, because Sawmill skips quickly over log entries it doesn't need, so it will only be spending real time on each log entry once.

Advantages of using multiple databases:

- **Smaller databases.** Though Sawmill has to create many databases instead of one, generally the total disk usage will be smaller, because each database is tightly focused on its site, and does not need to keep around information that applies only to other sites. In one real-world example, the total database size shrunk by a factor of 200 when the customer switched from one database to many.
- **Faster statistics browsing.** A small database is generally faster to browse than a large database, so using multiple small databases will make the statistics faster.
- **More flexibility.** Each profile can be configured separately, so you can have different cross-references, filters, database fields, etc. for different profiles. Using a single database locks you into a single database structure for all profiles.

In summary, you'll usually want to use multiple databases for multiple servers or sites. The main situation you'd want to use a single database for is if you're using FTP over a metered line to fetch the data; a single database will fetch it just once. Even then, though, you could set up an external script to fetch the log data to the local disk once, and then process it locally with Sawmill.

FAQ: Processing zipped, gzipped, or bzipped Log Data

Question: Can Sawmill process ZIPped, gzipped, or bzipped log data?

Short Answer: Yes, all three.

Long Answer

Any files that end with a .gz, .zip, .bz, or .bz2 will be treated as compressed files by Sawmill. It will uncompress them "on the fly" (not modifying the original file and not creating any new files), and process their uncompressed data the same way it reads normal log files.

FAQ: Clustered Servers

Question: Can Sawmill combine the logs from multiple clustered or load balanced web servers, so that the user has one view of the data? Can it report separately on the different servers?

Short Answer: Yes.

Long Answer

Sawmill can read any number of log files, from any number of servers, into a single database to show a single aggregate set of reports of all the data. If the logs also contain information about which server handled each request (or if each server has a separate log file, or a set of separate log files), then Sawmill can also show per-server statistics, if desired. Unlike many log analysis tools, Sawmill does not care if the files are in order, or if their date ranges overlap -- any combinations of any number of files with data in any order are possible.

To see per-server statistics, look in the reports for a report which breaks down the overall events by server. This might be called "Server domains" or "Server hosts" or "Server IPs" or something else, depending on the log data. Click on a particular server in that report; that zooms you in on that server. Now choose any other report from the "Default report on zoom" dropdown menu, to see a breakdown of the statistics for that server only. Alternatively, you can use the global filters to zoom "permanently" on a particular server, and then all reports will automatically show numbers for that server only.

If you don't have a field that tracks the server, you may still be able to get per-server statistics, by using the current_log_pathname() function detect which server each hit came from. You'll need to create a custom field in that case, with a log field to track the server, a filter to compute the field from the log pathname, and a database field and report for the field. For information on creating custom fields, see **Creating Custom Fields**.

FAQ: Protecting Clients' Statistics

Question: Can Sawmill be configured to limit access to statistics, so that a customer can only see the statistics associated with their section of my website?

Short Answer: Yes, you can password protect statistics in several ways.

Long Answer

Yes. Sawmill provides several ways to do this. In general, you will create a separate user for each client, and a separate profile for each client. Then you will configure their user to be non-administrative, and to have permission to access only their own profile. Finally, you will set up their profile to show only their data, either by pointing it only at their files, or if their data is interleaved with other clients' data, by using log filters to discard all events from the log which don't belong to them.

FAQ: Relabeling/White-labeling Sawmill

Question: I want to deploy Sawmill to my customers, but I want it to look like part of my site. I don't want the name Sawmill to appear -- I want my own name to appear. Can I relabel or white-label Sawmill?

Short Answer: Yes, but the degree to which you can relabel depends on your license.

Long Answer

You can relabel Sawmill and it's not very difficult, however the extent to which you can relabel depends on the license purchased (i.e. Professional, Enterprise etc.).

Sawmill Professional allows easy modification of certain screen attributes within the standard End User License, attributes such as colors, fonts, etc. Sawmill Professional also allows the language used on-screen to be modified or translated, plus it allows the addition of a graphic item by use of the custom HTML headers and footers, however the license does not allow the removal or replacement of any Sawmill logos or credits, etc. Should you require Professional Edition with even more customization ability and you are a qualifying user or integrator then we may be able to assist you. Under these circumstances you should forward your detailed proposal to us containing precise descriptions (preferably diagrams) of how you would wish the screens to look and we will respond to that proposal.

Sawmill Enterprise allows very considerable customization of the user interface and statistics screens to the point that just about every on-screen item can be modified deleted or replaced, or new items introduced. This ability is allowed within the standard license which should be consulted prior to making any final changes.

You can view the Sawmill End User License **here**. Note that under all circumstances, and for each product, the License requires that you leave the Flowerfire copyright notice untouched and visible together with a visible reference to Sawmill on every page.

Please contact **support@flowerfire.com** if our standard licensing does not fit your needs.

FAQ: Regular Expression Features

Question: What features can I use in Sawmill's regular expressions?

Short Answer: You can use whatever's documented (**Regular Expressions**), and possibly more. How much more you can use depends on your platform.

Long Answer

Regular expressions are not fully standardized -- different programs that support "regular expression" may support slightly different features. For instance, some will let you use {N} to repeat the preceding expression N times, and some will not (they will require you to write the expression N times yourself). Some will let you use \d to match any digit, and others will not, they will require you to use [0-9]. So the point of this question is, which of these "non-standard" features does Sawmill support? The answer depends on the platform you're running Sawmill on.

Sawmill's regular expressions vary depending on platform -- it uses the built-in regular expression library on some platforms, and the Boost library (modern C++ libraries) on others. Anything that is documented in **Regular Expressions** is available on all platforms. Anything that is not documented there may not be available. The easiest way to find out if something is available is to try it -- add a regular-expression filter to your Log Filters and see if it works. But if you want to make sure your profile is portable, and will work on other platforms, you should stick to the documented choices.

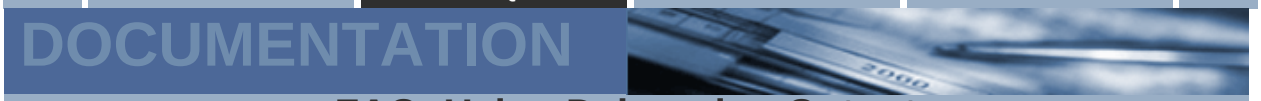
FAQ: Regular Expression Case-sensitivity

Question: Are Sawmill's regular expressions case-sensitive?

Short Answer: Yes.

Long Answer

Yes -- the regular expression `Dog` matches `Dog`, but not `dog` or `DOG`. If you need to match case-insensitively in a log filter, you can convert the field to lowercase first. Copy it to another temporary field if you don't want to modify the original, or you can explicitly list upper and lowercase values for every letter, e.g., [Dd][Oo][Gg] matches "dog" case-insensitively.



FAQ: Using Debugging Output

Question: How can I debug my custom log format, or my log filters?

Short Answer: Build the database from the command line with the -v option: `SawmillCL.exe -p` *`profilename`* `-a bd -v egblpfd`.

Long Answer

Custom log formats and log filters can be difficult to debug from the graphical interface, because there is little feedback about what Sawmill is doing as it processes the log. Fortunately, Sawmill has a powerful feature called "debugging output" that makes debugging custom log formats and filters much easier.

To see the debugging output, you need to use a command-line version of Sawmill . On Windows, that means using the SawmillCL.exe program, and running it from the command prompt. On Unix, you can use the normal Sawmill executable, since it works on the command line. On MacOS, you need to use the MacOS X command-line version of Sawmill.

Using the command shell, go to the Sawmill installation directory, using the "cd" command. Then rebuild the database like this:

```
sawmill -p profilename -a bd -v egblpfd | more
```

This command rebuilds the database for the *profilename* profile, and `-v egblpfd` tells Sawmill to report a great deal of information about what it's doing. Other -v options are available, but egblpfd are the seven options which are most useful for debugging profiles and filters. The results are piped through the "more" program, so you can page through the output using the space bar. Lines starting with "Processing line" show when Sawmill is processing a new log line. Lines starting with "Marking hits" show the end results that are being put into the database. Other lines provide information about log parsing and filtering that can be very useful when you're trying to debug a problem in the parsing of your custom format, or in your custom log filter.

FAQ: Excluding an IP Address or Domain

Question: How can I exclude hits from my own IP address, or from my organization's domain?

Short Answer: Add a Log Filter to exclude those hits.

Long Answer

One way to do this is to use a global filter in the statistics, and use "!(hostname within '123.124.125.126')", and this is often the first thing people try, but it's not the best choice. The speed of a statistics filter depends on the number of items checked, so if there are 100,000 IP addresses in your log file, and you check all 100,000, then Sawmill will take up to 100,000 times longer to generate each page. That is probably not what you had in mind. A much better option is to use the Log Filters.

Log filters are used to filter out or modify log data as it is being read (rather than filtering database data as it is being browsed, like the statistics filters). You can get to the Log Filters by clicking Show Config in the profiles list, and clicking the Log Filters category.

You want to create a filter that will reject any log entries whose hostname field is your IP address. If your IP address is 128.128.128.128, the filter you want is this:

```
if (hostname eq "123.124.125.126") then "reject"
```

The name of the field ("hostname" here) depends on your log data -- use the name that your log data uses. For instance, IIS W3C format calls the field c_ip, so for IIS you would use this:

```
if (c_ip eq "123.124.125.126") then "reject"
```

You can get a list of the fields in your profile by running Sawmill from the command line with "-p profilename -a llf".

The next time you rebuild the database, hits from your IP address will be rejected, and will not appear in the statistics.

Rejecting all hits from a particular domain is very similar; if your domain is mydomain.com, and your server is set to look up IP addresses, then you can use this filter:

```
if (ends_with(hostname, ".mydomain.com") then "reject"
```

If your server logs hostnames as IP addresses (and does not resolve them to hostnames with DNS), you can use the subnet for your domain instead; for instance, if all hits from mydomain.com will come from the subnet 128.128.128, then you can use this filter:

```
if (starts_with(hostname, "128.128.128.")) then "reject"
```

FAQ: Discarding hits from spiders

Question: How can I throw away all the spider hits, so I only see statistics on non-spider hits?

Short Answer: Use a Log Filter to reject all hits from spiders (and worms).

Long Answer

Create a new filter to reject all hits from spiders. The easiest way to create log filters is in the Log Filter Editor, in the Log Filters section of the Config. To get to the Log Filters editor, click Show Config in the Profiles list (or click Config in the reports), then click Log Data down the left, then click Log Filters. To create the filter:

- Select the Filter tab
- Select the filter type: "If a condition is met, then perform an action"
- Click the New Condition link to create the condition we will test for
- Select Spider from the drop down list of available fields
- Select "is NOT equal" from the Operator list
- Type in "(not a spider)" without the quotes, into the value field
- Click OK
- Click the New Action link
- Select "Reject log entry" from the drop down list of actions
- Click OK
- Click on the Sort Filters tab, and set this filter as the top filter, so that it runs first.
- You can create a name for this filter, like "Reject Spiders Filter" and a comment, on the Comment tab, as well.

You can also use the Advanced Expression Syntax option from the Filter Type drop down list, on the Filter tab, and type in this filter expression into the value field:

```
if (spider ne "(not a spider)") then "reject";
```

Then rebuild your database, and all hits from spiders will be discarded.

For more details on Filters see **Using Log Filters**.

FAQ: Filtering All but One Domain

Question: Can Sawmill generate statistics on just one domain, from a log file containing log data from many domains?

Short Answer: Yes. Add a log filter that rejects hits from all other domains.

Long Answer

Yes. This can be done easily using a log filter. To do this, click Show Config in the profiles list, click Log Filters, and create a new log filter with this value:

```
if (server_domain ne "mydomain.com") then "reject"
```

Replace mydomain.com with the actual domain, and replace server_domain with the name of the log field which reports the server domain in your log data. Sometimes, this field is called cs_host. If there is no such field in your log data, then you'll need to use a different log format in order to filter by domain.

The next time you rebuild the database, all log entries from domains other than the one you entered will be rejected, leaving only statistics from the one domain.

FAQ: Excluding a File or folder

Question: How can I remove a particular file or directory from the statistics?

Short Answer: Use a log filter to reject all hits on that file or directory.

Long Answer

Create a new log filter to reject all hits on that file or directory. To do this, click Show Config in the profiles list, click Log Filters, and create a new log filter with this value:

```
if (page eq "/robots.txt") then "reject";
```

The filter above rejects hits on the /robots.txt file. Or use this:

```
if (starts_with(page, "/somedir/")) then "reject";
```

The filter above rejects all hits on the /somedir/ directory.

The next time you rebuild the database, all hits on that page or directory will be rejected, so they will not appear in the statistics.

By the way, the same technique can be used to filter hits based on any field, for instance all hits from a particular host or domain, or all hits from a particular referrer, or all hits from a particular authenticated user.

FAQ: Creating Custom Fields

Question: How can I group my events in broad categories (like "internal" vs. "external" or "monitoring" vs. "actual"), and see the events on each category separately, or see them combined? How can I create content groups? How can I include information from an external database in my reports, e.g., include the full names of users based on the logged username, or the full names of pages based on the logged URL? How can I extract parts of the URL and report them as separate fields?

Short Answer: Create a new log field, database field, report and report menu item to track and show the category or custom value, and then use a log filter to set the log field appropriately for each entry.

Long Answer

It is often useful to publicize information in the reports which is not in the logs, but which can be derived from the data in the logs. For instance, it is useful to see events in categories other than those which naturally fall out of the data. Natural categories for web logs include page directories (the page field), months (the date/time field), or visitor domains (the hostname field). Similarly, it is useful to derive related values from the log fields values, and report them as though they were in the log data; for instance, if you have a username, you may want to report the full name, organization, and other information about the username. Sawmill treats every value of every field as a category, so you can categorize by any field in your log data. You can take advantage of this feature to make your own categories, even if those categories are not immediately clear in the log data. Categories like these are called "custom fields.". One common use of custom fields is to separate internal hits (hits from you) from external hits (hits from other people). Another use is to separate monitoring hits (hits from programs you use to monitor your own site) from actual hits (hits by browsing people). Another similar categorization is spider hits (hits from search engine robots and other robots) vs. human hits (hits by browsing people). Custom fields are also used to show metadata associated with a particular item, for instance to show whois information from an IP address, full name from a username, and other information. Sawmill does some common custom fields for you (geographic location derived from IP, hostname derived from IP, web browser derived from user-agent, and many more), but if you need to derive your own custom field, Sawmill also provides you with the "hooks" that you can do it with.

There are five steps to this, described in detail below:

1. Step 1: Create a log field.
2. Step 2: Create a database field based on that log field.
3. Step 3: Create a report based on that database field.
4. Step 4: Create a report menu item for that report.
5. Step 5: Create a log filter to populate the log field.

Step 1: Create a Log Field

Edit the profile .cfg file, in the profiles folder of the LogAnalysisInfo folder, using a text editor. Search for "log = {" and then search from there for "fields = {", to find the log fields list. Create a new field as shown below; enter the "internal" field name before the = sign (use only lower case letters, numbers, and underbars in the internal name), and enter the display "label" in the "label =" line. For instance, if you name the field category, the name and label will be the same; if you name it "my category", the name will be my_category and the label will be "my category". For this example, we will use "my category" as the field label throughout, and my_category as the field name.

```
my_category = {
  label = "my category"
  type = "flat"
  index = "0"
  subindex = "0"
} # my_category
```

Step 2: Create a Database Field Based on the Log Field

Still editing the profile .cfg from above, search for "database = {" and then search from there for "fields = {", to find the database fields list. Add a field like this:

```
my_category = {
  label = "my category"
```

```
    log_field = "my_category"
    type = "string"
    suppress_top = "0"
    suppress_bottom = "2"
  } # my_category
```

Step 3: Create a Report Based on the Database Field

Amending the profile .cfg from above, search for "statistics = {" and then search from there for "reports = {", to find the database fields list. Find an existing table report; the file_type report may be a good choice; otherwise pick any report with 'type = "table"'. Copy this entire report, paste to duplicate it. Now edit the report to customize it for the new field. The edited version is shown below, with modifications in bold. The modifications are:

1) The report name and report element name have been changed.

2) The database_field_name has been changed so that the table is generated from the my_category field.

3) The labels on the report element and table column have been changed to "My Category".

4) The field_name for first table column has been changed to my_category so the first column displays the my_category field values. The comments (#) have also been changed, though this is not essential.

```
  my_category = {
    report_elements = {
      my_category = {
        label = "My Category"
        type = "table"
        database_field_name = "my_category"
        sort_by = "hits"
        sort_direction = "descending"
        show_omitted_items_row = "true"
        omit_parenthesized_items = "true"
        show_totals_row = "true"
        starting_row = "1"
        ending_row = "10"
        only_bottom_level_items = "false"
        show_graph = "false"
        columns = {
          0 = {
            type = "string"
            visible = "true"
            field_name = "my_category"
            data_type = "string"
            header_label = "My Category"
            display_format_type = "string"
            main_column = "true"
          } # 0
          1 = {
            header_label = "{=capitalize(database.fields.hits.label)=}"
            type = "number"
            show_number_column = "true"
            show_percent_column = "true"
            show_bar_column = "true"
            visible = "true"
            field_name = "hits"
            data_type = "int"
            display_format_type = "integer"
          } # 1
          2 = {
            header_label = "{=capitalize(database.fields.page_views.label)=}"
            type = "number"
            show_number_column = "true"
            show_percent_column = "false"
            show_bar_column = "false"
            visible = "true"
            field_name = "page_views"
            data_type = "int"
            display_format_type = "integer"
          } # 2
          3 = {
            header_label = "{=capitalize(database.fields.visitors.label)=}"
            type = "number"
            show_number_column = "true"
```

```
            show_percent_column = "false"
            show_bar_column = "false"
            visible = "true"
            field_name = "visitors"
            data_type = "unique"
            display_format_type = "integer"
          } # 3
          4 = {
            header_label = "{=capitalize(database.fields.size.label)=}"
            type = "number"
            show_number_column = "true"
            show_percent_column = "false"
            show_bar_column = "false"
            visible = "true"
            field_name = "size"
            data_type = "float"
            display_format_type = "bandwidth"
          } # 4
        } # columns
      } # my_category
    } # report_elements
    label = "My Category"
  } # my_category
```

Step 4: Create a Report Menu Item for the Report

Still working with the profile .cfg from above, search for "reports_menu = {" to find the reports menu. This node describes the layout of the menu at the left of the reports. It includes hierarchical groups and report nodes within each group. Find a report menu item in there with 'type = "view"' (which means it clicks to a view on a report); duplicate that item and edit it so it looks like the node below. Again, the changes are to change the name of the node, the label, the view_name (which specifies which report it should click through to view), and optionally the comment:

```
      my_category = {
        type = "view"
        label = "My Category"
        view_name = "my_category"
        visible = "true"
        visible_if_files = "true"
      } # my_category
```

If you want the report to be in a different group from the one it's in, you can move it inside the "items =" list of any other group, or directly into the reports_menu node to make it a top-level report, not in any group.

Step 5: Create a Log Filter to Populate the Log Field

This step varies greatly depending on what you're doing. Overall, what you need to do is create a log filter, in the Log Filters editor of the Config section of the web interface. You can also do it in the log.filters section of the profile .cfg, by searching for "log = {" and then "filters = ". The log filter you create should set the value of your new field. It could be something as simple as this:

```
  my_category = "some value"
```

This sets the my_category field to the same constant value for every line, but that's not very useful. A slightly more useful example is to set it to part of another field:

```
  my_category = substring(file_type, 1)
```

In this example, my_category is set to the same value as file_type, but without the first character. Much more complex manipulations are possible; you can use any expression here. You could set it like this:

```
  my_category = agent . c_ip
```

to set my_category to the concatenation of the agent field and the c_ip field (which makes a pretty good "unique visitor" identified for web logs).

Here's one real-world example of the way you might create a lookup map to set the my_category field from the username field in web logs. Start by creating a file my_category_map.cfg in the LogAnalysisInfo folder, using a text editor. In that file, create a my_category for each possible username, like this:

```
  my_category_map = {
```

```
  jack = "Sales"
  jill = "Sales"
  bob = "Marketing"
  sue = "Marketing"
  sara = "Legal"
  ken = "Engineering"
}
```

Then you can use this log filter:

```
if (subnode_exists("my_category_map", username)) then
  my_category = node_value(subnode_by_name("my_category_map", username))
else
  my_category = "Unknown Category"
```

This works because when you create a file my_category_map.cfg in LogAnalysisInfo, you're automatically creating a variable that Sawmill can access as "my_category_map". You can also use directories; e.g., if you create a file "LogAnalysisInfo/log_filter_maps/my_category_map.cfg" you can access it from log filters as log_filter_maps.my_category_map. The function subnode_exists() checks if there is a subnode if its first parameter node whose name matches the second parameter, so it will be true if the username exists in my_category_map. If it does exist, then it gets that subnode's value (e.g. "Sales") and puts it in the my_category database field; otherwise, it sets it to "Unknown Category".

This is a fairly simple example; almost infinite flexibility is possible -- see (**The Configuration Language**).

FAQ: Removing Database Fields

Question: How do I remove fields from the database to save space?

Short Answer: Delete the database.fields entry from the profile .cfg file, and delete any xref groups and reports that use it.

Long Answer

Deleting database fields reduces the size of the database, and reduces the time required to build the database. Here's how you can delete a database field:

1. Using a text editor, edit the .cfg file for your profile, in LogAnalysisInfo/profiles.
2. Search for "database = {" and then search forward from there for "fields = {" to find the database fields section. Comment out the field you don't want (or delete it). For instance, to remove the screen_dimensions field, change this section:

```
      screen_dimensions = {
        label = "DOLLARlang_stats.field_labels.screen_dimensions"
        type = "string"
        log_field = "screen_dimensions"
        suppress_top = "0"
        suppress_bottom = "2"
        always_include_leaves = "false"
      } # screen_dimensions
```

   to this:

```
#       screen_dimensions = {
#         label = "DOLLARlang_stats.field_labels.screen_dimensions"
#         type = "string"
#         log_field = "screen_dimensions"
#         suppress_top = "0"
#         suppress_bottom = "2"
#         always_include_leaves = "false"
#       } # screen_dimensions
```

3. Now that the database field is gone, you will still need to remove any references to the field from other places in the profile. Typically, there is an xref group for this field, so this needs to be removed as well. Search from the top for cross_reference_groups, and comment out the group associated with the field or delete it. For instance, for screen_dimensions field, change this section:

```
      screen_dimensions = {
        date_time = ""
        screen_dimensions = ""
        hits = ""
        page_views = ""
      } # screen_dimensions
```

   to this:

```
#       screen_dimensions = {
#         date_time = ""
#         screen_dimensions = ""
#         hits = ""
#         page_views = ""
#       } # screen_dimensions
```

4. By default, there will also be a report for the field, which has to be removed. Search for "reports = {", then search forward for the appropriate report name, which is the same as the database field name. Comment it out or delete it. For instance, search for "screen_dimensions = {", and then comment it out, replacing this:

```
      screen_dimensions = {
        report_elements = {
          screen_dimensions = {
            label = "{=capitalize(pluralize(print(database.fields.screen_dimensions.label)))=}"
            type = "table"
            database_field_name = "screen_dimensions"
            sort_by = "hits"
            sort_direction = "descending"
            show_omitted_items_row = "true"
            omit_parenthesized_items = "true"
            show_totals_row = "true"
            starting_row = "1"
            ending_row = "10"
            only_bottom_level_items = "false"
            columns = {
              0 = {
                type = "string"
                visible = "true"
                field_name = "screen_dimensions"
                data_type = "string"
                header_label = "{=capitalize(database.fields.screen_dimensions.label)=}"
                display_format_type = "string"
                main_column = "true"
              } # 0
              1 = {
                header_label = "{=capitalize(database.fields.hits.label)=}"
                type = "number"
                show_number_column = "true"
                show_percent_column = "true"
                show_bar_column = "true"
                visible = "true"
                field_name = "hits"
                data_type = "int"
                display_format_type = "integer"
              } # 1
              2 = {
                header_label = "{=capitalize(database.fields.page_views.label)=}"
                type = "number"
                show_number_column = "true"
                show_percent_column = "false"
                show_bar_column = "false"
                visible = "true"
                field_name = "page_views"
                data_type = "int"
                display_format_type = "integer"
              } # 2
            } # columns
          } # screen_dimensions
        } # report_elements
        label = "{=capitalize(pluralize(print(database.fields.screen_dimensions.label)))=}"
      } # screen_dimensions
```

to this:

```
#      screen_dimensions = {
#        report_elements = {
#          screen_dimensions = {
#            label = "{=capitalize(pluralize(print(database.fields.screen_dimensions.label)))=}"
#            type = "table"
#            database_field_name = "screen_dimensions"
#            sort_by = "hits"
#            sort_direction = "descending"
#            show_omitted_items_row = "true"
#            omit_parenthesized_items = "true"
#            show_totals_row = "true"
#            starting_row = "1"
#            ending_row = "10"
#            only_bottom_level_items = "false"
#            columns = {
#              0 = {
#                type = "string"
#                visible = "true"
#                field_name = "screen_dimensions"
#                data_type = "string"
```

```
#                header_label = "{=capitalize(database.fields.screen_dimensions.label)=}"
#                display_format_type = "string"
#                main_column = "true"
#              } # 0
#              1 = {
#                header_label = "{=capitalize(database.fields.hits.label)=}"
#                type = "number"
#                show_number_column = "true"
#                show_percent_column = "true"
#                show_bar_column = "true"
#                visible = "true"
#                field_name = "hits"
#                data_type = "int"
#                display_format_type = "integer"
#              } # 1
#              2 = {
#                header_label = "{=capitalize(database.fields.page_views.label)=}"
#                type = "number"
#                show_number_column = "true"
#                show_percent_column = "false"
#                show_bar_column = "false"
#                visible = "true"
#                field_name = "page_views"
#                data_type = "int"
#                display_format_type = "integer"
#              } # 2
#            } # columns
#          } # screen_dimensions
#        } # report_elements
#        label = "{=capitalize(pluralize(print(database.fields.screen_dimensions.label)))=}"
#      } # screen_dimensions
```

5. Now you need to remove the report element from the single_page_summary report. Search for single_page_summary, then search forward for the field name (e.g., search for "screen_dimensions = {"). Again, comment out the whole report element or delete it, replacing this:

```
          screen_dimensions = {
            label = "{=capitalize(pluralize(print(database.fields.screen_dimensions.label)))=}"
            type = "table"
            database_field_name = "screen_dimensions"
            sort_by = "hits"
            sort_direction = "descending"
            show_omitted_items_row = "true"
            omit_parenthesized_items = "true"
            show_totals_row = "true"
            starting_row = "1"
            ending_row = "10"
            only_bottom_level_items = "false"
            columns = {
              0 = {
                type = "string"
                visible = "true"
                field_name = "screen_dimensions"
                data_type = "string"
                header_label = "{=capitalize(database.fields.screen_dimensions.label)=}"
                display_format_type = "string"
                main_column = "true"
              } # 0
              1 = {
                header_label = "{=capitalize(database.fields.hits.label)=}"
                type = "number"
                show_number_column = "true"
                show_percent_column = "true"
                show_bar_column = "true"
                visible = "true"
                field_name = "hits"
                data_type = "int"
                display_format_type = "integer"
              } # 1
              2 = {
                header_label = "{=capitalize(database.fields.page_views.label)=}"
                type = "number"
                show_number_column = "true"
                show_percent_column = "false"
```

```
                show_bar_column = "false"
                visible = "true"
                field_name = "page_views"
                data_type = "int"
                display_format_type = "integer"
              } # 2
            } # columns
          } # screen_dimensions
```

with this:

```
#           screen_dimensions = {
#             label = "{=capitalize(pluralize(print(database.fields.screen_dimensions.label)))=}"
#             type = "table"
#             database_field_name = "screen_dimensions"
#             sort_by = "hits"
#             sort_direction = "descending"
#             show_omitted_items_row = "true"
#             omit_parenthesized_items = "true"
#             show_totals_row = "true"
#             starting_row = "1"
#             ending_row = "10"
#             only_bottom_level_items = "false"
#             columns = {
#               0 = {
#                 type = "string"
#                 visible = "true"
#                 field_name = "screen_dimensions"
#                 data_type = "string"
#                 header_label = "{=capitalize(database.fields.screen_dimensions.label)=}"
#                 display_format_type = "string"
#                 main_column = "true"
#               } # 0
#               1 = {
#                 header_label = "{=capitalize(database.fields.hits.label)=}"
#                 type = "number"
#                 show_number_column = "true"
#                 show_percent_column = "true"
#                 show_bar_column = "true"
#                 visible = "true"
#                 field_name = "hits"
#                 data_type = "int"
#                 display_format_type = "integer"
#               } # 1
#               2 = {
#                 header_label = "{=capitalize(database.fields.page_views.label)=}"
#                 type = "number"
#                 show_number_column = "true"
#                 show_percent_column = "false"
#                 show_bar_column = "false"
#                 visible = "true"
#                 field_name = "page_views"
#                 data_type = "int"
#                 display_format_type = "integer"
#               } # 2
#             } # columns
#           } # screen_dimensions
#         } # report_elements
```

6. Finally rebuild the database.

DOCUMENTATION

FAQ: Eliminating Internal Referrers

Question: Most of the referrers listed in the "Top referrers" view are from my own site. Why is that, and how can I eliminate referrers from my own site from the statistics?

Short Answer: These are "internal referrers"; they represent visitors going from one page of your site to another page of your site. You can eliminate them by modifying the default "(internal referrer)" log filter, changing http://www.mydomain.com/ in that filter to your website URL.

Long Answer

Referrers show which page a hit came *from* -- i.e. they show what page a visitor was on when they clicked the link that took them to your page. For most web sites, visitors arrive and then click through several pages before leaving, so most web log data has a lot of referrers that are pages on the site being analyzed. For instance, if someone visits http://www.yoursite.com/index.html, and then clicks on a link pointing to http://www.yoursite.com/page2.html, it will show up in the log data (and in the statistics) as a referrer http://www.yoursite.com/index.html. These referrers are called an "internal referrer," and under normal circumstances, you don't really care about them-- what you really want to know is which referrers brought traffic to your site, not what the referrers were once they got there.

Sawmill can't distinguish internal referrers from external referrers because it doesn't know your site's URL. So it doesn't know if a referral from http://www.yoursite.com/index.html is internal (which it is if your site is yoursite.com), or external (which it is if your site is anything else). To help Sawmill identify and hide internal referrers, you need to modify a log filter that Sawmill creates for you. Here's how:

1. Go to the Config section of your profile.
2. Click Log Filters.
3. Edit the log filter which sets referrers from "yoursite.com" to "(internal referrer)"
4. Replace "yoursite.com" with your actual site name, in that log filter.
5. Rebuild the database.

Once you've done that, the internal referrers will be suppressed in the "Top referrers" view (or they will appear as "(internal referrer)" if you've turned on parenthesized items).

DOCUMENTATION

FAQ: Page Parameters

Question: I use parameters on my pages (e.g. index.html?param1+param2), but Sawmill just shows "index.html?(parameters)." How can I see my page parameters?

Short Answer: Delete the Log Filter that converts the parameters to "(parameters)."

Long Answer

By default, Sawmill creates a log filter to convert everything after the ? in the page field to "(parameters)". In most cases that's best, because it reduces the size of the database significantly. But if you need the parameter information, it's easy to get it back--just delete that filter. You can do that like this:

1. Go to the Config section of your profile.
2. Click Log Filters.
3. If your log format is Apache or similar, find the log filter which replaces everything after "?" with "(parameters)", and delete or disable that log filter.
4. If your log format is IIS or similar, find the log filter which appends the cs_uri_query field to the cs_uri_stem field, and enable that log filter.
5. Rebuild the database.

When you view the reports, you'll see that "(parameters)" has now been replaced by actual parameters.

FAQ: Recent Statistics

Question: How can I see just the most recent day/week/month of statistics?

Short Answer: Use the Calendar, or the Filters, or use a `recentdays` filter on the command line.

Long Answer

In the reports, you can go to the Calendar view and click on a recent day, week, or month to see the statistics for that time period. You can also edit the global filters to zoom in on any collection of months or days, including the most recent ones.

However, filters made in that manner will not move forward as the date changes. If you want a statistics filter that will always show the most recent seven days automatically, then you will need to use the command line, or edit the profile manually. Sawmill's command-line filtering options are slightly more powerful than the filtering options available from the web interface. Though it's not possible in the web interface to create a filter which always shows the last seven days, it *is* possible to do this from the command line, using a `recentdays:N` filter on the date/time field. For instance, to send email showing the past seven days of data, use a command line like this:

```
SawmillCL.exe -rfcf  -cm svbe -f "recentdays:7"
```

It is also possible to use this kind of a filter in a profile file, by editing the file manually. So for instance, if you want to use a `recentdays` filter on a particular report or report element always shows the most recent seven days of data, you can edit the profile file (in the profiles folder of LogAnalysisInfo) to change the "filters" value within the report or report_element node to `recentdays:7`

FAQ: Combining Referring Domains

Question: How can I combine referrers, so hits from http://search.yahoo.com, http://dir.yahoo.com, and http://google.yahoo.com are combined into a single entry?

Short Answer: Create a log filter converting all the hostnames to the same hostname.

Long Answer

You can do this by converting all of the hostnames to a single hostname, so for instance they all appear as http://yahoo.com referrers. To do this, you need to convert all occurrences of /search.yahoo.com/, /dir.yahoo.com/, or /www.yahoo.com/ into /yahoo.com/, in the referrer field. The easiest way is to make three log filters, in the Log Filters section of the Config part of your profile:

```
referrer = replace_all(referrer, "/search.yahoo.com/", "/yahoo.com/")
referrer = replace_all(referrer, "/dir.yahoo.com/", "/yahoo.com/")
referrer = replace_all(referrer, "/www.yahoo.com/", "/yahoo.com/")
```

Then rebuild the database; the resulting statistics will combine all three referrers in a single /yahoo.com/ referrer.

A more sophisticated filter is necessary if you need to preserve some parts of the URL while converting others. In that case, you can use a regular expression filter:

```
if (matches_regexp(referrer, "^http://us\.f[0-9]*\.mail\.yahoo\.com/ym/(.*)")) then referrer = "http://us.f*.mail.yahoo.com/1"
```

This matches any referrer starting with http://us.f*N*.mail.yahoo.com/ym/ (where *N* is any integer), and while it's matching, it extracts everything after the /ym/ into the variable 1. The leading ^ ensures that the referrer starts with http://, the trailing ensures that the parenthesized .* section contains all of the remainder after /ym/, [0-9]* matches any integer, and \. matches a single period (see **Regular Expressions** for more information about regular expressions). If it matches, it sets the referrer field to http://us.f*.mail.yahoo.com/1, where 1 is the value extracted from the original URL. This allows you to collapse all http://us.f*N*.mail.yahoo.com/ URLs into a single one without losing the extra data beyond /ym/. If you don't care about the data beyond /ym/, you can use somewhat simpler (or at least easier-to-understand) filter:

```
if (matches_wildcard(referrer, "^http://us.f*.mail.yahoo.com/ym/*")) then referrer = "http://us.f*.mail.yahoo.com/ym/"
```

This one uses a wildcard comparison (if matches wildcard expression) rather than a regular expression, which allows the use of * in the expression in its more generally used meaning of "match anything". Note also that in the first line, * appears twice and each time matches anything, but in the second line it appears only once, and is a literal *, not a "match-anything" character.

FAQ: Resolving IP Numbers

Question: When I look at the top hosts and top domains, all I see are numbers (IP addresses). How do I get the domain information?

Short Answer: Turn on reverse DNS lookup in the Network options (or in your web server), or use Sawmill's "look up IP numbers" feature.

Long Answer

Your web server is tracking the IP numbers of visitors, but not their hostnames or domains. If you need hostname or domain information, you need to tell Sawmill (or your web server) to look up the IP addresses using DNS (domain name service). One way to do this is to turn on DNS lookup in your web server; that will slow down your server, but then Sawmill will report hostnames and domains without any performance penalty during log data processing.

If you're not willing to take the performance hit on your server, or if you want to analyze log data that has already been generated with IP addresses, you can turn on Sawmill's reverse DNS feature like this:

1. Log in to Sawmill as an administrator.
2. Click "Show Config" next to the profile you want to modify.
3. Click "DNS Lookup" on the left.
4. Click "Edit DNS Lookup" in the upper right.
5. Check the box labeled "Look up IP numbers using domain nameserver (DNS)."
6. Enter the hostnames or IP addresses of one or two DNS servers in the DNS server fields. You can get this information from your network administrator, or your ISP.
7. Click "Save and Close".
8. Rebuild the database, e.g., click "Rebuild Database" at the top.

Processing log data will be slower with reverse DNS turned on, but you will get full hostname and domain information.

If you have problems getting the DNS feature to resolve IP addresses, see **Problems With DNS Lookup**.

A third option is to use a separate DNS resolving program to compute your log files after the server is done writing them, and before Sawmill analyzes them. Examples include `logresolve`, which is included with the popular Apache web server, DNSTran, which runs on several platforms including Macintosh, Linux, Solaris, and IRIX.

If you're using UNIX or MacOS X, another good option is **adns**, an asynchronous DNS lookup library that includes some command-line tools for looking up IP addresses, including `adnslogres` (for Common Access format and Apache Combined format files) and `adnsresfilter` (for other types of log files). For instance, you can use the command "adnsresfilter < /path/to/my/log.file" as your log source command to use adns. adns is faster than logresolve, but more difficult to configure initially.

You can plug any command-line DNS resolver directly into Sawmill by and entering a UNIX command, surrounded by backtick characters (`), that resolves the IPs in the log file and dumps the resolved log data to the standard output stream, in this case

```
`logresolve < /path/to/my/log.file`
```

Once you've done that, Sawmill will automatically run logresolve when you process your log data, and it will resolve the data before feeding it to Sawmill.

FAQ: Adding Search Engines

Question: Can I configure Sawmill to recognize search engines other than the ones it knows already?

Short Answer: Yes -- just edit the search_engines.cfg file in the LogAnalysisInfo folder with a text editor.

Long Answer

Yes; Sawmill's search engine recognition mechanism is easily extensible. All the search engines Sawmill recognizes are described in a text file called search_engines.cfg, which is found in the LogAnalysisInfo folder of your Sawmill installation. Sawmill puts several dozen search engines in there to begin with (the big, well-known ones), but you can add as many more as you like, by editing the file with a text editor. Just add a new line for each new search engine, and the next time Sawmill processes log data, it will recognize those search engines, and it will include them in the database.

The "name" value is for whatever you want to name the search engine. This name is what will appear in the statistics. The "substring" value is a "quick check" that Sawmill uses to check if a URL *might* be a URL from that search engine. If the URL contains the "quick check" string, Sawmill then does a slower check using the "regexp" column, which is a regular expression. If the regular expression matches, Sawmill uses the parenthesized section of the regular expression as the search terms, it should be a series of search terms, separated by the plus sign (+). The parenthesized section is used to compute the search terms and search phrase statistics.

You might notice that the "substring" column is redundant -- Sawmill doesn't really need it at all, since it could just check every URL with the regular expression. The reason that second column is there is that regular expressions are relatively slow -- Sawmill can process log data much faster if it doesn't have to check every URL in the log data against dozens of regular expressions. This way, it only has to use the regular expressions on a tiny proportion of the URLs that it sees.

FAQ: Changing the Time Zone in Statistics

Question: My server logs times in GMT or UTC, but I'm in a different time zone. How can I get the statistics in my own time zone?

Short Answer: Set the date_offset option in the profile.

Long Answer

Sawmill reports times exactly as they appear in the log data -- if the time shows up as 8:00 AM in the log data, that hit will appear as 8:00 AM in the statistics. Since servers sometimes log in GMT or UTC, or some other time zone from where Sawmill is running, you may want to offset the time in your statistics to match your own time zone, rather than the server's time zone or GMT. This is easily done by editing this option in Config-> LogData-> Log Processing in the GUI. The number of hours specified in that option is added to the date/time, so if it's a negative number, it moves times backwards, and if it's positive, it moves them forwards. For instance, if you're 8 hours behind GMT (GMT-0800), and your server logs in GMT, you can set this value to -8 to get statistics in your own time zone. This option affects log entries as they are processed, so you'll need to rebuild the database after setting this option, to see the changes in the statistics.

FAQ: Hits, Visitors, etc.

Question: What are "hits"? What are "page views"? What is "bandwidth"? What are "visitors"? What are "sessions"?

Short Answer: Hits are accesses to the server; page views are accesses to HTML pages; visitors are unique visitors to the site, and sessions are visits to the site.

Long Answer

Sawmill can count web log traffic in several ways. Each way is counted independently of the others, and each has its own advantages in analyzing your traffic. The different types are:

- **Hits**. Hits are accepted log entries. So if there are 5000 entries in your log file, and there are no log filters, and all the entries are valid, i.e., none of them have corrupt dates, then Sawmill will report 5000 hits for the file. If there are log filters that reject certain log entries, then those will not appear as hits. Log entries that are accepted by the log filters will count toward the hits totals. Because there are no default filters that reject, you will generally have nearly as many reported hits as you have log entries. You can view and edit the log filters by Opening your profile from the Administrative Menu, clicking Profile Options, and then clicking the Log Filters tab. See also **Using Log Filters**.
- **Page views**. Page views correspond to hits on pages. For instance, if you're analyzing a web log, and a hit on /index.html is followed by 100 hits on 100 images, style sheets, and JavaScript files, that appear in that page, then it will count as a single page view -- the secondary files do not add to the total. This is implemented in the log filters -- page views are defined as log entries that are accepted by the log filters, and that have a page_view value set to 1 by the log filters. Log entries that are accepted by the filters, but have page_view of 0 set by the filters do not contribute to the page views total. Therefore, you have complete control over which files are "real" page views and which are not -- if Sawmill's default filters do not capture your preferred definition of page views, you can edit them until they do. By default, page views are all hits that are not GIF, JPEG, PNG, CCS, JS, and a few others. See Hits, above, for more information on log filters.
- **Visitors**. Visitors correspond roughly to the total number of people who visited the site. If a single person visits the site and looks at 100 pages, that will count as 100 page views, but only one visitor. By default, Sawmill defines visitors to be "unique hosts" -- a hit is assumed to come from a different visitor if it comes from a different hostname. This can be inaccurate due to the effects of web caches and proxies. Some servers can track visitors using cookies, and if your web logs contain this information, Sawmill can use it instead of hostnames -- just change the log_field value for the visitors database field to point to the cookie field, rather than the hostname field.
- **Bandwidth**. Bandwidth is the total number of bytes transferred. It is available only in log formats that track bytes transferred. Bandwidth is tracked for every log entry that is accepted, whether it is accepted "as a hit" or "as a page view". For log formats which track both inbound and outbound bandwidth, Sawmill can report both simultaneously.

- **Sessions**. Several of Sawmill's reports deal with "session" information, including the "sessions overview" and the "paths (clickstreams)" report. Sessions are similar to visitors, except that they can "time out." When a visitor visits the site, and then leaves, and comes back later, it will count as two sessions, even though it's only one visitor. To reduce the effect of caches that look like very long sessions, Sawmill also discards sessions longer than a specified time. The timeout interval is also customizable.

FAQ: Dynamic URLs

Question: My website uses dynamic URLs instead of static pages; i.e., I have a lot of machine-generated URLs that look like /file?param1=value1&param2=value2.... Can Sawmill report on those?

Short Answer: Yes, but you need to delete the "(parameters)" log filter first.

Long Answer

Sawmill can handle URLs/pages in any format, but by default it strips off the parameters (the part after the question mark) to save space in the database. Most people don't need the parameters, but if you have a dynamic website, you do. To see the parameters, do this:

1. Go to the Config section of your profile.
2. Click Log Filters.
3. Find the Log Filter which replaces everything after "?" with "(parameters)".
4. Delete that log filter.
5. Rebuild the database.

Now, when you look at the "Pages" or "Pages/directories" view, you should see your complete URLs, along with the parameters.

If you want to take it a step further, you can also set up log filters to extract certain sections of your URLs, and put them in custom fields, to make your statistics more readable. For instance, if you have a store with several items in it, you can create an "items" field, with an associated "Top items" view, and you can set up a log filter to extract the item number (or name) from the URL and put it in the "items" field. Or you can even set up a filter to extract the item numbers from your URLs, convert them to the actual name of the item, stick them in the "item" field, and report them in the "top items" view. This is an example of a "custom field" -- see **Creating Custom Fields** for information on how to create one.

FAQ: Parenthesized Items Omitted

Question: There's a line above some of the tables in the statistics that says, "parenthesized items omitted." What does that mean?

Short Answer: It means that some items (probably useless ones) have been omitted from the table to make the information more useful--you can show them by choosing "show parenthesized items" from the Options menu.

Long Answer

Sawmill omits parenthesized items; i.e., any item that starts with "(" and ends with ")" from some tables to make the information more useful. For instance, most hits on a web site do not come directly from a search engine, some come from links in other pages on the site, and others come from links on web sites that are not search engines, so usually the largest item in the search engines table would be on the item called "(no search engine)." Because hits from non-search-engines are not important in the search engines table, and because they dominate the numbers, making it difficult to compare "real" search engines, this item is omitted by default from the table. The way Sawmill omits it is by omitting all parenthesized items. Other examples of parenthesized items include the "(no search terms)" item in the search terms table, and the "(internal referrer)" item in the referrers table.

If you want to see all the hits in these tables, you can turn on parenthesized items in the Table Options page.

DOCUMENTATION

FAQ: Default Page Hits

Question: In my reports, I see entries for /somedir/, and /somedir/{default}, and /somedir, and /somedir/ (default page). What's the difference? I seem to have two hits for each hit because of this; one on /somedir and then one on /somedir/; what can I do to show that as one hit?

Short Answer: /somedir/ is the total hits on a directory and all its contents; /somedir is an attempt to hit that directory which was directed because it did not have the trailing slash; and the default page ones both indicate the number of hits on the directory itself (e.g., on the default page of the directory).

Long Answer

To understand why there are hits shown on both /somedir/ and /somedir, where "somedir" is the name of a directory (folder) in the web site, it is necessary to understand what happens when there is a browser tries to access http://hostname/somedir . That URL is incorrect (or at best, inefficient), because it lacks the trailing divider, which implies that somedir is a file. Here's what happens in this case:

1. The web browser asks for a file named /somedir .
2. The server checks, and finds that there is no file by that name (because it's a directory). It responds with a 302 redirect to /somedir, which basically means, "no such file, but there is a directory; maybe that's what you meant?"
3. The browser accepts the redirect, so now it requests a directory named /somedir/
4. The server notes that there is a directory by that name, and that it contains an index or default file. It responds with a 200 event, and the contents of the index file.

This looks like this in the web logs:

```
server_response=302, page=/somedir
server_response=200, page=/somedir/
```

Sawmill reports this as two hits, because it *is* two hits (two lines of log data). Sawmill differentiates the aggregate traffic *within* a directory from traffic which directly hits a directory, by using /somedir/ to represent aggregation of traffic in the directory, and using /somedir/{default} to represent hits on the directory itself (i.e., hits which resulted in the display of the default page, e.g., index.html or default.asp). So in reports, the second hit above appears as a hit on /somedir/{default}, which apppears in HTML reports as a hit on "/somedir/ (default page)".

A good solution to this is to make sure that all links refer to directories with the trailing slash; otherwise the server and browser have to do the elaborate dance above, which slows everything down and doubles the stats.

Another option is to reject all hits where server response starts with 3, using a log filter like this one:

```
if (starts_with(server_response, '3')) then 'reject'
```

This discards the first hit of the two, leaving only the "real" (corrected) one.

In summary, hits on /somedir/ in reports represent the *total* number of hits on a directory, including hits on the index page of the directory, any other files in that directory, and any other files in any subdirectory of that directory, etc. Hits on /somedir in reports represent the 302 redirects caused by URLs which lack the final /. Hits on /somedir/{default} or /somedir/(default page) represent hits on the default page of the directory.

DOCUMENTATION

FAQ: Zooming on single files

Question: How do I see the number of downloads for a particular file, i.e., a newsletter PDF, or a template file PDF?

Short Answer: Select PDF from the 'File Types' table and then use the Zoom Menu to zoom to the URL's report, then Select the PDF you need to get an overview of that file.

Long Answer

Click on the 'Content' report group from the left hand menu, then click on the 'File Types' report. When the File Types report loads, click on 'PDF' from the table and the table will re-load with just a PDF entry and a menu will appear above the table with a list of all tables in it.

From that drop down (**The Zoom To Menu**) select the 'Pages' or 'URL's' (it could be either) option and you should then see a page load that has only pages/URL's in where the file type is PDF. You can then select the PDF from that list, and you would next see an Overview for that file only.

This type of filtering uses the **Zoom Filters**, they are temporary filters that are applied on the report(s) as you click about (zoom about) the report. By clicking on any item from the left hand menu they are cancelled and you are returned to that reports default view where there are no filters set, unless the default has a filter set via the Report Editor, in which case that filter set will be applied.

If you want to filter items in the report, have it apply to the whole report and be able to turn on the filter when you need to, it is better to use the **Global Filters** that are available from the Filter Icon in the Toolbar, just above the report. These can be created and enabled and disabled as you need them, and you only need to create them once and they are stored under your username and the profile you are using for use next time you need them, zoom filters are not stored anywhere and need re-applying each time you need the filter set.

FAQ: Zooming Further

Question: How do I see more levels of statistics, i.e., how can I zoom in further?

Short Answer: Increase the "suppress below" level for this database field in the profile options.

Long Answer

Sawmill limits the number of levels you see by default to save memory, disk space, and time. You can increase the levels on any database field like this:

1. Using a text editor, open the .cfg file for your profile, in the LogAnalysisInfo/profiles folder.
2. Find the "database = {" section.
3. Within that section, find the "fields = {" section.
4. Within that section, find the database field you want to change.
5. Increase the suppress_below value for that field.
6. Save the file.
7. Rebuild the database.

Then you'll be able to see as many levels as you choose. See also **Memory, Disk, and Time Usage**.

FAQ: Weekly Statistics

Question: Can I see the number of hits per week? Can I see a "top weeks" report?

Short Answer: Yes, by using the Calendar, and/or creating a database field and a report tracking "weeks of the year."

Long Answer

The date/time field in Sawmill tracks years, months, days, hours, minutes, and seconds. Each of these units fits evenly into the larger unit (24 hours in a day, 12 months in a year, etc.). Because weeks do not fit evenly into months, Sawmill cannot easily fit weeks into the date/time hierarchy. Still, there are several ways to see weekly statistics.

One way is to use the Calendar. In the Calendar, each week is represented as a link called "week"-- clicking the link applies a filter to the date/time field that shows the hits on those seven days. This lets you zoom in on a particular week, so you can see the statistics for that week, or you can switch to other views to learn more about the activity for that week. However, if you do it that way you can't see a list or graph of weeks, with the hits for each week, the way you can for days in the "Days" report.

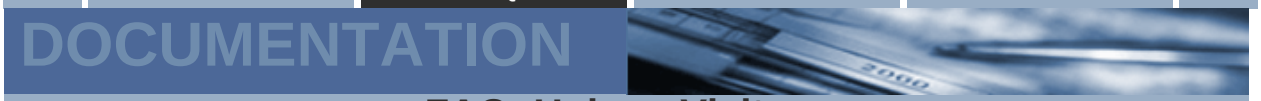
If you need a weekly graph or table, you need to track the "week of the year" log field in your database. The week of the year is a number between 1 and 52 that represents the week of the year (e.g. 1 means January 1 through January 8, etc.). You can track the week of the year field like this:

1. Open the profile file (Sawmill/LogAnalysisInfo/profiles/profilename .cfg) you want to add week_of_year reports to your favorite text editor (notepad).
2. Search for "database = {", then search for "fields = {" and scroll down until you see "day_of_week = {"
3. Copy this line and all lines until the line "} # day_of_week" and paste it all just underneath.
4. Where you see day_of_week in the new section change it to week_of_year (except use "string" where you see "display_format_type"), so it becomes:

```
day_of_week = {
  label = "day of week"
  type = "string"
  log_field = "day_of_week"
  display_format_type = "day_of_week"
  suppress_top = "0"
  suppress_bottom = "2"
  always_include_leaves = "false"
} # day_of_week
week_of_year = {
  label = "week of year"
  type = "string"
  log_field = "week_of_year"
  display_format_type = "string"
  suppress_top = "0"
  suppress_bottom = "2"
  always_include_leaves = "false"
} # week_of_year
```

5. Then search for "reports = {" and duplicate (by copy/paste as above) an existing report (the Day of week report is a good choice), and again where you see day_of_week in the new section change it to week_of_year (except use "string" where you see "display_format_type").
6. Then search for "reports_menu = {" and then "date_time_group = {" and duplicate (by copy/paste as above) an existing report menu (the Day of week report is a good choice), and again where you see day_of_week in the new section change it to week_of_year (except use "string" where you see "display_format_type").
7. Save the changes you have made .
8. Rebuild the database.

The new report will show you traffic for each week of the year.



FAQ: Unique Visitors

Question: Can Sawmill count unique visitors?

Short Answer: Yes, using unique hostname or using cookies.

Long Answer

Yes; Sawmill can tell you the number of unique visitors for any item in the database, including the number of visitors for a particular day, the number of visitors from a particular domain, the number of visitors who hit any particular page or directory, or any other type of data Sawmill can display.

By default, Sawmill uses the hostname field of your log data to compute visitors based on unique hosts. That works for all log files, but it's a somewhat inaccurate count due to the effect of proxies and caches. If your log data tracks visitors using cookies, you can easily configure Sawmill to use the cookie information instead, by changing the "visitors" database field so it is based on the cookie log field instead (in the Log Filters section of the profile Config). See also **Counting Visitors With Cookies**.

FAQ: Counting Visitors With Cookies

Question: Can Sawmill count visitors using cookies, rather than unique hostnames?

Short Answer: Yes -- it includes a built-in log format to do this for Apache, and other servers can be set up manually.

Long Answer

Yes. The use of unique browsing hostnames (or IPs) to count visitors is an imprecise method, since the same actual visitor may appear to come from several hostnames -- the same person may dial up and receive random IP addresses, or in some extreme cases, their ISP may be set up so that they have a different IP address for each hit, or several actual visitors may appear as one hostname if they're all using the same proxy. The solution to this problem is to set your web server to use cookies in order to keep track of visitors. Apache and IIS can be configured to do this, and in both cases, Sawmill can be configured to use the cookie log field, instead of the hostname, as the basis for its "visitor" field. To do this, edit your profile (in LogAnalysisInfo/profiles) with a text editor, find the "visitors" database field (look for "database = {", then "fields = {", then "visitors = {"), and change the log_field value to your cookie field; for instance, if your cookie field is cs_cookie, change it to log_field = "cs_cookie". Note that this will only work if your entire cookie field tracks the visitor cookie, and does not track any other cookies; if you have multiple cookies, you can't use the whole cookie field as your visitor ID, and you need to use the approach described below to create a visitor_id field and use a regular expression to extract your visitor cookie into it, and then change log_field to visitor_id.

Using Cookie-based Visitors IDs in Apache

In the case of Apache, it's even easier, because Sawmill includes a log format descriptor for a special "combined format plus visitor cookie" log format. The format is just the normal combined format, with the visitor ID stuck at the front of each log entry. You can log in this format by adding the following lines to your httpd.conf file:

```
  CookieTracking on
  CookieExpires "2 weeks"
  CustomLog /var/log/httpd/cookie.log "%{cookie}n %h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%
{User-Agent}i\""
```

(replace /var/log/httpd/cookie.log above with the pathname of the log you want to create). When you point Sawmill at this log file, it will recognize it as an "Apache Combined With Visitor Cookies" log, and it will set up the log filter described above for you, so you don't have to create a manual profile.

Using Cookie-based Visitors IDs in IIS

IIS has built-in support for visitor cookies -- just turn on logging of the Cookie field (extended property), or tell IIS to use "W3C Extended Log File Format" for logging, and you'll get cookies in your log data. Once you've done, that, you'll need to create a "visitor_id" log field to hold the cookie information, and use that field as the basis for your visitor database field.

An Example Filter For Extracting Cookies

If your cookie field contains more than just a visitor ID, you'll need to extract the visitor ID part of the field, and put it into a separate Sawmill's "visitor id" log field. This can be done using a regular expression filter with variable replacement. First, you'll need to create a visitor ID log field. You can do this by editing the profile .cfg file (in the profiles folder of the LogAnalysisInfo folder in your installation), and find the log.fields group (search for "log =" and then forward from there for "fields ="). Add the following log field:

```
  visitor_id = {
    label = "visitor ID"
    type = "flat"
  }
```

Next, in the same .cfg file, change database.fields.visitors.log_field to `visitor_id` (i.e. search for "database =", then search for "fields =", then search for "visitors =", and then set the `log_field` value within visitors to `visitor_id`), so the visitors field uses the visitor_id to determine whether two events are from the same visitor.

Then, add a log filter (in the Log Filters section of the profile Config, or in the log.filters section of the .cfg file) to extract the

visitor ID from the cookie. For example, suppose that the cookie field value looks like this:

```
var1=value1&var2=value2&userid=123456789&var3=value3
```

The userid cookie (the visitor id, 123456789 in this case) is buried inside the field, surrounded by other cookie names and values. To extract it you need a filter that grabs the part after "userid=" and before "&". This can be done most easily with the following filter:

```
if (matches_regular_expression(cookie, "&userid=([^&]*)&")) then visitor_id = $1
```

(for IIS, the value in quotes will be ASPSESSIONID[A-Z]*=([^&]*). This filter finds a section of the field starting with &userid=, followed by a series of non-& characters, followed by a &, and it sets the visitor id to the sequence of non-& characters it found 123456789, in this case.

Once you've added the visitor id log field, and the filter to set it, and modified the visitors database field to use the visitor id as its log field, rebuild the database. Sawmill is now using the userid value from your cookie field as your visitor id, which should make your visitors count more accurate.

FAQ: Clickstreams (Paths Through the Site)

Question: Can Sawmill show me the paths visitors took through my web site?

Short Answer: Yes; its "session paths (clickstreams)" report is very powerful.

Long Answer

Yes, very well. Most statistics packages will only show you the "top paths" or maybe the entry and exit pages; Sawmill shows you *all* the paths visitors took through the sites, in an easily navigated hierarchical report. You get complete data about every path that every visitor took through your site, click-by-click. For even more detail, you can zoom in on a particular session in the "individual sessions" report, to see the full log data of each click in the session.

FAQ: Tracking Conversions

Question: I want to track conversions-- i.e. I want to know which of my ads are actually resulting in sales. Can Sawmill do that?

Short Answer: Yes -- encode source information in your URLs and use global filters to show the top entry pages for your "success" page.

Long Answer

If you advertise your web site, one of the most useful pieces of information you can get from Sawmill is information on "conversions"; i.e. how effective your ads are at actually generating sales, sign-ups, or whatever it is that makes your site a success. Sawmill can provide highly detailed conversion information with a little effort. Here's how you do it:

1. Make sure that every URL leading to your site is tagged with information that tells you where it came from. For an Overture keyword "red umbrellas", for example, use http://www.mysite.com/?source=overture&keyword=red+umbrellas. Do the same for all ads. This is a good idea anyway (and Overture recommends it), but it's essential if you want to track conversions in Sawmill. Do this for every link leading to your site. Obviously, you can't do this for URLs you have no control over (like Google searches), but you can do it for all your ads, which are the important ones from a conversion perspective.
2. Wait for some traffic to arrive with the parameterized URLs.
3. Remove the "page parameters" log filter, in the Log Filters section of the profile Config, so Sawmill will track page parameters (see http://sawmill.net/cgi-bin/sawmilldocs?ho+faq-pageparameters).
4. View the statistics.
5. Go to the "Entry pages" view in your statistics. You should see all your full URLs there, with percentages if you want, which will tell you how much traffic each ad brought to your site. For instance, if you see that you got 1000 entries to the http://www.mysite.com/?source=overture&keyword=red+umbrellas page, then you know that your Overture ad for "red umbrellas" brought 1000 hits. That's useful information, but not conversion information--that comes next.
6. Edit the global filters in the reports, and set the filters to show only sessions that went through your "success" page. This is the page that people see after they've done whatever you wanted them to do. For instance, if success for you means a sale, then this would be the "thank you for your order" page. If success means that they sign up, this is the "you have signed up" page. If success means that they submitted a feedback form, this is the "thanks for your feedback page."
7. Now you're looking at the "Entry pages" view, but it's been filtered to show only those sessions which eventually "converted". This is exactly what you want to know -- if you see 100 entrances at http://www.mysite.com/?source=overture&keyword=red+umbrellas , then you know that 100 visitors found your site from your "red umbrellas" ad on Overture, and eventually hit your success page later in the same session. This is pure marketing gold -- by comparing the total cost of the ad, e.g., if each click is 0.10, and there were 1000 total clicks, then you spent 100 on that keyword, with the total payout of the ad. If each "success" is worth 5, then you know you made 500 from the 100 successful "red umbrellas" clicks, you can tell whether the ad is worth it. In this example, you paid 100 for the ad and got 500 in sales from it -- keep that one running!

FAQ: Finding the Top (field) for a Particular (field)

Question: How can I see the top (insert field here) for each (insert field here)? For instance, how can I see the pages hit by a particular visitor? Or the top visitors who hit a particular page? Or the top referrers for a particular day, or the top days for a particular referrer? Or the top search phrases for a search engine, the top authenticated users for a directory, the top directories accessed by an authenticated user, etc.?

Short Answer: Click on the item you're interested in, and chose the other field from "default report on zoom".

Long Answer

Sawmill can answer this sort of question for any combination of fields. All you need to do is use the zoom filters (or global filters) to zoom in on the item you want specific information for, and then use "default report on zoom" to switch to the report that shows the data you want. For instance, if you want to know the top search engines for a particular search phrase, click Search phrases, then click a particular search phrase, and then choose "Search engines" from the "Default report on zoom" menu. The resulting table will show you a breakdown by search engine of the hits for the search phrase you selected.

FAQ: Sessions For A Particular Page

Question: How can I only see the visitors that entered at a particular page, or only the visitors that hit a particular page at some point in their session?

Short Answer: Use the global filters to show only sessions containing that page; reports will only show sessions including that page.

Long Answer

In the global filters, add a filter to show only sessions containing that page. Then return to the reports; until you remove that global filter, all reports will show only traffic for sessions containing a particular page.

DOCUMENTATION

FAQ: Sessions For A Particular Search Engine

Question: How can I see only the visitors that came from a particular search engine?

Short Answer: Direct that search engine to a particular entry page, and then use global filters to show only sessions for that page.

Long Answer

Some information of this type is available in the "Search engines" view -- you can zoom in on a particular search engine by clicking its name there, and then switch to the top visitor hostnames view to see which hosts came from that search engine, and other information about the traffic from that search engine. But that only works for the first click, because after that, the log data no longer lists the originating search engine (the referrers are internal from that point on). So you can see how much traffic search engines brought, but what if you want to see what the visitors from a particular search engine did after they came to the site?

You can do that by using custom entrance pages and Global Filters. Start by pointing each search engine to its own URL, where possible. For instance, instead of pointing Overture to http://www.mysite.com/index.html, you can point it to http://www.mysite.com/index.html?source=overture. Once you've done that, then all traffic from Overture will initially arrive at the /index.html?source=overture page. By showing only sessions containing that page (see **Sessions For A Particular Page**), you can show the session activity of Overture visitors, including what paths they took, how long they stayed, and more.

You can do the same thing for any search engine, advertising campaign, or link exchange that allows you to choose your URL. It won't work quite as easily for broad search engines like Google, which let people enter your site at any point, but it's still possible to "tag" the URL similarly using a log filter -- see **Tracking Conversions**.

FAQ: Visitors vs. Session Users

Question: Why doesn't the number of visitors in the Overview match the number of session users in the "Sessions Overview" report?

Short Answer: Session information only shows users contributing page views, and other views show all visitors. Also, long sessions are discarded from the session information.

Long Answer

The configuration database is split into two major sections: the main statistics, and the session information. The main statistics contains information on all hits; the session information shows the "sessions" -- i.e. it tracks the sequence of page views of each person who visits the site. Most views show the main statistics; only the session-related views (Sessions (summary), Sessions, Paths (clickstreams), Paths through a page, Entry pages, Exit pages, Paths through a page, Session pages, etc.) show the session information. Because these two types of data are computed differently, the numbers may vary between the two.

There are two major factors that affect the session users, but do not affect the visitors. First, session information is based on page views only, while visitor information is computed based on all hits in the database. So for instance, if the website is accessed by a browser that fetches only a single image file, and never hits a page, that hit (and that host) will appear in the main statistics, but not in the session statistics. To put it another way, the visitors are the number of unique hosts who contributed hits; the session users are the number of unique hosts contributing page views. If your database is set up to track hits or bandwidth, these numbers may be significantly different. If your database tracks only page views, then visitor information will also be based on page views, and visitors and session users will be closer.

The second factor is that long sessions are discarded from the session information. By default, sessions longer than 2 hours are assumed to be "fake" sessions, resulting from the use of a large cache server by multiple users, or from spiders. Sawmill discards these sessions from the statistics, because these sessions are not accurate representations of the way any single user moves through the site -- they are semi-random juxtapositions of multiple true sessions, and are not very useful. The default maximum session duration is 2 hours; this can be customized in the Stats Misc tab of the Configuration Options. Setting this to 0 will cause all sessions to be included, eliminating this difference between visitors and session users.

Incidentally, using "visitor cookies" or "session cookies" in your log data and configuring Sawmill to use those as visitor ids (see [faq-visitorcookies] will eliminate the need for this 2-hour maximum.

DOCUMENTATION

FAQ: Emailed Reports in Outlook 2003

Question: Why do my emailed reports from Outlook 2003 not line up, everything is out of alignment?

Short Answer: Change the settings in Outlook to not load content automatically.

Long Answer

Follow these directions from the main menu in Outlook 2003. Go to the Tools menu, select Option and then Security. In the Download Settings, click change automatic download settings, then uncheck the setting for don't download pictures or content automatically in html e-mail. Click OK to close the download images dialogue box and the options dialogue box and now view your email message. The report should look fine, with all of the headers and graphs lining up.

FAQ: Exporting Data From Statistics

Question: Can I export the data from Sawmill reports to Excel or other programs?

Short Answer: Yes; click the "export" link in the toolbar above reports to export the data from that report's table in CSV format. Many programs, including Excel, can import CSV format files.

Long Answer

Sawmill supports CSV export of any table. Just view the statistics, find the table you want, and click the "export" link in the toolbar. Save the resulting file from your browser, and import it into Excel or any other program that supports CSV.

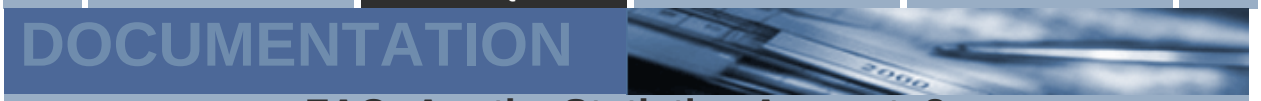
You can also generate CSV from the command line, like this:

```
SawmillCL.exe -p profilename -a ect -rn view-name
```

for instance,

```
SawmillCL.exe -p myprofile -a ect -rn file_type
```

You can also use the -f option (**Using Log Filters**) on the command line to use filters on the table data.



FAQ: Are the Statistics Accurate?

Question: I've heard that statistics like visitors, "sessions," and "paths through the site" can't be computed accurately. Is that true? Are the statistics reported by Sawmill an accurate description of the actual traffic on my site?

Short Answer: Sawmill accurately reports the data *as it appears in the log file*. However, many factors skew the data in the log file. The statistics are still useful, and the skew can be minimized through server configuration.

Long Answer

Sawmill, and all other log analysis tools, reports statistics based on the contents of the log files. With many types of servers, the log files accurately describe the traffic on the server; i.e., each file or page viewed by a visitor is shown in the log data, but web log files are trickier, due to the effects of caches, proxies, and dynamic IP addresses.

Caches are locations outside of the web server where previously-viewed pages or files are stored, to be accessed quickly in the future. Most web browsers have caches, so if you view a page and then return in the future, your browser will display the page ***without contacting the web server***, so you'll see the page but the server will not log your access. Other types of caches save data for entire organizations or networks. These caches make it difficult to track traffic, because many views of pages are not logged and cannot be reported by log analysis tools.

Caches interfere with all statistics, so unless you've defeated the cache in some way (see below), your web server statistics will not represent the actual viewings of the site. The logs are, however, the best information available in this case, and the statistics are useful. Caching means that none of the numbers you see are accurate representations of the number of pages actually viewed, bytes transferred, etc. However, you can be reasonably sure that if your traffic doubles, your web stats will double too. Put another way, web log analysis is a very good way of determining the *relative* performance of your web site, both to other websites and to itself over time. This is usually the most important thing, anyway-- since nobody can really measure true "hits," when you're comparing your hits to someone else hits, both are affected by the caching issues, so in general you can compare them successfully.

If you really need completely accurate statistics, there are ways of defeating caches. There are headers you can send which tell the cache not to cache your pages, which usually work, but are ignored by some caches. A better solution is to add a random tag to every page, so instead of loading /index.html, they load /index.html?XASFKHAFIAJHDFS. That will prevent the page from getting cached anywhere down the line, which will give you complete accurate page counts (and paths through the site). For instance, if someone goes back to a page earlier in their path, it will have a different tag the second time, and will be reloaded from the server, relogged, and your path statistics will be accurate. However, by disabling caching, you're also defeating the point of caching, which is performance optimization-- so your website will be slower if you do this. Many choose to do it anyway, at least for brief intervals, in order to get "true" statistics.

The other half of the problem is dynamic IP addresses, and proxies. This affects the "visitor" counts, in those cases where visitors are computed based on the unique hosts. Normally, Sawmill assumes that each unique originating hostname or IP is a unique visitor, but this is not generally true. A single visitor can show up as multiple IP addresses if they are routed through several proxy servers, or if they disconnect and dial back in, and are assigned a new IP address. Multiple visitors can also show up as a single IP address if they all use the same proxy server. Because of these factors, the visitor numbers (and the session numbers, which depend on them) are not particularly accurate unless visitor cookies are used (see below). Again, however, it's a reasonable number to throw around as the "best available approximate" of the visitors, and these numbers tend to go up when your traffic goes up, so they can be used as effective *comparative* numbers.

As with caching, the unique hosts issue can be solved through web server profile. Many people use visitor cookies, a browser cookie assigned to each unique visitor, and unique to them forever, to track visitors and sessions accurately. Sawmill can be configured to use these visitor cookies as the visitor ID, by extracting the cookie using a log filter, and putting it in the "visitor id" field. This isn't as foolproof as the cache-fooling method above, because some people have cookies disabled, but most have them enabled, so visitor cookies usually provide a very good approximation of the true visitors. If you want to get as much information as possible, you can configure Sawmill and/or your server to use the cookie when it's available, and the IP address when it's not, or even the true originating IP address, if the proxy passes it. Better yet, you can use the concatenation of the IP address and the user-agent field to get even closer to a unique visitor id even in cases where cookies are not available. So you can get pretty close to accurate visitor information if you really want to.

To summarize, with a default setup, with caching allowed and no visitor cookies, Sawmill will report hits and page views based on the log data, which will not precisely represent the actual traffic to the site. Sawmill goes further into the speculative realm than some tools by reporting visitors, sessions, and paths through the site. With some effort, your server can be

configured to make these numbers fairly accurate. Even if you don't go beyond the default settings, you can still use this as valuable comparative statistics, to compare the growth of your site over time, or to compare one of your sites to another.

FAQ: Session Computation

Question: How does Sawmill compute session information, like total sessions, repeat visitors, paths through the site, entry pages, exit pages, time spent per page, etc.?

Short Answer: Sawmill uses the visitor id field to identify unique visitors. It decides that a new session has begun if a visitor has been idle for 30 minutes. It rejects sessions longer than 2 hours.

Long Answer

Sawmill computes session information by tracking the page, date/time, and visitor id (which is usually the originating hostname) for each page view in the log data. When a session view is requested, it processes all of these page views at the time of the request, ignoring those that are filtered out by filters on the page or date/time fields. All other hits are included--filters on other fields are ignored in session information.

Sawmill groups the hits into initial sessions based on the visitor id-- it starts by assuming that each visitor contributed one session. It sorts the hits by date so it has a click-by-click record of the movement of each visitor.

Then it splits the sessions, using the customizable session timeout interval (30 minutes by default). Since there is no real "log out" operation in HTTP, there is no way for Sawmill to know the *real* time that a user leaves the site; it can only guess by assuming that if they didn't click anything for 30 minutes, they must have left and come back. The split step, then, increases the number of sessions, resulting in possibly more than one session per visitor.

Next, Sawmill discards sessions over 2 hours long (this is configurable). The idea behind this is that most web sessions are considerably shorter than that, so there's a good chance that any really long session is actually caused by multiple visitors using the same proxy server to visit the site. That looks like one long session because all of the hits seem to come from the proxy server. Sawmill rejects these because there is no way to tell which hits were from a particular visitor. If you're using visitor cookies to track unique visitors, this will not be a problem, so you can turn this option to a high value to see all your sessions, even those over 2 hours.

Finally, Sawmill discards sessions based on the Session Filters (which you can set in the Session Filters bar at the top of the statistics). The session filters can be set to discard all sessions except those from a particular visitor, or they can be set to discard all sessions except those which go through a particular page.

After that, Sawmill is ready to generate the statistics reports. The "Sessions Overview" report is generated by examining the sessions in various ways (for instance, the repeat visitors number is the number of visitors which have more than one session; i.e. those whose sessions were "split" by the timeout interval). The "entry pages" and "exit pages" report is generated by tabulating the first and last pages of every session. The "session pages" report is generated by finding every occurrence of each page in any session, computing how long it was from then until the next page in that session (exit pages are considered to have zero time spent per page), and tabulating the results for all pages to compute time per page and other statistics. The "paths (clickstreams)" report shows all the sessions in a single expandable view.

FAQ: Changing the graph field

Question: How do I change the field which is already graphed, e.g., from page view to bandwidth?

Short Answer: Edit the profile .cfg file, and change the field name in the numerical_fields section of that report element.

Long Answer

If you want to change the field which is graphed, in a particular report table, do this:

1. Open the profile .cfg file (in the profiles folder of the LogAnalysisInfo folder) in a text editor.
2. Find the Reports section (Search for "`reports = {`")
3. Scroll down until you see the report you want to change, for example "Days", so look for "`days = {`"
4. A few lines below that find the line that says "`graph = {`". You should see this:

```
numerical_fields = {
  hits = "true"
} # numerical_fields
```

5. Change this so that it reads:

```
numerical_fields = {
  visitors = "true"
} # numerical_fields
```

   You can substitute any numerical field name here, so page_views/hit/visitors/bytes, etc. You *must* use the internal name for the field, not the "display" label.

6. Refresh the browser to see the new graph.

NOTE: In some cases, just refreshing the browser may not actually show the new graph. You can be sure that once these changes have been made Sawmill will be producing the new graph, it is the browser's job to show it to you. You may need to empty your browser's cache for this to be seen.

FAQ: Peak Period Reports

Question: Does Sawmill do "peak period" reports (by weekday, or hour)?

Short Answer: Yes.

Long Answer

Yes. Sawmill lets you break your statistics down by any of a large number of criteria, and by more than one at a time. Among these criteria are "day of week" and "hour of day," so you can see weekday or hour information just by adding the appropriate field to your database.

FAQ: Time of Day Statistics

Question: Does Sawmill do time of day?

Short Answer: Yes.

Long Answer

Yes, Sawmill can pinpoint your hits to the second. By default, it also breaks down hits by the hour, so you can detect peak usage and other hourly information. The Log Detail report shows complete information about each event, down to the second, so you can zoom in on any part of your statistics, and then zoom down to the level of the log data to see event-by-event second-by-second what occurred.

FAQ: Tracking Exit URLs

Question: How can I tell where visitors went when they left the site?

Short Answer: Normally, you can't. However, you can set up "reflector" pages if you need this information.

Long Answer

Sawmill can show you the last page visitors hit before they exited the site, but it cannot usually show you where they went. The reason is that when they click a link on your site leading to another site, their web browser contacts the other site (not your site) for the new page--your web server is not contacted at all when someone clicks a link to leave your site. So the hit appears in the remote site's log files, not yours, and Sawmill cannot report on it because it's not in your log files.

Nevertheless, you can track exits from your site if you're willing to set up "reflector" pages. A reflector page is a page whose sole purpose is to reflect a visitor to another page. This can be done with a trivial HTML page containing only a META RELOAD tag in the HEAD section. For instance, the following simple HTML page will cause a visitor to be immediately redirected to http://www.flowerfire.com:

```
<html>
<head>
<meta http-equiv="Refresh" content="0; URL=http://www.flowerfire.com/">
</head>
</html>
```

By creating a page like this for every exit link on your site, and changing your links to point to the reflector page rather than the actual destination page, you can track exit link usage. When a visitor clicks the exit link, they will be taken to the reflector page, and then immediately reflected to the actual destination. This will happen quickly enough that they will not notice the reflection happening--it will seem to them that they went straight to the destination page. But your log data will include a hit on the reflector page, so you will be able to see which exit links are being taken. In the "exit pages" view, the reflector links will show which links were taken when leaving the site.

A more sophisticated way of doing this is to create a CGI script (or other type of script) which generates the reflector page on the fly, given a URL parameter. If you do it that way, you won't need to create a separate reflector page for each link; you can just use the same script for all your external links.

FAQ: Showing All Files

Question: How can I see *all* files that were hit on my website, not just the pages?

Short Answer: Delete or disable the 'Strip non-page-views' log filter, and rebuild the database

---

Long Answer

By default, Sawmill does not track the hits on individual image files and other non-page files when analyzing web log data, to save space in the database and reduce clutter in the "Pages" report. It does this by replacing the filename portion of the page with the value '(nonpage)', so all non-page hits will appear as values ending with '(nonpage)'. If you need this information, you need to tell Sawmill to track filenames for all hits. To do this, go to the Log Filters section of the Config section of your profile, and delete or disable the log filter called 'Strip non-page-views', which replaces the filename for non-page-view hits with '(nonpage)'. Then rebuild the database and view the reports, and *all* files (not just pages) will appear in the "Pages" and "Pages/directories" reports.

FAQ: robots.txt

Question: Why do I see hits on a file called "robots.txt" in my statistics?

Short Answer: robots.txt is a file that tells search engine spiders and robots what they can do, so a hit on robots.txt means that a spider visited your site.

Long Answer

robots.txt is a "standard" file that appears at the root level of many websites to tell search engine robots what to do on the site. Robots, also known as spiders, are computer programs that attempt to systematically visit and catalog all the pages on the Web. The file, robots.txt, tells the robots what they can or can't do on the site (whether they can index the site, which pages they may not index, etc.). Any correctly written robot will hit that page first, and follow the instructions it finds there. So the hits you're seeing are from robots.

If you don't have a robots.txt file on your site, the robots don't actually get any information--they get a "404 File Not Found" error instead, which they generally interpret as "index whatever you want."

FAQ: favicon.ico

Question: Why do I see a hits on a file called "favicon.ico" in my statistics?

Short Answer: favicon.ico is a special icon file that Internet Explorer looks for when it first visits the site.

Long Answer

Recent versions of Microsoft Internet Explorer, Safari, and other web browsers have a feature that lets website owners define an icon for their site, which will appear in the address bar, the Favorites menu, and other places. If you create an icon file called favicon.ico in a directory of your website, then any page in that directory that is bookmarked will appear in the Favorites menu with your custom icon. The browser checks for this file whenever a bookmark is created, so if you don't have the file, it will show up as a 404, "File not Found", link. As a side note, this is a good way to see who is bookmarking your site.

FAQ: Adding columns to report tables

Question: How can I add additional columns to report tables, e.g., to add a single report which reports source IP, destination IP, source port, and destination port?

Short Answer: Edit the report in the profile .cfg file to add a new item to the columns group.

Long Answer

Edit the profile .cfg file, which is in the profiles folder of the LogAnalysisInfo folder. Look for "reports = {" to find the reports list. Look down until you find a report which shows a table for one of the fields you want, e.g., in the source_ip/destination_ip/source_port/destination_port example, you would look for the destination_port report (the actual name of this report, and of field values, will vary depending on your log format). The report for destination port and number of events will look something like this:

```
destination_port = {
  report_elements = {
    destination_port = {
      label = "$lang_stats.destination_port.label"
      type = "table"
      database_field_name = "destination_port"
      sort_by = "events"
      sort_direction = "descending"
      show_omitted_items_row = "true"
      omit_parenthesized_items = "true"
      show_totals_row = "true"
      starting_row = "1"
      ending_row = "10"
      only_bottom_level_items = "false"
      show_graph = "false"
      columns = {
        0 = {
          type = "string"
          visible = "true"
          field_name = "destination_port"
          data_type = "string"
          header_label = "{=capitalize(database.fields.destination_port.label)=}"
          display_format_type = "string"
          main_column = "true"
        } # 0
        1 = {
          header_label = "{=capitalize(database.fields.events.label)=}"
          type = "events"
          show_number_column = "true"
          show_percent_column = "false"
          show_bar_column = "false"
          visible = "true"
          field_name = "events"
          data_type = "int"
          display_format_type = "integer"
        } # 2
      } # columns
    } # destination_port
  } # report_elements
  label = "Destination report"
} # destination_port
```

There may be other columns, but the two shown here are a minimum -- one for the destination port field, and one for the "events" field (might be called "packets" or something else).

To add a four-column source_ip/destination_ip/source_port/destination_port report, copy the entire thing and change the

name to custom_report. Then duplicate the destination_port column three times, and edit the copies so they're source_ip, destination_ip, and source_port. The result:

```
custom_report = {
  report_elements = {
    custom_report = {
      label = "Custom Report"
      type = "table"
      database_field_name = "destination_port"
      sort_by = "events"
      sort_direction = "descending"
      show_omitted_items_row = "true"
      omit_parenthesized_items = "true"
      show_totals_row = "true"
      starting_row = "1"
      ending_row = "10"
      only_bottom_level_items = "false"
      show_graph = "false"
      columns = {
        source_ip = {
          type = "string"
          visible = "true"
          field_name = "source_ip"
          data_type = "string"
          header_label = "{=capitalize(database.fields. source_ip.label)=}"
          display_format_type = "string"
          main_column = "true"
        } # source_ip
        destination_ip = {
          type = "string"
          visible = "true"
          field_name = "destination_ip"
          data_type = "string"
          header_label = "{=capitalize(database.fields. destination_ip.label)=}"
          display_format_type = "string"
          main_column = "true"
        } # destination_ip
        source_port = {
          type = "string"
          visible = "true"
          field_name = "source_port"
          data_type = "string"
          header_label = "{=capitalize(database.fields. source_port.label)=}"
          display_format_type = "string"
          main_column = "true"
        } # source_port
        destination_port = {
          type = "string"
          visible = "true"
          field_name = "destination_port"
          data_type = "string"
          header_label = "{=capitalize(database.fields.destination_port.label)=}"
          display_format_type = "string"
          main_column = "true"
        } # destination_port
        1 = {
          header_label = "{=capitalize(database.fields.events.label)=}"
          type = "events"
          show_number_column = "true"
          show_percent_column = "false"
          show_bar_column = "false"
          visible = "true"
          field_name = "events"
          data_type = "int"
          display_format_type = "integer"
        } # 2
      } # columns
    } # custom_report
  } # report_elements
  label = "Custom report"
} # custom_report
```

Finally, add it to the reports_menu list (again, this is easiest to do by duplicating the existing reports_menu item for destination port), like this:

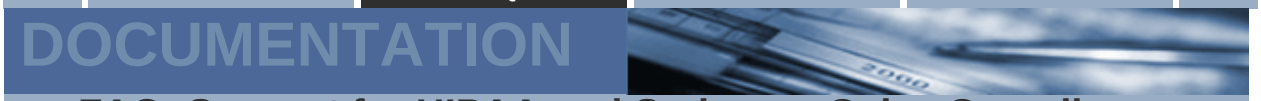
```
custom_report = {
  type = "view"
  label = "Custom Report"
  view_name = "custom_report"
  visible = "true"
  visible_if_files = "true"
} # custom_report
```

And you should have a Custom Report item in your reports menu, which links to the multi-column report.

If you're creating a two-column report, you can get an indented layout with subtables (rather than a "spreadsheet" layout) by adding the following section to the report group (e.g. right above the "} # custom_report" line, above):

```
  sub_table = {
    ending_row = "10"
    omit_parenthesized_items = "true"
    show_omitted_items_row = "true"
    show_averages_row = "false"
    show_totals_row = "true"
  } # sub_table
```

This sub_table node will work only for reports which have exactly two non-numerical columns, e.g., source_ip/destination_ip.



FAQ: Support for HIPAA and Sarbanes-Oxley Compliance

Question: Does Sawmill produce reports for HIPAA and Sarbanes-Oxley (SOX) compliance?

Short Answer: Yes, run the Single-Page Summary report.

Long Answer

Sawmill produces reports that will track the network usage, network security and give a comprensive view of who is accessing your website at any given date or time. The Single-Page Summary report will give the network detection and audit history reporting that is needed to be compliant with both HIPAA and SOX.

FAQ: Can't Access the Server

Question: When I run Sawmill, it tells me that the server is started (it shows me the URL), but when I try to access that URL, the browser says it's not available. How can I fix this?

Short Answer: You may be using a proxy server which prevents you from accessing a server running on your own machine. Try reconfiguring the proxy to allow it, or try running Sawmill on IP 127.0.0.1 (the loopback interface).

Long Answer

If you're running Windows 2003 and using Internet Explorer, look at **Can't access server with Windows 2003 and IE** first, and return here if that doesn't help.

When you first start Sawmill in web server mode, it tries to start a web server, running on the local machine, using port 8987. If this fails, it should give you an error message; if it succeeds, it should give you a URL. If you're seeing a URL when you start Sawmill, it generally means that the Sawmill server started successfully, and is ready to answer web browser requests.

Sometimes, though, when you actually try to access that URL, you may find that the server doesn't answer. Your browser may tell you that there's a DNS error, or that it couldn't contact the server, or that there's some other kind of error. If Sawmill displayed a URL, the server itself is probably working fine-- the problem is not with the server, but with the network connection to the server. This can happen, for instance, if you're using a web server proxy or cache server, and it doesn't know about the IP address of your own machine. When you contact the cache and ask to connect to your own machine, it gets confused, because normal web requests come from inside machines contacting *outside* machines, and this one is an inside machine contacting another inside machine (itself). A well-configured proxy server can handle this, but one that is not configured to handle internal requests may attempt to get the URL from the outside, and may give an error when it doesn't find it there. Some proxies/caches/firewalls will also refuse to let through traffic on port 8987, i.e., Sawmill's default port, regardless of other settings.

There are several solutions. One choice is to reconfigure the proxy or cache server to allow HTTP connections from internal machines to other internal machines, on port 8987. Then Sawmill will be able to operate in its preferred mode, on port 8987 of the machine's first IP address.

If that's not an option, you may be able to get Sawmill to work by running it on the loopback interface (IP 127.0.0.1), or on port 80 (the standard web server port). The easiest way to find a working solution is to use the command-line interface to Sawmill, at least until you have it working; you can go back to using the graphical version later. From the command line, run Sawmill like this:

```
SawmillCL.exe -ws t -sh 127.0.0.1 -wsp 80
```

This will attempt to start Sawmill's web server on IP 127.0.0.1 (the loopback interface), using port 80. This will only work if there is not a web server already running on the system-- only one server can use port 80 at a time. If you already have a web server running, use port 8987 instead. Try the command above with different IP addresses (127.0.0.1, and any IP addresses you know belongs to your computer), and different ports (try 8987 first, then 80). With a little luck one of the choices will start a server that you can connect to. Once you've got the Sawmill interface working in your web browser, you can set it to use that IP and port permanently in the Preferences, from the Administrative Menu. Once you've set the IP and port in the Preferences, you can quit the command-line Sawmill, and start using the graphical version, if you prefer.

If that still doesn't work, check if there is a firewall on your system or on your network, which is blocking traffic from your machine to itself, on port 8987. If there is, try disabling the firewall temporarily (or reconfigure it to allow the traffic), and see if it works then. If it works with the firewall disabled, and doesn't work with the firewall enabled, then the firewall is probably blocking the necessary traffic. You'll probably want to reconfigure the firewall to let the network traffic through on 8987.

If none of these work, and you have a web server running on your system there is always CGI mode. Sawmill can run under any running web server in CGI mode; if you can connect to the web server itself, you'll be able to use Sawmill by running Sawmill under your local server as a CGI program.

Finally, if you can't get Sawmill to work to your satisfaction, please contact **support@flowerfire.com**.

DOCUMENTATION

FAQ: Can't access server with Windows 2003 and IE

Question: On Windows 2003, I can't access the Sawmill server using Internet Explorer. Why not?

Short Answer: The "Internet Explorer Enhanced Security Configuration" may be enabled, blocking access; uninstall it or add 127.0.0.1:8987 to the trusted sites.

Long Answer

Windows 2003 starts up with Internet Explorer "locked down" in a highly secure mode where only certain sites are accessible. In particular, Sawmill's default URL cannot be accessed by Internet Explorer.

To enable access to Sawmill from Internet Explorer, do this:

1. Go to Internet Explorer.
2. Go to the Tools menu.
3. Choose Internet Options.
4. Click the Security tab.
5. Click the Trusted Sites icon.
6. Click the Sites button.
7. Add 127.0.0.1:8987 to the list.

Now you should be able to access Sawmill with Internet Explorer.

Alternatively, use a different browser which does not restrict access.

You can also go to the Add/Remove Programs control panel and uninstall "Internet Explorer Enhanced Security Configuration".

FAQ: Login Loops Back to Login

Question: When I try to log in to Sawmill, I get to the Admin page, but the next thing I click on takes me back to the login page. Why?

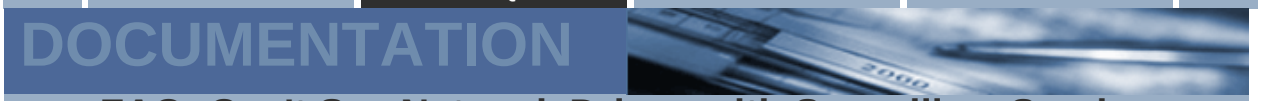
Short Answer: Your browser isn't storing the cookie Sawmill needs to maintain the login, or something is blocking the browser from sending the cookie. Make sure cookies are enabled in the browser, firewalls aren't blocking cookies, and don't use Safari 1.2.1 or earlier as your browser.

Long Answer

Sawmill keeps you logged in by using cookies in your web browser to store your login information. You'll keep getting the login screen if your browser is not passing the cookie back to Sawmill properly.

To keep this from happening, make sure cookies are enabled in your web browser. If you want to be selective about who gets cookies, at least make sure that the hostname or IP where Sawmill is running is allowed to get cookies. If your browser differentiates "session cookies" from other cookies, all you need is session cookies.

Use an approved browser--some browsers don't handle cookies very well. Approved browsers are Internet Explorer 6, Safari 1.2.2 or later, and Firefox. Others may work, but have not been verified. In particular **Safari** 1.2.1 and earlier does not handle cookies properly -- this is fixed in 1.2.2 and later.



FAQ: Can't See Network Drives with Sawmill as Service

Question: Why can't Sawmill see my mapped drive, share, directory, or mount points when I run it as a Windows Service?

Short Answer: The Service must run with the same privileged user account that has the mapped drive, share, directory, or mount point privilege.

Long Answer

The mapped drive, share, directory, or mount point is a permission issue that involves security. It is therefore necessary to have the service run using that same privileged account that the drive was originally mapped from, or an account which has permissions to access the share, etc. If the service cannot connect as the same user that has the privilege, the network resource will not be available.

Here is a step-by-step walkthrough on how to change the service logon permission:

1. Go to Control Panel.
2. Open up Services (location varies slightly with particular OS version).
3. Find the Sawmill entry (or the entry for the service running which is being used to run Sawmill and right mouse click it.
4. Select Properties.
5. Under the 'Log On' tab deselect the 'Local System Account' radio button by selecting 'This account' and hit the browse button.
6. In the 'Select User' dialog box, you may type in the privileged user's UserID or you may also browse for it. Once you have selected the correct user, click the OK button and the 'This account' field will be populated by a period, then a back slash () then the users' ID.
7. Enter the privileged user's password twice. This will show up as asterisks. This is for security reasons and to verify the password.
8. Back at the Control Panel properties for the Sawmill entry, right mouse click and select the 'restart' option.
9. When you next run Sawmill, access to the mapped drive, share, directory, or mount point will be available.
   .

DOCUMENTATION

FAQ: Can't See Network Drives in Windows 2003

Question: Why can't Sawmill see my mapped drive, share, directory, or mount points when I run it under Windows 2003?

Short Answer: Windows 2003 has a strict security policy which prevents access to network drives from Sawmill. To make it work, you need to let "everyone" permissions apply to anonymous, and remove the restriction on anonymous access to named pipes and shares (in Administrative Tools).

Long Answer

The Windows 2003 security policies prevent programs like Sawmill from accessing network drives (mapped or UNC). In order to enable access to these drives, you need to do this:

1. Go to Control Panel.
2. Open Administrative Tools.
3. Click Local Security Policy.
4. Click the Local Policies folder.
5. Click the Security Options folder.
6. Under Network Access, turn on "Let Everyone permissions apply to anonymous users."
7. Under Network Access, turn off "Restrict anonymous access to named pipes and shares."

Now Windows 2003 will let Sawmill see and access network drives.

DOCUMENTATION

FAQ: Sawmill uses too much memory for builds/updates, and is slow to view

Question: When I build or update my database with Sawmill, it uses a huge amount of memory. Then, when I view statistics, it's very slow. What can I do about that?

Short Answer: Decrease the complexity of the database.

Long Answer

The main portion of the database that uses memory are the "item lists". There is one list for each database field, and each list contains all the unique values for that field. If one of the fields in your database has many unique values, (millions) it can require a very large amount of memory to track. Simplifying the field can save memory.

To check which database field is the main culprit, look at the sizes of the files in the "items" subfolder, in the database folder (in the Databases folder of the LogAnalysisInfo folder). For instance, if `location` folder is the largest, at 500 MB, then you know that the "location" database field is responsible for the largest part of the memory usage.

When you've found the culprit, you need to reduce its memory usage. This is where you'll have to make compromises and cuts. The simplest solution is to delete the database field, and stop tracking and reporting on it. If that's not an option, you'll need to simplify the field in some way. The key point here is that you are trying to reduce the number of unique field values that Sawmill sees and tracks. The `pool` file, which is usually the largest one, contains a back-to-back list of all field values that are used in the database; if you can reduce the number of possible field values used by Sawmill, you will reduce the size of the file.

If the field is a hierarchical one (like a pathname, hostname, date/time, or URL), you can simplify it by tracking fewer levels, by adjusting the suppress_top and suppress_bottom values in the database.fields section of the profile .cfg file, in the profiles folder of the LogAnalysisInfo folder. For instance, the page field of web logs is tracked nine directories deep by default; you can simplify it by tracking only the top three levels directories. If your date/time field is set to track information to the level of minutes, you can change it back to tracking hours or days only. Usually, you will want to turn off bottom-level items checkbox for the field, since it's usually the bottom level that has all the detail.

Another possibility is to use a Log Filter to simplify the field. The default filter for web logs which replaces everything after ? with "(parameters)" is an example of this. By replacing all the various parameterized versions of a URL with a single version, this filter dramatically decreases the number of different page field values that Sawmill sees, therefore dramatically decreasing the memory usage of the "page" field. Similarly, if you have a very complex section of your directory structure, but you don't really need to know all the details, you can use a Log Filter to delete the details from your field, collapsing the entire structure into a few items.

A common source of high memory usage is a fully-tracked hostname/IP field. By default, Sawmill tracks only the first two levels of hostnames for web and proxy logs; i.e., it will tell you that a hit came from .sawmill.net, but not that it came from some.maching.sawmill.net. Because of the tremendous number of IP addresses that appear in large log files, this field can be a problem if it's set to track individual IPs, there's a checkmark that lets you do this when you create the profile. If this is happening, consider tracking only a few levels of the hostname hierarchy, instead of the full IP address.

Of course, sometimes you really need the full detail you're tracking in a very large field. If you can't reduce the detail, and you can't reduce the amount of log data, then the only solution is to get enough memory and processing power to efficiently handle the data you're asking Sawmill to track.

FAQ: Database Memory Usage

Question: I get an error 'Unable to allocate *N* bytes of memory' while building a database, and Sawmill seems to have used all my available memory. What can I do about it?

Short Answer: Use a MySQL database, and/or use a 64-bit computer and operating system, and/or simplify your database

---

Long Answer

This error means that Sawmill tried to allocate another chunk of memory (*N* additional bytes, on top of whatever it was already using), and the operating system told it that there was no more memory available for it to use. This error is usually *not* a bug; it almost always indicated that Sawmill really has exhausted all memory available. This error typically happens when using the "internal" database with a very large dataset.

The "internal" database is not nearly as efficient in its use of memory as MySQL. To put it another way, the "internal" database aims for performance above all, and mostly does it by keeping everything in memory. This means that it does not scale as well to extremely large datasets, or extremely large reports. Typically, the internal database will work well up to about 10 GB of uncompressed log data. Above that, scalability may become an issue. There are several ways in which the internal database does not scale well:

- Itemnums are kept in memory. This is the major problem when building a database. Sawmill keeps a list of all values seen for each field, e.g., a list of all IP addresses which appear in a particular field, or a list of all URLs which appear in another field, in the "itemnum" tables. These tables are kept in memory, or at least mapped to memory, so they still use available memory addressing space. In the case of an IP address field, for instance the source IP address of a web server log, each value is about ten bytes long. If there are 10 million unique IPs accessing the site, this table is 100 million bytes long, or 100 MB. Similarly for a proxy log analysis, if each unique URL is 100 bytes long and there are 10 million unique URLs in the log data, the table will be 1 GB. Tables this large can easily exceed the capabilities of a 32-bit system, which typically allows only 2 GB of memory to be used per process.
- Session analysis is done in memory. This can be a problem during reporting. When using an internal database, Sawmill computes session information in memory. The session calculation involves direct manipulation of a table with one line per page view, and 20 bytes per line. If the dataset is 100 million lines, this is 2 GB of data, which again exceeds the capacity of a 32-bit system.
- Report tables are held in memory. This can be a problem during reporting. When using the internal database, Sawmill keeps the temporary tables used during report generation in memory, and also keeps the final table in memory. The final table in particular uses a fairly memory-inefficient representation, with about 200 bytes per table cell. This is no problem for most tables, but in the extreme examples above (10 million unique IPs), a table might contain 10 million rows; if there are 5 columns that's 50 million cells, which would require about 10 GB of RAM, exceeding the abilities of any 32-bit system, and many 64-bit ones.

There are many solutions to these memory usage problems. All three issues can be improved by using a MySQL database, which uses disk files instead of RAM, for most operations. This is usually the best solution to extreme memory usage. However, even with a MySQL database, Sawmill keeps the final report table in memory, so even with a MySQL database, very large tables may exceed the memory of a 32-bit system.

Another solution is to use a 64-bit system and operating system; with a 64-bit processor, Sawmill will be able to allocate as much RAM as it needs, provided the RAM is available on the system (and it can use virtual memory if it isn't). This is the most complete solution; with a large amount of RAM on a 64-bit system, it should be possible to build extraordinarily huge databases without running out of memory.

Combining both solutions, and running Sawmill on a 64-bit system using MySQL, provides the best scalability.

If neither option is available, you'll need to simplify the dataset; see **Memory, Disk, and Time Usage** for suggestions.

If report generating is the problem, you may also be able to avoid the problem report, or use a simpler version of it. For instance, if it's the Pages report in a web server analysis, using the Pages/directories report instead will usually greatly reduce memory usage, because it generates a much smaller table by grouping files by directory. For session reports, zooming in on a particular day can greatly reduce session memory usage, since the memory usage is proportional to the size of the ***filtered*** dataset.

FAQ: Sawmill Server is Down

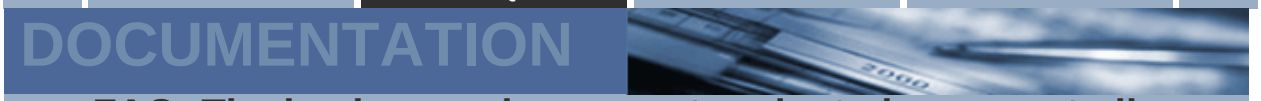
Question: I can't access Sawmill where I usually do (http://*www.xxx.yyy.zzz*:8987/) -- is your (Flowerfire's) server down?

Short Answer: No -- *your* server is down. Sawmill runs on your computer, not on ours -- contact your network administrator if you're having problems accessing it.

Long Answer

Sawmill runs as a web server on the computer where it was installed, which is a client computer, not one of our servers. So if you're having trouble accessing Sawmill through your web browser, it means that your installation of Sawmill is messed up in some way (Sawmill may not be running where you expected it to be). If you installed Sawmill yourself, you may need to restart it. If someone else installed Sawmill, please contact them (it may be your network administrator) for assistance in getting Sawmill up and running again.

On a related note, Sawmill never contacts Flowerfire, or any of Flowerfire's computers. It does not transmit log data to Flowerfire, it does not transmit statistics to Flowerfire, it does not receive any information or data from Flowerfire (the sole exception being the download of the GeoIP database, if it isn't present in the installation), and in all other ways it is a complete self-contained program that does not rely on Flowerfire's servers. Because Sawmill runs as a web server, people often assume that Sawmill is actually running on the Internet, on one of our servers, but it isn't -- it runs on your computers, and does not use the Internet or the network except where you specifically ask for it (i.e., to download files by FTP when you've requested that it do so, or to send mail when you've asked it to, or to look up IP numbers using DNS when you've asked it to).



FAQ: The background process terminated unexpectedly

Question: Sawmill displays the following error: "The background process terminated unexpectedly, without returning a result." What does that mean, and how can I fix it?

Short Answer: Sawmill has probably crashed, so this could be a bug in Sawmill. See the long answer for suggestions.

Long Answer

This error message means that Sawmill tried to do a long task, like a report generation or a database build, and while it was trying to display progress for the task, it noticed that the task was no longer running, but had not properly computed and stored its result. A task always returns a result, so this means that something has gone wrong internally in Sawmill. The most likely cause is a crash: the background task crashed, so it will never able to complete and return the result.

A crash is often due to a bug in Sawmill, but it's also possible if Sawmill runs out of memory. Make sure there is enough memory available; if you watch the memory usage while you repeat the task, does it seem to reach a high level, near the maximum memory of the system, before failing? If so, you may need more memory in your system, in order to perform that task.

If it's not memory, try running the task from the command line. If it's a database build, you can run it from the command line using this: **Building a Database from the Command Line**. If it's a crash during the report generation, you can run it from the command line similarly to a database build, but using "-a grf -rn *reportname* -ghtd report" instead of "-a bd", where *reportname* is the internal name of the report. Run Sawmill from the command line with "-p *profilename* -a lr" to get a list of reports. For instance,

```
sawmill -p myprofile -a grf -rn single_page_summary -ghtd report
```

will generate the single-page summary to a folder called "report". If this report fails, it may give a better error message about what happened to it.

Whether it fails or succeeds, email **support@flowerfire.com** with the outcome of your test. If possible, include the profile, and enough log data to reproduce the error (up to 10 MB, compressed). Report that you are seeing a crash on report generation (or database build, or whatever), and we will attempt to reproduce it on our own systems, determine the cause, and fix it, or help you resolve it, if it's not a bug.

DOCUMENTATION

FAQ: Winsock 2

Question: When I run Sawmill on Windows, I get an error: "A required DLL is missing: WS2_32.DLL." What's going on?

Short Answer: You need **Winsock 2**.

Long Answer

To run on Windows 95, and some early versions of Windows 98, Sawmill requires Winsock2, a networking component available for free from Microsoft. You can download Winsock2 from **here**.

Winsock2 is already part of Windows 98 (newer versions), Windows NT 4.0, and Windows 2000, so you do not need to download this component unless you are using Windows 95 or an older version of Windows 98.

FAQ: Missing DLL: OLEACC.DLL

Question: When I run Sawmill on Windows 98, I get an error: "A required DLL is missing: OLEACC.DLL." What's going on?

Short Answer: You need to download and install the latest **Service Pack** for Windows 98.

Long Answer

Sawmill requires a DLL called OLEACC.DLL. This DLL is part of recent versions of Windows 98, but it is not part of older versions of Windows 98. If you're running an older Windows 98, you'll need to install the latest **Service Pack** before you can run Sawmill. The service pack is a free download from Microsoft.

DOCUMENTATION

FAQ: Missing DLL: URLMON.DLL

Question: When I run Sawmill on Windows, I get an error: "A required DLL is missing: URLMON.DLL." What's going on?

Short Answer: Install the latest Internet Explorer, and the problem should go away.

Long Answer

This DLL is part of Microsoft Internet Explorer. It is also included in many recent versions of Windows. If you see this error, download and install the latest **Internet Explorer**, and the problem should go away.

FAQ: libstdc++ missing

Question: When I run Sawmill, I get an error: './sawmill: error while loading shared libraries: libstdc++.so.5: cannot open shared object file: No such file or directory'. What's going on?

Short Answer: Sawmill requires the libstdc++ library. This is available by default on many platforms, and is included in the Sawmill distribution on others (including Solaris)

Long Answer

Sawmill requires the libstdc++ library. This is available by default on many platforms, but it is not available on some older platforms, and it is often not available on Solaris. There are several ways of making this available:

- Install the g++ compiler. This is available for all platforms from **GNU**. g++ is also available as a package (e.g. a Red Hat RPM) for most platforms, and is available as an installation option on most platforms. libstdc++ is part of the g++ compiler, so installing it will install libstdc++.
  OR
- Use the libstdc++ included with Sawmill. On Solaris, the standard download of Sawmill includes the libstdc++ file (whose name starts with libstd++). If you have root access, the easiest way to install this is to copy it to /usr/lib. If you don't, you can set the environment variable LD_LIBRARY_PATH to point to your Sawmill installation. For instance, if your Sawmill installation is at /usr/sawmill, you can run this:

  ```
  setenv LD_LIBRARY_PATH "$LD_LIBRARY_PATH:/usr/sawmill"
  export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:/usr/sawmill"
  ```

  to add /usr/sawmill to the end of the LD_LIBRARY_PATH variable. You'll only need one of these two commands (the first if you're using csh as your shell, and the second if you're using bash), but it won't hurt to run them both if you're not sure which to use; you'll just get a harmless error message from the wrong one.

  These commands will last for one command-line session. If you need to make this change permanent, you can add the pathname to a separate line in the /etc/ld.conf file, or you can add the command above to one of your login scripts, e.g., .login, .cshrc, .bashrc.

  After setting LD_LIBRARY_PATH, you should be able to run Sawmill
  OR
- Find an existing libdstdc++ on your system. It is possible that you do have libstdc++ installed on your system, but it's not in your LD_LIBRARY_PATH. If that's the case, you can add the location of libstdc++ to the LD_LIBRARY_PATH using the instructions above. For instance, if it is in /usr/local/lib, you can add that to LD_LIBRARY_PATH to use it.

FAQ: Relocation error: __dynamic_cast_2

Question: When I try to run Sawmill, I get an error "relocation error: sawmill: undefined symbol: __dynamic_cast_2". How can I fix this?

Short Answer: This is a GNU library incompatibility; build Sawmill from source instead of using the binary distribution.

Long Answer

This occurs on UNIX systems, and is due to Sawmill being built expecting a different version of the GNU libraries than the one you have on your system (libstdc++). In other words, this is an operating system incompatibility -- we're building on a different version than you're running on.

The best solution is to use the "encrypted source" version of Sawmill, rather than the binary distribution for your platform; i.e., choose "encrypted source" as the "operating system" when you're downloading Sawmill. This version requires that you have a C/C++ compiler installed on your system. Follow the instructions to build Sawmill from source -- it's easy. The resulting binary will run properly on your system

If you don't have a compiler installed, please contact support@flowerfire.com.

FAQ: Problems With DNS Lookup

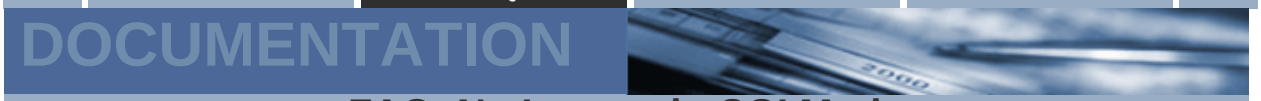
Question: Sawmill only shows me the IP addresses of my visitors, even when I turn on DNS lookup. Why?

Short Answer: Try deleting the IPNumbersCache file in LogAnalysisInfo -- see the long answer for other solutions.

Long Answer

(See **Resolving IP Numbers** for information about reverse DNS lookup).

Usually, this occurs because the DNS server can't resolve the IPs. The DNS server you're using needs to know about the IPs you're resolving. For instance, you can't use an external DNS server to resolve internal IP addresses, unless the external DNS server knows about them. Try using an internal DNS server, or another DNS server, if the first DNS server you try can't seem to resolve the IPs. It's useful to manually query the DNS server to see if it can resolve a particular IP; on most operating systems, this can be done with the "nslookup" command.



FAQ: No Images in CGI Mode

Question: I run Sawmill in CGI mode, and all the images in the menus and the reports are missing or broken. Why?

Short Answer: You may have set the "temporary folder" incorrectly during installation. Try deleting the preferences.cfg file in LogAnalysisInfo, and access Sawmill to try again.

Long Answer

When Sawmill runs as a CGI program, it includes images in its pages by creating them in a temporary folder in the web server folder, and then embedding links in the HTML so that the images it created are served by the web server. This is done by selecting a "temporary folder" and a "temporary folder URL" which point to a folder inside the web server's root folder. They both point at the same folder, but one of them is the pathname of the folder, and the other one is the URL of the folder. These two must point at the same folder for images to appear in the pages generated by Sawmill in CGI mode. If images are not appearing, it is usually because this is set incorrectly.

To correct the temporary folder, delete the preferences.cfg file in the LogAnalysisInfo folder, and access Sawmill. You will be prompted to enter the pathname and URL of the temporary folder. Make sure you see the logo on the page after you enter the temporary folder -- if the logo does not appear, click your browser's Back button and try again until you see the logo. If the logo does not appear, no other images in the Sawmill interface will either.

FAQ: Years are wrong in the statistics

Question: The statistics show the wrong years -- when I analyze data from previous years, it appears as this year, or data from this year appears in last year. Why?

Short Answer: Your log format does not include year information, so Sawmill has to guess the year. Use a different log format if possible (one which includes year information). See the long answer for a way of manually setting the year for blocks of log data.

Long Answer

Most log formats include the year as part of the date on every line, but a few (in particular, Unix Syslog format) include only month and day. In this situation, Sawmill has no way of knowing which year a particular event occurred in, so it has to guess. Recent versions of Sawmill will always guess that the event occurred in the current year; previous versions may have a particular year hard-coded in the default_log_date_year option in the profile, and will put all events in that year.

The best solution, if possible, is to use a different log format--use a log format that has year information. Then Sawmill will always categorize events in the correct year.

If that's not an option, then you will need to help Sawmill to know which data belongs in which year. There are several options, but the easiest one, if you are using Unix Syslog format, is to rename your log files so they end in *yyyy*.log, where *yyyy* is the year the log data is from. If some logs span multiple years, you will need to split those logs into files which do not cross year boundaries. For instance, if you have mail.log which contains data from 2004, 2005, and 2006, you can split it into three files, mail_2004.log, mail_2005.log, and mail_2006.log. The Unix Syslog plug-in automatically recognizes filenames which end with *yyyy*.log, and uses that value as the year when no year is available in the log data.

Another option, also for logs written by Unix Syslog, is available if the *message* part of each log line contains a full date, including year. For instance, some logging devices include "date=2006-02-01" in the log data, indicating the date of the event. In this case, even though the syslog format may not have the year, the device plug-in can extract the year from the message. This is usually a simple modification of the plug-in, but not all plug-ins have been modified to support this yet. If your log data contains year information in the message, but the reports show data from the log year, please contact **support@flowerfire.com** and we will add extraction of years from the message of your format (include a small sample of log data, as a compressed attachment).

You can also put the data in folders by year; e.g., put all of your 2005 data in a folder called /logs/2005, and all your 2006 log data in a folder called /logs/2006, and then process the data in stages using the following command lines:

```
sawmill -p profilename -a bd log.source.0.pathname /logs/2005 log.processing.
default_log_date_year 2005
sawmill -p profilename -a ud log.source.0.pathname log.processing.default_log_date_year 2006
```

The first command creates a database using all the data from 2005, using 2005 as the date. The second command processes all the data from 2005, adding it to the existing database, using 2006 as the date. The final result is that you have a database which has 2005 data in 2005 and 2006 data in 2006. From then on, you can update your database normally, and the new log data (from the most recent day) will be correctly categorized in the current year. If new data continues to be added in the wrong year, make sure that the `default_log_date_year` option is set to `thisyear` in your profile .cfg file (in LogAnalysisInfo/profiles), and in LogAnalysisInfo/profiles/default_profile.cfg.

FAQ: IIS CGI Timeout

Question: When I run Sawmill as a CGI program under IIS, I get an error message "CGI Timeout: The specified CGI application exceeded the allowed time for processing. The server has deleted the process." What can I do about that?

Short Answer: Set the IIS CGI timeout to a high value, like 999999.

Long Answer

Microsoft Internet Information Server (IIS) automatically terminates CGI programs that run for more than five minutes. Unfortunately, Sawmill can easily use that much when building a database, and if IIS terminates it, it may leave the database partly built and unusable. The solution is to reconfigure the IIS server to increase the CGI timeout to a much larger value. The following instructions are for Windows 2000 Server; other Windows variants may be slightly different:

1. In the Start Menu, go to the Settings menu, and choose Control Panels.
2. Open the Administrative Tools control panel.
3. Open the Internet Services Manager item.
4. Right-click on the computer icon in the left panel and choose Properties from the menu that appears.
5. Click "Edit..." next to "WWW Services".
6. Click the "Home Directory" tab.
7. Click the "Profile..." button.
8. Click the "Process Options" tab.
9. Enter a large value in the CGI script timeout field, like 999999.

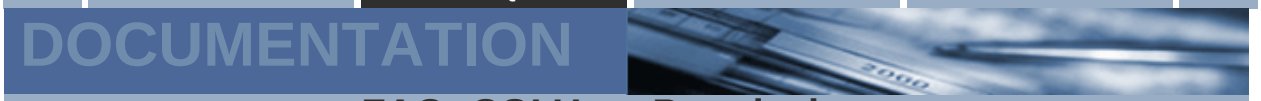
FAQ: Resetting the Administrative Password

Question: I've forgotten the password I chose for Sawmill when I first installed it; how can I reset it?

Short Answer: Delete the users.cfg file in the LogAnalysisInfo folder/directory of your installation.

Long Answer

For security reasons, Sawmill requires an administrative username and password whenever you use it, otherwise, anyone could use it to access your computer, since Sawmill is normally accessible by anyone on your network. You choose this username and password when you first run Sawmill, and it asks you for it whenever you run it again. If you forget the username or password you originally chose, you can reset your password by deleting the users.cfg file, which you will find in the LogAnalysisInfo folder of your Sawmill installation folder. **Please note: This will delete all users from Sawmill.** If you just want to delete the administrator while leaving the other users intact, you can edit users.cfg with a text editor and delete that administrator's group from the file. Once you have deleted users.cfg, access Sawmill again through a web browser, and you will be prompted to choose a new administrative username and password.



FAQ: CGI User Permissions

Question: When I run Sawmill as a CGI, it runs as a special user (nobody, web, apache, etc.). Then when I want to use Sawmill from the command line or in web server mode, the permissions don't allow it. What can I do about this?

Short Answer: Loosen the permissions in the Preferences, or run your CGI programs as a different user, or run your command line programs as the CGI user.

Long Answer

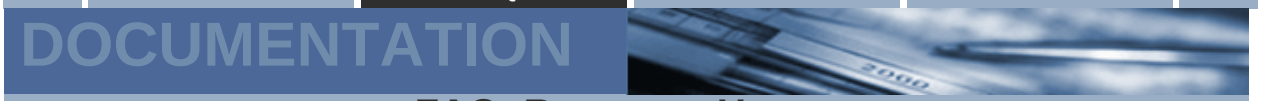
For security reasons, UNIX web servers often run CGI programs as a special user, often user nobody, or user web, or user cgi, or user apache. When you run Sawmill in CGI mode, it runs as this user, and any files it creates are owned by that user. This can cause problems if you later need to run Sawmill as a different user, for instance to run a command-line database update-- the files which were created as the CGI user will not be accessible to the non-CGI user, and you will get errors about Sawmill not being able to read or write certain files.

There are several possible solutions to this problem, and you do not need to do more than one of them:

1. **You can run your command lines as the CGI user.** This is often the easiest solution. If your CGI user is user nobody, then use "su nobody" to change to user nobody, and then run your commands as that user. Since both the CGI version and the command-line version will be running as the same user, there will be no permissions issues. You may need to configure a password, shell, and home directory for user nobody before you can log in as that user, which will require root access. This option is slightly insecure because giving user "nobody" a home directory and a shell makes it a slightly more powerful user; if the purpose of using "nobody" as the CGI user was to run CGI programs with a powerless user, this circumvents that security somewhat.

2. **You can run your CGI program as the command-line user.** If your username is "myself", then you can reconfigure your web server to run CGI programs as that user, rather than the user it's using now. You may even be able to configure the server to run only Sawmill as that user, while continuing to run other programs with the usual CGI user. Because both the CGI version of Sawmill and the command line version will be running as user "myself", there will be no permissions issues. This may be difficult to configure, however; see your web server documentation for instructions on how to configure your server to run CGI programs as a different user. On some servers, this may not be possible.

3. **You can change the permissions of the files that Sawmill creates, by editing the permissions options in the Preferences.** This is usually an insecure solution, however, since you'll need to loosen many of the permissions to 777 (everyone can read, write, execute/search), which makes your files vulnerable to modification by unauthorized users on the machine. This option may be acceptable, however, if access to the machine is limited to authorized users; i.e., if the only ones who can log in by telnet, SSH, FTP, etc. are those who are trusted Sawmill administrators.



FAQ: Resource Usage

Question: How much memory/disk space/time does Sawmill use?

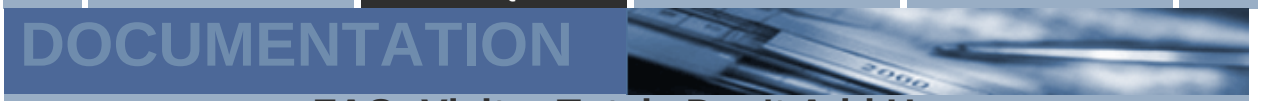
Short Answer: It depends on how much detail you ask for in the database. It uses very little if you use the default detail levels.

Long Answer

Memory usage depends mostly on the complexity of your data set (not the size). If your database has fields with millions of unique values, it will use many megabytes for each of those fields. It's uncommon for any particular field to require more than 100 MB, but in extreme cases, fields can use over 1 GB.

Disk usage is roughly proportional to the size of your uncompressed log data. As a general rule of thumb, Sawmill will use about as much disk space for its database as the uncompressed log data uses on disk. So if you're processing 500 GB of log data, you'll need about 500 GB of disk space to hold the database.

The time to process a dataset is roughly proportional to the size of a database. As of 2004, on a moderately fast single-CPU system, Sawmill typically processes between 5,000 and 10,000 lines of log data per second.



FAQ: Visitor Totals Don't Add Up

Question: When I add up the number of visitors on each day of the month, and I compare it to the total visitors for the month, they're not equal. Why not? Also, why doesn't the sum of visitors on subpages/subdirectories add up to the total for the directory, and why doesn't the sum of visitors on subdomains add up to the total for the domain, etc.? Why are there dashes (-) for the visitor totals?

Short Answer: Because "visitors" is the number of *unique* visitors, a visitor who visits every day will show up as a single visitor in each day's visitors count, but also as a single visitor for the whole month -- not 30 visitors! Therefore, simple summation of visitor numbers gives meaningless results.

Long Answer

We get this a lot as a bug report, but Sawmill is *not* counting visitors wrong. "Visitors" in Sawmill's terminology refers to unique visitors (see **Hits, Visitors, etc.**). So:

- The total hits in a month is equal to the sum of the hits on the days of the month

  and

- the total bandwidth for a month is equal to the sum of the bandwidth on the days of the month.

  and

- the total page views for a month is equal to the sum of the page views for each day of the month

  BUT

- the total number of visitors in a month is *not* usually equal to the sum of the visitors on the days of the month.

Here's why: Suppose you have a website where only one person ever visits it, but that person visits it every day. For every day of the month, you will have a single visitor. For the entire month, too, you will have a single visitor, because visitors are *unique* visitors, and there was only one visitor in the entire month, even though that visitor came back again and again. But in a 30-day month, the sum of the visitors per day will be 30, or one visitor per day. So though Sawmill will correctly report one visitor that month, it will also correctly report one visitor per day.

If what you're really looking for is "visits" rather than "visitors", so each visit will count once, even if it's the same visitor coming back over and over, then that's what Sawmill calls "sessions," and you can get information about them in the Sessions Summary and other session-related views (paths through the site, entry pages, exit pages, time spent per page).

In table reports, the total row is calculated by summing all other rows. Because visitors cannot be summed in this way, the visitors column in the total row will always be a dash (-).

FAQ: Days Are Missing from the Log Data

Question: When I look at my statistics, I see that some days are missing. I know I had traffic on those days. Why aren't they shown?

Short Answer: Your ISP may be regularly deleting or rotating your log data. Ask them to leave all your log data, or rotate it over a longer interval. It's also possible that your log data does not contain those days for another reason.

Long Answer

To save disk space, many ISPs delete, or "rotate" (rename and/or compress) the server log data regularly. For instance, instead of letting the log file grow forever, they may rename it every day, start a new one, and compress the old one; then, every week, they may delete the logs older than seven days. In other more dramatic cases, they may simply delete the log file every month or week, and restart a new one.

Though this does save disk space on the server, it presents serious problems for log analysis. When you rebuild the database with Sawmill, it processes all the existing log data, and creates a new database from it. If some of the old log data has been deleted, that data will no longer be available in the statistics. So if the ISP deletes the logs every month, and you rebuild your database, your statistics will go back one month at the most.

Similarly, when you update the database, Sawmill adds any new data in the existing log data to the database. So if the ISP deletes log files every month, and you only update your database every month on the 15th, then all the data from the 15th to the end of each month will be missing, because it was not added through an update, and it was deleted on the 1st of the month.

The best solution is to convince your ISP to keep all of your log data, and never delete any of it. If you can do that, then there will be no problem-- you'll always be able to rebuild or update your database and get all of the statistics. Since this will require more of your ISPs disk space, however, they may not be willing to do this, especially if you have a very large site, or they may charge extra for the service. Of course, if you own and manage your own server, you can do this yourself.

The second best solution, if you can't convince the ISP to keep all log data, is to store your back log files on your own system. If your ISP rotates the data through several logs before deleting the oldest one, this is easy-- just download the logs you don't have regularly (you may be able to automate this using an FTP client). If they only keep one copy, and delete it and restart it regularly, then you'll need to download that file as close to the reset time as possible, to get as much data as possible before it is deleted. This is not a reasonable way for ISPs to rotate logs, and you should try to convince them to rotate through several files before deleting the oldest one, but some of them do it this way anyway. You'll never get *all* of your log data if they use this technique-- the very last entries before deletion will always be lost-- but if you time it right you can get pretty close.

Once you have the logs on your system, you can analyze that at your leisure, without worrying about them being deleted. In this situation, you'll probably want to run Sawmill on the system where you keep the back logs.

If your log rotation is not the issue, then it may be that your log data does not contain the data for another reason. Maybe the server was down for a period, or the log data was lost in a disk outage, or it was corrupted. Look at the log data yourself, using a text editor, to make sure that it really does contain the days that you expected it to contain. If the data isn't in your logs, Sawmill cannot report statistics on it.

FAQ: Referrer Reports Missing

Question: My log data contains referrer information, but I don't see referrer reports, or search engines, or search phrases. Why not?

Short Answer: Sawmill includes referrer reports if the *beginning* of the log data includes referrers. If your log data starts without referrers, and adds it later, you won't see referrer reports. Create a new profile from the latest log file (with referrers), and change the log source to include all log data.

Long Answer

When a profile is created, Sawmill looks at the first few lines of the log data when determining which fields are present, and which reports to generate. If it sees a referrer field there, it will create a Referrer report, and Search Engines and Search Phrases reports, and other referrer-related reports.

This can be a problem if the log data does not contain referrer data at the beginning of the dataset. For instance, IIS often default to minimal logging (without referrers), and Apache often defaults to logging in Common Access Log Format (without referrers). If you later reconfigure the server to log referrers, Sawmill still won't know that, because the beginning of the log data does not contain referrers, and that's where it looks. So a profile created from the whole dataset will not report referrers, even though the later data contains referrer information.

The solution is to recreate the profile, and when it asks you where the log data is, point it to the most recent file. That file will certainly have referrer information at the beginning, so the referrer reports will be set up properly. After creating the profile, and before viewing reports or rebuilding the database, go to the Config for the profile and change the Log Source to include all your log data. Then view reports, and referrer reports will be included.

Documentation for "Maximum CPU usage"

Short description

Percent of CPU time to use while processing log data

Long description

This controls how much CPU (processor) time Sawmill uses while it is processing log data. If this is set to 100, Sawmill will use as much CPU time as possible, resulting in highest performance. If this is set to 50, Sawmill will pause for one second every second of processing when possible, resulting in an average CPU usage of 50%; all tasks will take twice as long to complete. Any value from 1 to 100 is allowed, and on most platforms Sawmill will use the requested percentage of the CPU, but on some platforms (especially older platforms), any value other than 100% will cause Sawmill to use 50% of the CPU.

Lower values may be useful in environments where other users or processes need higher priority than Sawmill, and where the operating system's own priority mechanisms are not enough to provide that. In general, you should leave this at 100 unless Sawmill's CPU usage is causing problems, and when possible you should use the operating system's own priority mechanism (e.g. `nice` for UNIX style systems, or the Task Manager in Windows) to set the process priority lower, rather than using this option. Process management is best performed by the operating system-- individual processes like Sawmill cannot manage themselves nearly as well as the operating system can manage them.

Command line usage

Command line name:	`preferences.server.maximum_cpu_usage_percent`
Command line shortcut:	`mcup`
Command line syntax:	`-mcup` *integer*
Default value:	100
Maximum value	100
Minimum value	1

All Options

FAQ: Sawmill Uses Too High a Percentage of CPU

Question: When I process log data with Sawmill, it uses most or all of my processor; it says it's using 90%, or even 100% of the CPU. Should it be doing that? Is that a problem?

Short Answer: Yes, it should do that, and it's not usually a problem. Any CPU-intensive program will do the same. However, you can lower the CPU usage if you need to with **Maximum CPU usage**.

Long Answer

Sawmill is a "CPU-bound" program while it's processing logs, which means that the microprocessor (a.k.a. CPU) is the bottleneck; the disk feeds data to Sawmill as fast as the processor can handle it. Most programs you use daily (web browsers, mail programs, word processors, etc.) are probably not CPU-bound, but any number-crunching or data-crunching program is. Other examples of programs that are typically CPU-bound include compression/decompression programs like ZIP, 3D rendering programs, and encryption programs (or encryption breakers).

Any well-behaved operating system will give a CPU-bound process as much CPU as it has available, provided that the processing needs of all other processes are met as well. Because most systems use only a small fraction of their processing power, there is usually more than 90% free CPU available at any time. This CPU is wasted unless it is used, so if there's a program like Sawmill that's continually asking for more CPU, the operating system should and will give it as much CPU as possible. If nothing else is running on the system, Sawmill will use 100% of the CPU. Since nothing else needs the CPU, that's as it should be--if the operating system only gave Sawmill 50% of the CPU, it would take twice as long to process the log data, and during the other 50% of the time, the CPU would be sitting idle, wasting time. So don't worry if Sawmill is using nearly 100% of your CPU--that's the way it's supposed to be, and it will generally have no negative effects.

The one time you may see negative effects of Sawmill's CPU usage is if there are other CPU-bound or CPU-intensive programs running on the system. In this case, because all the programs want as much CPU as possible, the operating system will split the CPU evenly between them. For instance if there are three CPU-intensive processes running, each of them will get 33% of the CPU, and each will run 1/3 as fast as it would on a lightly loaded machine. If you have an important CPU-intensive process running on your server (for instance, a very busy web server), you may want to give Sawmill a lower priority than the other processes. You can do this on UNIX systems using the "nice" command, and on Windows systems using the Process Manager. When you set Sawmill's priority to lower than the rest, it will get less than its share of CPU time, and the other processes will run faster. Sawmill, of course, will run slower. Similarly, if other processes are interfering with Sawmill's performance and you don't care about the performance of the other processes, you can increase Sawmill's priority to make it run faster, at the expense of the other processes.

Even programs that are not normally CPU-bound will have moments when they become briefly CPU-bound. For instance, a web browser sits idle most of the time, using almost no CPU, but when you load a complex page, it briefly uses as much CPU as it can get to compute and display the page. During that period, if Sawmill is running, each program will get 50% of the CPU. So the layout will take twice as long as it does when Sawmill is not running, which will make the web browser feel more sluggish than usual. Other programs, and the operating system itself, will similarly feel more sluggish while Sawmill is processing the log data. This is a side effect of having a CPU-bound program running on the system--everything else will slow down. Setting Sawmill to a lower priority will help in this situation, because the web browser will get nearly 100% of the CPU (while Sawmill is temporarily halted) while it's rendering.

If you're in an environment where you are not permitted to use more than a certain percentage of CPU, you can force Sawmill to use less processor by setting the **Maximum CPU usage** option to 50 or lower. This will cause Sawmill's long-term CPU usage to be lower, though it will continue to spike at 100% for as much as 1 second at a time. If operating-system-level priority settings are not available, this may be a reasonable option to keep Sawmill from dominating the CPU.

DOCUMENTATION

FAQ: Year 2000 Compatibility

Question: Is Sawmill Year 2000 Compatible?

Short Answer: Yes.

Long Answer

Sawmill uses an internal date/time format that will continue to work properly well beyond the year 2000, so for all of its normal operations, there will be no problem. Sawmill supports some log formats that use two-digit dates; in these cases, Sawmill assumes years 70-99 to be 1970-1999, and years 00-69 to be 2000-2069. So even with Y2K-incompatible log formats, Sawmill will work properly as long as your log data does not extend before 1970, or beyond 2069.

DOCUMENTATION

FAQ: Building a Database from the Command Line

Question: How do I build a database from the command line?

Short Answer: Run "*executable* -p *profilename* -a bd" from the command line window of your operating system.

Long Answer

You can build a database from the command line, it is not necessary to use the web interface to build a database. This is useful for debugging problems with profiles, or for building when the web interface is not available, e.g., from scripts. The exact method, and the exact command, depends on the platform; see below. See also **Additional Notes For All Platforms**.

Windows

To build a database from the command line, first open a command prompt window. One method to open a command prompt window (sometimes called a DOS window) is to click "start" in the Windows Task bar then click "run", enter "cmd" in the text box and hit return.

You will get a new window that will display something like this:

```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\username>
```

In the command prompt window you will need to move to the Sawmill installation directory using the "cd" command. Sawmill is installed by default to "C:\Program Files\Sawmill 7", to move to this directory type cd C:\Program Files\Sawmill 7 or whatever path you specified during installation.

```
C:\Documents and Settings\username >cd C:\Program Files\Sawmill 7\

C:\Program Files\Sawmill 7>
```

To get a list of internal profile names type the command "SawmillCL.exe -a lp" at the command prompt. This will display a list of the internal profile names from which you can select the profile you want to build.

```
C:\Program Files\Sawmill 7>SawmillCL.exe -a lp
Sawmill 7.2.13; Copyright (c) 2006 Flowerfire
default_profile
myprofile
```

To build you will run Sawmill with the "-p profilename -a bd" options. Replace profilename with the internal name of your profile from the list of internal profile names. The build command and related output are shown below. If you wanted to update your database you can run Sawmill with the -p profilename -a ud options.

```
C:\Program Files\Sawmill 7>SawmillCL.exe -p myprofile -a bd
Sawmill 7.1.13; Copyright (c) 2005 Flowerfire
 Reading log file: C:\Apache       [                    ] 0.00%  00:00
 Reading log file: C:\Apache       [-                   ] 3.16%  00:01
 Reading log file: C:\Apache       [######-             ] 33.33% 5000e 00:02
 Building cross-reference table 4 (worm)             [#############       ] 66.67%  00:03
 Building cross-reference table 12 (search_engine)  [##############=     ] 73.68% 00:04
 Building cross-reference table 18 (server_response) [####################] 100.00%  00:05
```

Mac OS X

To build a database from the command line, first open a terminal window. On Mac, you do this by selecting the Finder, navigating to the Applications folder, Utilities, and double clicking the Terminal application.

You will get a new window that will display something like this:

```
Last login: Mon Sep 1 10:46:44 on ttyp1
Welcome to Darwin!
[host:~] user%
```

In the terminal window you will need to move to the Sawmill installation directory using the "cd" command. Typically Sawmill is located in "/Applications/Sawmill". If you installed Sawmill somewhere else, change the directory name in the command to match. To move to this directory type "cd /Applications/Sawmill":

```
[host:~] user% cd /Applications/Sawmill
[host:/Applications/Sawmill] user%
```

To get a list of internal profile names type the command "./sawmill -a lp" at the command prompt. This will display a list of the internal profile names from which you can select the profile you want to build.

```
[host:/Applications/Sawmill] user% ./sawmill -a lp
Sawmill 7.1.13; Copyright (c) 2005 Flowerfire
default_profile
myprofile
```

To build you will run Sawmill with the "-p profilename -a bd" options. Replace profilename with the internal name of your profile from the list of internal profile names. The build command and related output are shown below. If you wanted to update your database you can run Sawmill with the -p profilename -a ud options.

```
[host:/Applications/Sawmill] user% ./sawmill -p myprofile -a bd
Sawmill 7.1.13; Copyright (c) 2005 Flowerfire
 Reading log file: /logs/Apache       [                    ] 0.00%  00:00
 Reading log file: /logs/Apache       [-                   ] 3.16%  00:01
 Reading log file: /logs/Apache       [######-             ] 33.33% 5000e 00:02
 Building cross-reference table 4 (worm)             [#############       ] 66.67%  00:03
 Building cross-reference table 12 (search_engine)  [###############=     ] 73.68% 00:04
 Building cross-reference table 18 (server_response) [####################] 100.00%  00:05
```

Linux/UNIX

Follow the Mac OS X instructions, which are basically UNIX instructions (since Mac OS X is basically UNIX); change the directories to match the location where you installed Sawmill. The executable file usually ends with the version number on Linux/UNIX platforms, so you'll need to change references from "./sawmill" to "./sawmill-7.1.14" (or whatever the version is).

Additional Notes For All Platforms

When the command completes, the database will be built. If there is an error, it will be displayed in the command line window.

To get debugging output from the build (not usually useful), you can set the SAWMILL_DEBUG environment variable to 1, before rebuilding the database with the command above. On Windows, you can set this variable with "set SAWMILL_DEBUG=1". On Mac or other operating systems, you can run "export SAWMILL_DEBUG=1" (if you're using the bash shell), or "setenv SAWMILL_DEBUG 1" (if you're using csh). If you're not sure which shell you're running, type them both; one will work (it will not give any response), and one will give an error, which you can ignore.

You can also use the -v option to get "verbose" output from the build. There are many -v options available, documented in the "Command-line output types" page of the technical manual (http://www.sawmill.net/cgi-bin/sawmill7/docs/sawmill.cgi?dp+docs.option+webvars.option_name+command_line.verbose). For very high detail (too slow for any significant build), add "-v egblpfdD" to the command line. If you add much debugging output, you may also want to add "| more" to the end of the command line to pipe the output to a pager, or to add "> out.txt" to the end of the command line to redirect the output to a file.

For more examples of command-line usage, run Sawmill from the command line with the --help option.

FAQ: Exporting Symantec SGS/SEF data to text format

Question: Sawmill does not recognize my Symantec SGS/SEF log data, because it is binary. How can I export this data to a text format so Sawmill can process it?

Short Answer: Use flatten8, or remorelog8

Long Answer

The Symantec Security Gateways plug-in is based on a text export of a binary data file on the SGS/SEF device.

To use "remotelogfile8.exe" to extract the text log from the binary data:

1. Browse to "**http://www.symantec.com/search/**"
2. search for document "2004021815290054"

To use the "flatten8" utility to extract the text log from the binary data:

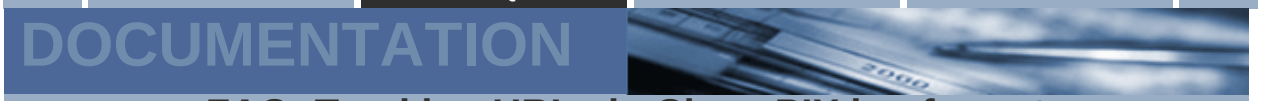
1. Review page 102 of "**Symantec™ Security Gateways - Reference Guide**" - Version 8, this is an excerpt:

Flatten utility
The flatten8 utility is shipped on the included CD and lets you perform simple log file management from the command-line. The flatten8 utility reads in the log message information from the system's XML files, and then parses in real-time the binary log file, substituting the actual error text message for its binary counterpart.
Most often, this utility is used to convert the binary log file to a more usable format for a third party utility, such as an ASCII text editor. This utility is also used to review the most recent messages, or directed to
show just statistics messages.

usage: flatten8 [-h] [-r|-s|-D] [-f] [-u seconds] [-t n] [-x xmlpath] log file ...

Where:

-h Print this message and exit.
-r Only has an effect when -s is used. Do reverse lookups on IP addresses.
-s Output stats only.
-D Do not print out error information.
-f Follow output. (Binary files, default interval 2 seconds).
-u Follow update interval in seconds. (Implies -f).
-t Tail the last 'n' log messages.
-x Next argument specifies path to XML dictionary files. This argument should not need to be used, as the XML files are placed in the default location during installation.



FAQ: Tracking URLs in Cisco PIX log format

Question: How can I track full URLs, or HTTP domains, or resolved hostnames, when analyzing PIX log data?

Short Answer: You can't track full URLs or HTTP domains, because PIX doesn't log them; but you can turn on DNS lookup in the PIX or in Sawmill to report resolved hostnames.

Long Answer

The Cisco PIX log format can be configured to log hostnames as well as IPs; if it does, the PIX plug-in will report the hostnames. This is the preferred way to get hostname information from PIX. If that's not an option, Sawmill can be configured to look up IP addresses using the DNS Lookup section of the Config page. In this case, the IP address field value will be replaced by the resolved hostname, so this resolved hostname will appear in the IPs reports. PIX does not log URLs, however, so it is not possible for Sawmill to report domains accessed. PIX reports lines like this:

Accessed URL 12.34.56.78:/some/file/test.html

This shows the source IP, which we have from another line, and the URL stem, which is slightly useful, but it does not show the domain; and resolving the IP just gives the resolved hostname, not the domain from the URL. Still, it's better than nothing; resolving the hostname might give something like server156.microsoft.com, which at least tells you it's microsoft.com traffic, even if you can't tell whether it was mdsn.microsoft.com or www.microsoft.com.

PIX can also be configured to log hostnames in the Accessed URL lines, which looks something like this:

Accessed URL 12.34.56.78 (server156.microsoft.com):/some/file/test.html

But this has the same problem; it shows the hostname, not the HTTP domain. It seems that the HTTP domain is not available from PIX log data.

The reason we recommend doing DNS lookup in PIX rather than Sawmill are twofold:

1. DNS lookup after-the-fact may give a different hostname than it would have given at the time, and the one at the time is more accurate.

2. DNS lookup in Sawmill *replaces* the IP address with the hostname, so the IP is not available in the reports. DNS lookup in PIX *adds* the hostname as a separate field, so both are available in the reports.

DOCUMENTATION

FAQ: The Name "Sawmill"

Question: Where did the name "Sawmill" come from?

Short Answer: A sawmill is a tool that processes logs, and so is Sawmill.

Long Answer

A sawmill is a tool that processes logs (the kind made from trees), and so is Sawmill (it processes web server logs).

FAQ: Frequent New Versions of Sawmill

Question: Why are new versions of Sawmill released so often? Is it buggy? Do I need to download every new version?

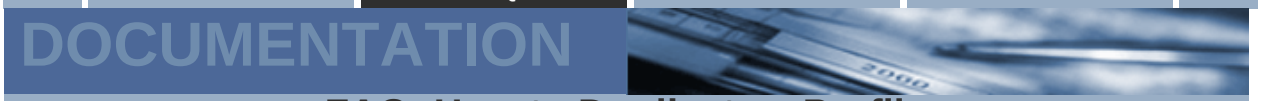
Short Answer: We ship new versions to provide our customers with the latest minor features and bug fixes quickly. Sawmill is no buggier than any other software, and you don't need to download a new release unless you're having problems with the current one.

Long Answer

We've had a few people ask us why we ship new versions of Sawmill so often. The reason is that we want to provide our customers with access to the latest minor features (e.g. new log formats) and bug fixes. Our shipping process is highly automated, so it is relatively easy for us to ship a new version, so we do it frequently.

There are bugs in Sawmill, just like there are bugs in *all* computer programs. Of course, we strive to keep the bugs to a minimum, but Sawmill is very complex software, and we get reports of a few new bugs every week. We roll these into new releases every couple of weeks, and ship them so that new downloaders won't be troubled by these bugs, and people who are experiencing them will be able to get a fixed version. Other computer programs have similar numbers of bugs, but they package more bug fixes in each release, and release versions less frequently.

Unless you're having problems with the version of Sawmill you're currently running, if you need a new feature we've added (like support for a new log format), there is no need to upgrade. You can upgrade at whatever pace you like, and skip any upgrades in the middle; each new release of Sawmill is a full release, so you don't have to have any previous version installed to use it.



FAQ: How to Duplicate a Profile

Question: How do I duplicate or clone a profile? I want to create a new profile with all the settings in my existing profile.

Short Answer: Copy the profile .cfg file, change the first line to match the filename, and change the label in the file; or use Create Many Profiles.

Long Answer

Suppose your profile appears as "My Profile" in the web interface (i.e., the label is "My Profile"), and its internal name is my_profile (i.e., the name of the file is my_profile.cfg, in the LogAnalysisInfo/profiles directory), and you want to duplicate it to create "My New Profile". Here's how:

1. Copy the file LogAnalysisInfo/profiles/my_profiles.cfg to LogAnalysisInfo/profiles/my_new_profile.cfg .
2. Edit my_new_profile.cfg with a text editor.
3. In my_new_profile.cfg, change the first line from this:

   ```
   my_profile = {
   ```

   to this:

   ```
   my_new_profile = {
   ```

4. In my_new_profile.cfg, change this line:

   ```
   label = "My Profile"
   ```

   to this:

   ```
   label = "My New Profile"
   ```

An Overview of Sawmill

Overview

Sawmill reports on your website activity. As a User of Sawmill you will have access to the following areas:

- **Admin Screen**
- **Report Interface**

Sawmill creates reports for you on-demand, so when you want to look at a report, you are seeing the most up to date information Sawmill knows about. Sawmill can be configured to update the reports whenever you want (you need to contact your Sawmill Administrator to change this) but typically they are updated daily. Sawmill's reports are on-demand and are created when you click on the "Show Reports" link. This allows you to "drill down" into the reports by selecting only those parts of the data you are interested in. The process of selecting only parts of the data is called "filtering" in Sawmill and you have a number of filter options to choose from. To find out about how to use the Sawmill filters, see **Filters**

Sawmill filters can be used to break down the reports into manageable chunks, for example, to see all traffic for a given visitor or all traffic on only one of your domains. By breaking down the reports into different types of data you are able to use Sawmill to answer some of your questions.

Examples of the questions you might have are:

- **Who is coming to my site?** - You can see the domains they're coming from (or just the IP addresses) and if your visitors login, their login username as well. The information that Sawmill can show about your visitors is limited to the IP address and domain, the username and the geographic location where the IP address or domain is registered. If your users login to your website and they have provided further information about themselves to you, then it might be possible to incorporate that information into the Sawmill reports, but this will need to be done by your Sawmill Administrator.
- **Where are they coming from?** - Sawmill can show you the geographic location of your visitors by country and by city.
- **What are they looking at?** - Sawmill can show you all the pages that have been viewed on your site, not just the top ten. Use the **Row Numbers** section of the report to see more rows. You can also view the page they looked at by clicking on the → icon next to each URL.
- **When are they visiting?** - You can see which days received the most traffic, how your traffic is growing over time, which weekdays and hours are peak, and more.
- **What are they doing?** - With the Entry Pages report, you can see where they enter the site, which paths they take through the site, the Session Paths report, and where they leave, the Exit Pages report.
- **How long do they stay?** - Sawmill can show you how long visitors are looking at your web site and how long they look at individual pages on your site. Use **The Session Overview** and **The Session Views** for this.
- **Who brought them?** - By using the Referrers report, you will see where they found the link that brought them to your site, whether it was a search engine (if it was, what they searched for in the search engine), an advertisement, a link on some other web page or if they came directly to your site. For more info, see the **Referrers** report. You can also visit the site that brought them to your site by clicking on the → icon next to each one.
- **What are they using to browse?** - This is very useful information that can tell you which operating systems and web browsers they're using. It is also possible to find out what the screen resolution and screen depth is of your visitors screens, this can be set up by your Web developer and your Sawmill Administrator.
- **Which are links broken?** - You can see a report of which pages were requested on your site that are no longer available, or were never available.

For more detail about how to interpret the reports, see **Understanding the Reports**.

Reports

The Reports present log file statistics in an attractive and easily navigable graphical format and comprise of the following components (for a clickable screenshot see the **Report Interface**):

The Report Header

The header of the Reports page is a bar containing the following:

- **Profiles**: A link to the Admin Screen, a list of all your profiles
- **Logout link**: A link to log out of Sawmill

The Report Toolbar

Below the header is a bar which contains the following:

- **The Calendar**: Click this to open the Calendar window, where you can select a single day, month, or year to use as the date/time filter. When you have selected an item in the Calendar, all reports will show only information from that time period, until the date/time filter is removed (by clicking "Show All" in the Calendar).
- **The Date Range Selector**: Click this to open the Date Range window, where you can select a range of days to use as the date/time filter. When you have selected a range in the Date Range, all reports will show only information from that time period, until the date/time filter is removed (by clicking "Show All" in the Calendar).
- **Filters**: Click this to open the Global Filter window, where you can create filters for any fields, in any combination. Filters created here dynamically affect all reports; once you have set a Global Filter, all reports will show only information for that section of the data. Global Filters remain in effect until you remove them in the Global Filter window or by unchecking and refreshing the reports filter checkbox in the upper right hand corner.

The Report Menu

Below the Reports Tool Bar and to the left of the main window is the Reports Menu, which lets you select the report to view. Clicking a Report Group will expand or collapse that group; clicking a report view will change the report display to show that one. Clicking a report view will remove any **Zoom Filters**, but will not remove **Global Filters** or **Date/Time Filters**.

The Report Menu includes the following Report Views:

- **The Overview**
- **Referrers**
- **The Session Overview**
- **The Session Views**

The Report

The main portion of the window, lower right, is occupied by the report itself. This is a view of the data selected by the filters (**Global Filters**, **Date/Time Filters**, and **Zoom Filters**). This provides one breakdown of the data specified by the filters -- you can select another report in the Reports Menu (see above) to break down the same data in a different way.

There are several parts of the report:

The Report Bar

At the top of the report is a bar containing the report label and the current global, date/time or zoom filters, if any.

The Report Graph

For some reports, there will be a graph above the table. The existence of this graph, its size, type; e.g., pie chart or bar or line, and other characteristics, varies from report to report, and can be changed by the Sawmill Administrator. The graph displays the same information as the table below it.

The Report Table

The Report Table contains the main information of the report. It displays one row per item, with the aggregated numerical values; e.g., sum of hits/page views, etc. in columns next to it. It may also include columns showing a graph or a percentage of the total traffic.

- **User Guide Index**

Filters

The Filters

 There are many filters available to you when viewing reports. The filters that are applied to the report determine what statistics you see. The filters let you "zoom in" on one part of your data (in a similar way to selecting a new view from **The Report Menu**). You can use the filters to get information about a particular day, a particular directory, a particular domain, or more.

- **Global Filters** These remain in effect until they are removed in the Global Filters page.
- **Date/Time Filters** These remain in effect until they are removed in the Calendar page.
- **Zoom Filters** These remain in effect until they are removed in the Calendar page.

All of these filters are combined when used together; i.e., an item is included if it is selected by the Global Filters AND by the Date/Time Filters AND by the Zoom Filters. For instance, if the Global Filters show events during 1am-2am, and the Zoom Filters show events on January 1, then the table will show events from January 1, during 1am-2am.

If there are no filters in place, that means you are looking at your complete data; all available data is represented by the graphs and tables shown. If the **The Report Bar** shows that there are filters active, then you are not seeing your entire data; you are seeing only a portion of it. The portion you're looking at depends on the filters. For example, if the only filter is a /dir1/ filter on the page field, then the data displayed shows only those hits which were on /dir1/ or pages contained in /dir1/ (or in other directories contained in /dir1/, or pages in them, etc.). If you have 1000 hits on your site, and 500 of them were inside /dir1/, then if there are no filters active, you will see 1000 hits in the tables and graphs, or all the hits on your site. But if there is a Filter /dir1/ on the page field, you will see 500 hits, or only those hits in /dir1/.

The filters are an extremely powerful way of getting detailed information about your site. If you want to know what day you got the most hits on /dir1/, you can do that by adding /dir1/ as a filter, and then changing to the "Years/months/days" view (see **The Report Menu**). With /dir1/ as a filter, you will see only those hits on /dir1/ (500 of them, in the example above), and you will see how those 500 hits break down by date and time. You can add an additional filter to the date/time field if you want to examine just the hits on /dir1/ on a particular day. This gives you almost infinite flexibility in how you want to examine your data.

Another way to change filters is to click on something in the statistics. For instance, clicking on a directory name in a page table will "zoom in" on that directory by adding it as the page field filter. Clicking on a month in the calendar view will "zoom in" on the month by adding it as the date/time field filter. If the "filter checkboxes and menu" option is turned on (using the Show menu; see below), you can click the check box next to any item and choose a view from the menu below the table, to set that item as a filter, and then change the view to the view you select. To answer the question, "What days did I get hits on this directory?" you can click the checkbox next to a particular directory name, and select "Show checked data in 'Top days' view", this will add the directory as a page filter, and also change to the top days statistics view.

Understanding the Reports

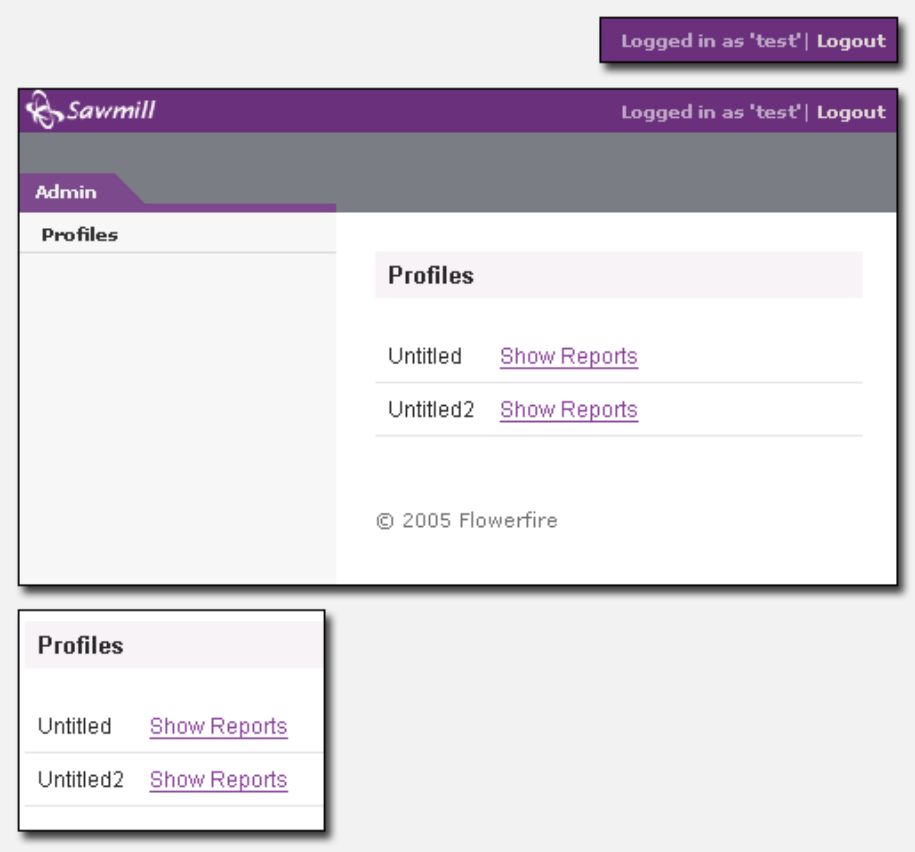
Understanding What You Are Looking At

Admin Screen

What Am I Looking At?

Below is a screenshot of what the user "test" will see once they have logged into the Sawmill interface. The box above the main image shows the user name that was used to log into Sawmill ('test' in this example) and the **Logout link**. Below the main image is the list of profiles the user has access to. If you do not see the reports you expect when you have logged in - or want to see more reports - contact your Sawmill Administrator.

This view is the Admin Screen and displays the profiles that you have access to, the Sawmill Administrator will provide access to your profiles. Click on "Show Reports" to view the **Report Interface**.



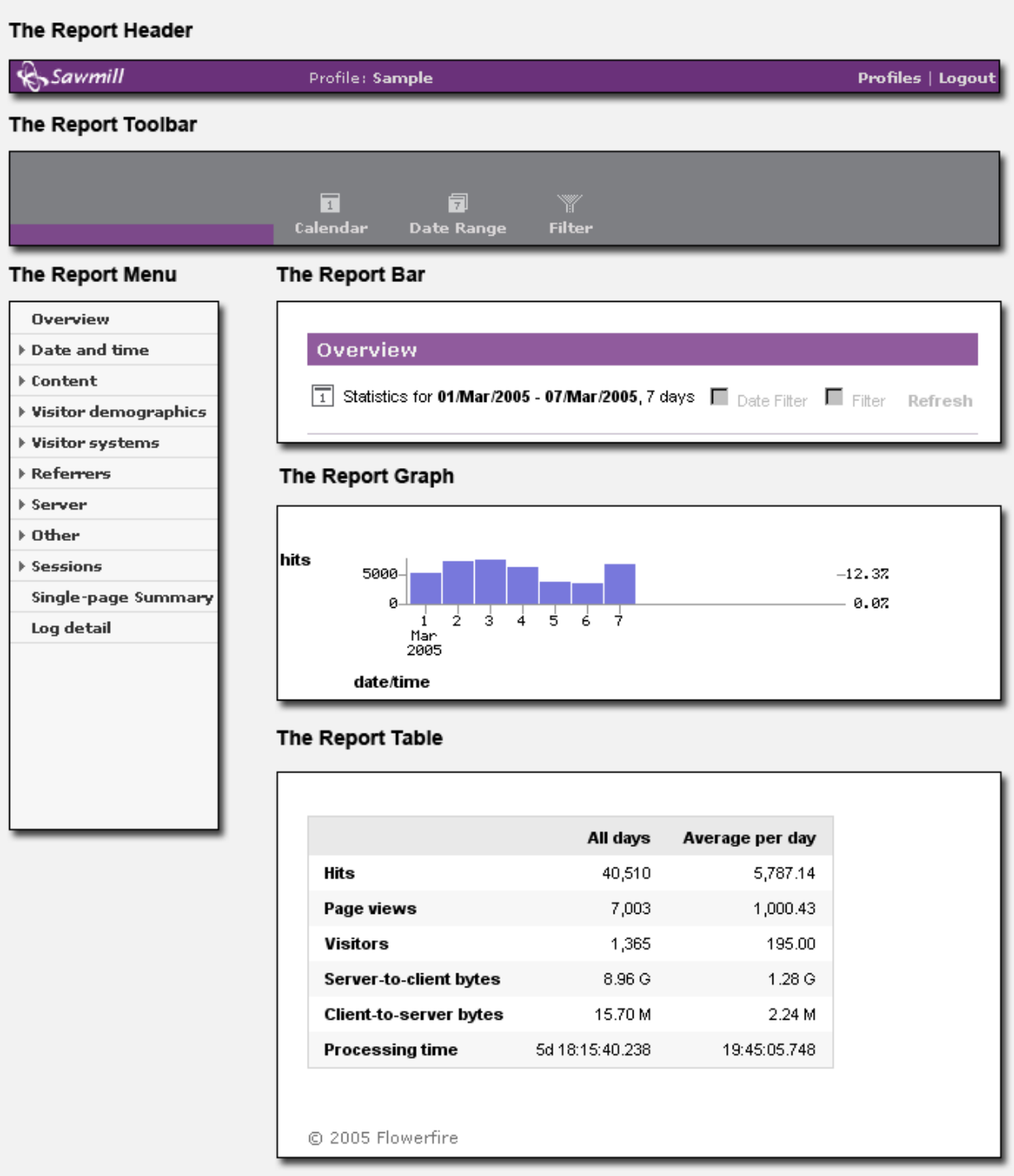
For details of the Report Interface, see **Report Interface**

Report Interface

What Am I Looking At?

Below is a screenshot of what you will see once you have logged into the Sawmill interface and clicked on the "Show Reports" link from the **Admin Screen**. It is the report interface and displays the Report Views for that profile.

Click on the different sections of the image to take you to an explanation of that section.



For details of the Admin Interface, see **Admin Screen**

- **User Guide Index**

The Report Header

The Report Header

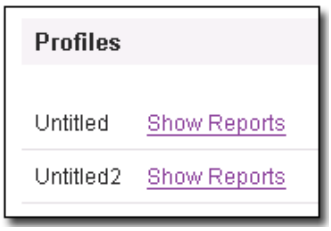
- Profiles
- Logout link

- User Guide Index

Profiles

The Profiles

The profile list on the **Admin Screen** is a list of your profiles. The Sawmill Administrator will give you access to your reports.



- User Guide Index

Logout link

Clicking the Logout link in the upper right hand corner of the browser window (right hand end of **The Report Header**) will completely log you out of Sawmill and you will need to log in again to see your profile(s).

You do not need to log out to change the active profile, just go to the **Admin Screen** by clicking on the 'Profiles' link (next to the logout link).

- **User Guide Index**

The Report Toolbar

The Report Toolbar

- The Calendar
- The Date Range Selector
- Filters

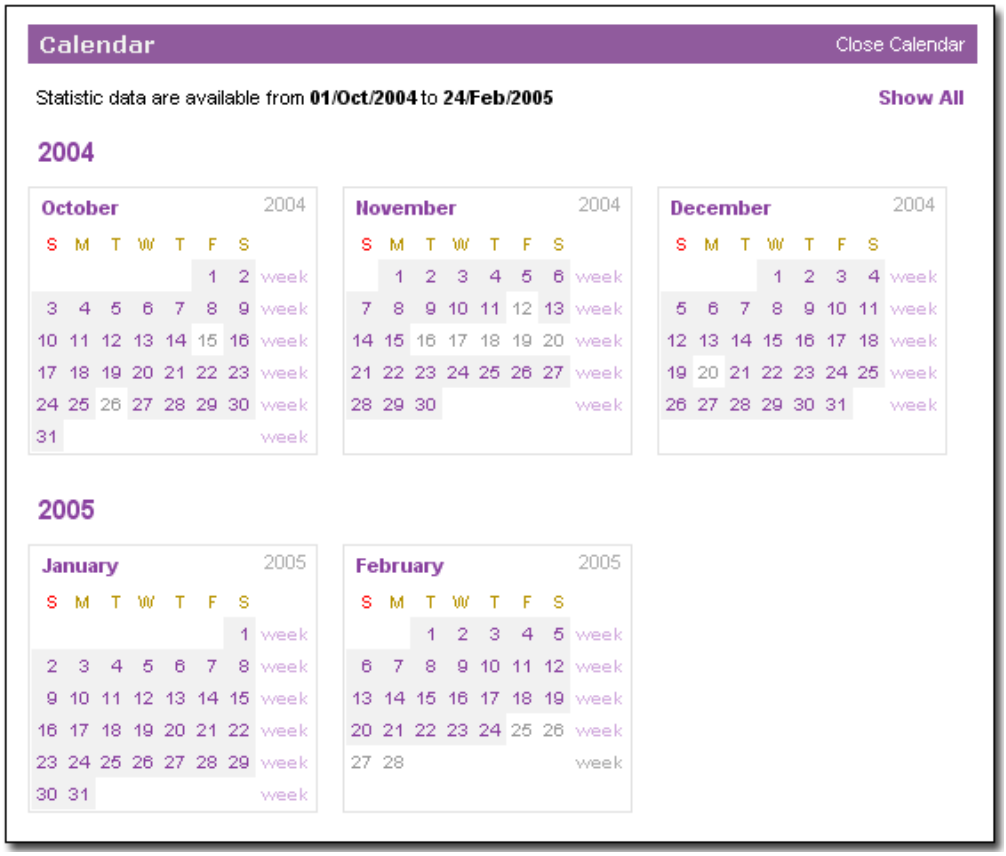
- User Guide Index

The Calendar

• User Guide Index • An Overview of Sawmill • Reports • Filters • Understanding the Reports •

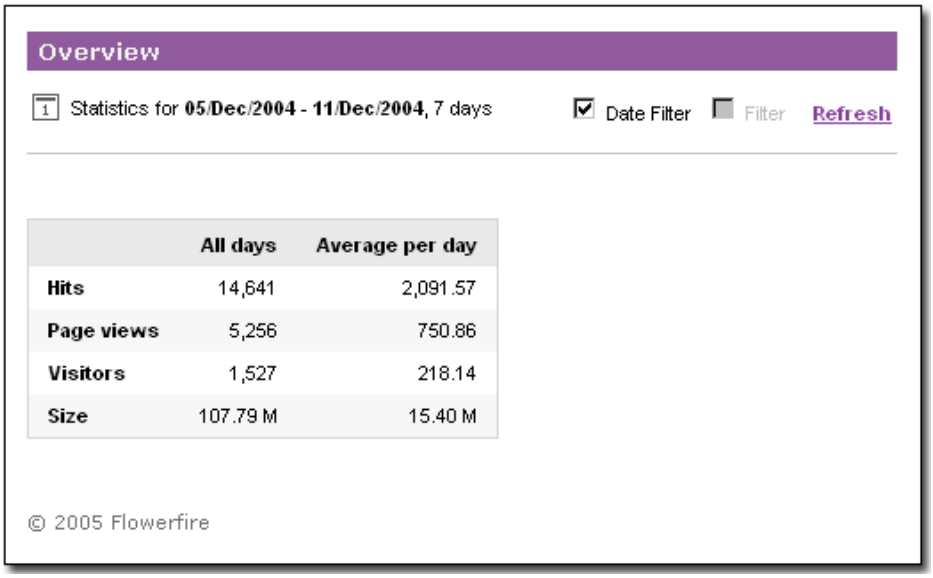
The Calendar

The Calendar shows the years, months, weeks and days during which there was traffic by displaying a clickable link for each day, week, month or year. All links that are not clickable have no data in the database. Clicking any day, week, month, or year adds **Date/Time Filters** to the reports for the selected period, thereby "zooming in" on that data.



Each day, week, month or year in the calendar that has data will be highlighted when you move the mouse over it. The images below are selecting the day 15th December 2004 and the week of 5th -> 11th:

The Calendar controls the date and time filtering in the report and once filtered, **The Report Bar** shows the time period that the report is displaying. The screenshot below shows an Overview report filtered by one week, 5th to 11th December 2004.



The **Date/Time Filters** can be removed by clicking the 'Show All' link in the top right hand corner.

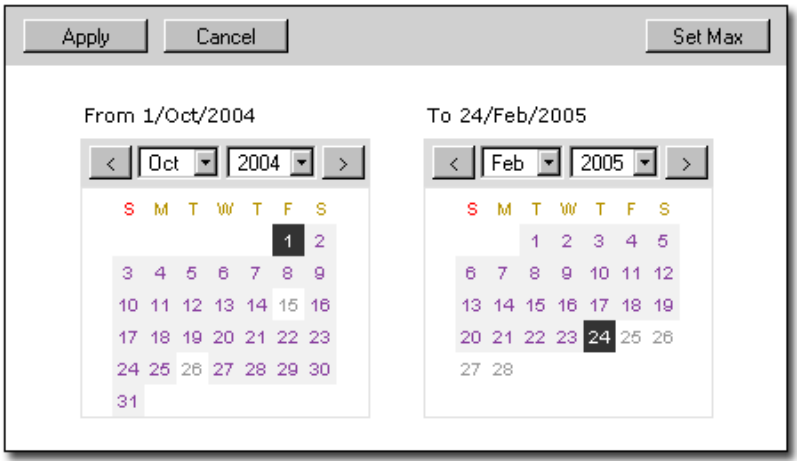
- **User Guide Index**

The Date Range Selector

The Date Range Selector


The Date Range window is where you can select a range of days to use as the date filter.

You can select any range by clicking on the dates in from and to calendars and selecting from the drop down menus, then clicking apply, or use the "Set Max" button to select all available dates.



When you have selected a range in the Date Range window, all reports will show only information from that time period, until the date filter is changed (by going into the Date Range Selector again) or removed (by clicking 'Show All' in **The Calendar**).

- **User Guide Index**

The Report Menu

The Report Menu

The Report Menu is a navigation menu for the report and its views. The screenshots show a collapsed menu and an expanded menu. The Report Group "Date and Time" has been expanded and shows the sub menu items under that group. Clicking on the group will expand that group, clicking again will collapse it.

Sawmill has many views, follow these links for more information (if you need help with other views contact your Sawmill Administrator):

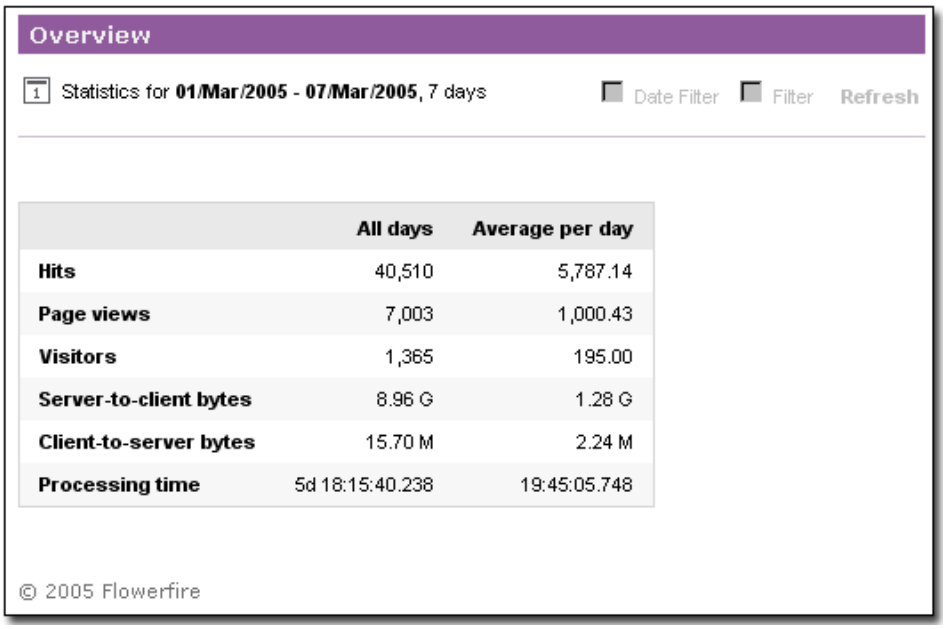
- The Overview
- Referrers
- The Session Overview
- The Session Views

The Overview

Overview

This view shows an overview of the traffic for the period indicated in **The Report Bar**. The traffic is broken down into hits, visitors, page views and bytes transferred (bandwidth).

For details of what Hits, Visitors, Page Views, Bandwidth and other terms used in this User Guide mean, see **Understanding the Reports**.

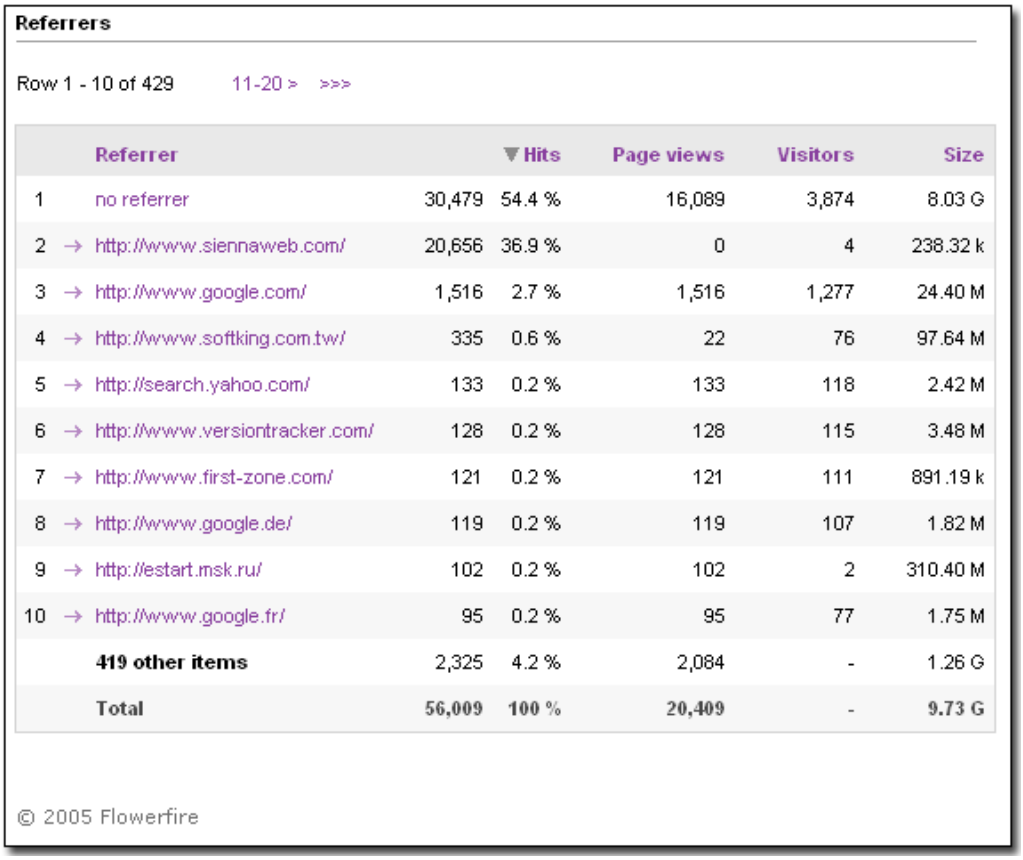


• User Guide Index

Where did my visitors come from?

Referrers

The URL of the previous webpage from which a link was clicked is known as Referrers. For instance, if a visitor first does a search on Google and then clicks on your entry in the search results, when that visitor arrives at your site, the referrer is Google for that session and Sawmill will report Google as the referring web site in the Referrers view.



Referrers

Row 1 - 10 of 429 11-20 > >>>

	Referrer	Hits		Page views	Visitors	Size
1	no referrer	30,479	54.4 %	16,089	3,874	8.03 G
2 →	http://www.siennaweb.com/	20,656	36.9 %	0	4	238.32 k
3 →	http://www.google.com/	1,516	2.7 %	1,516	1,277	24.40 M
4 →	http://www.softking.com.tw/	335	0.6 %	22	76	97.64 M
5 →	http://search.yahoo.com/	133	0.2 %	133	118	2.42 M
6 →	http://www.versiontracker.com/	128	0.2 %	128	115	3.48 M
7 →	http://www.first-zone.com/	121	0.2 %	121	111	891.19 k
8 →	http://www.google.de/	119	0.2 %	119	107	1.82 M
9 →	http://estart.msk.ru/	102	0.2 %	102	2	310.40 M
10 →	http://www.google.fr/	95	0.2 %	95	77	1.75 M
	419 other items	2,325	4.2 %	2,084	-	1.26 G
	Total	**56,009**	**100 %**	**20,409**	-	**9.73 G**

My site is the highest referrer, why?

Sawmill reports every referrer for every hit on your site. So the referrers table typically shows your web site as the highest referrer. For example, if someone had arrived at your site after clicking on the search results in Google, a Google page will be the referrer (the last page you are on is the referrer for the current page you are on). If your current page is a page on your site and the previous page is also a page on your site, then your site will be the referrer. As your visitors will visit more than one page on your site, your site will be the referrer far more than any other site.

Sawmill can be set up to remove your site as the referrer by categorizing your site as an "internal referrer", contact your Sawmill Administrator to do this.

How many people come direct to my site, rather than being referred?

Where visitors arrive at your site and do not have a referring page, Sawmill will categorize these as having "no referrer" and remove them from the reports, but this is quite a useful metric. If someone arrives without any referrer then it means they either clicked on a bookmark / email link or that they typed in your site address into the address bar of their browser directly; it reports how many visitors to your site actually know about you.

It is not always possible to get referrer information from the browser (some browsers can be configured to block this information) but the majority of browsers do allow it.

NOTE: This graphic shows the small links to the sites listed. To the left of each URL in the list is a small arrow, click this to visit the site.

- **User Guide Index**

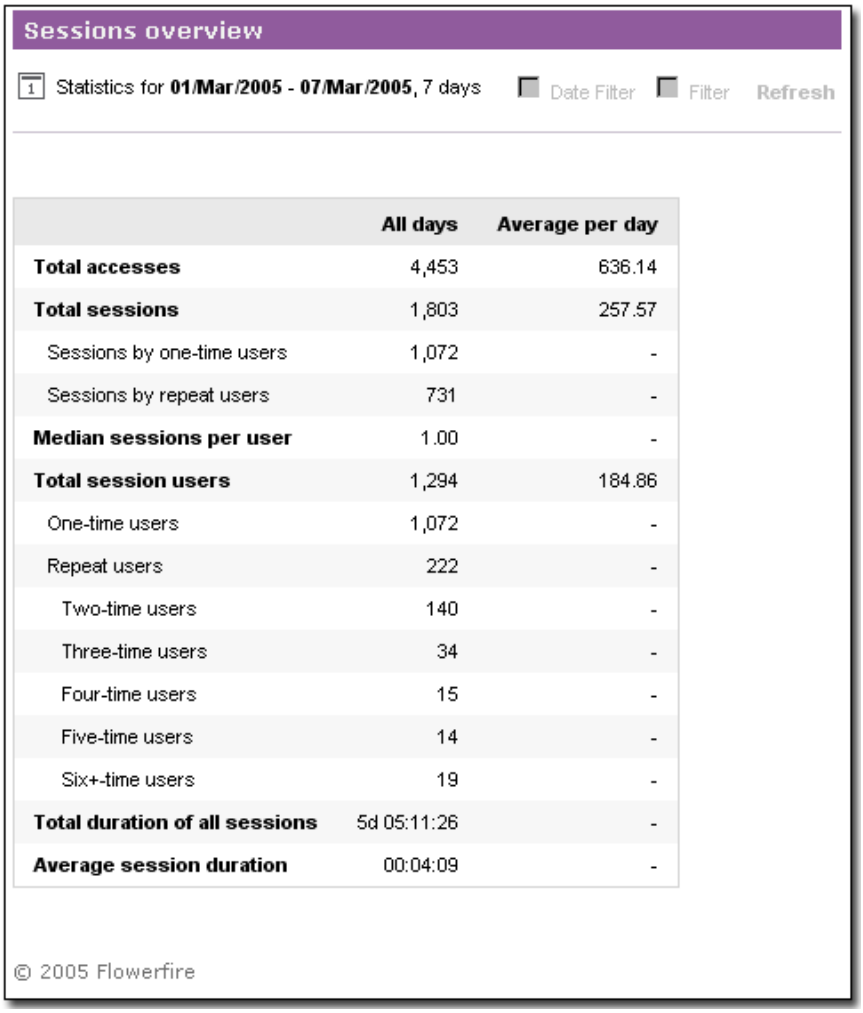
The Session Overview

• User Guide Index • An Overview of Sawmill • Reports • Filters • Understanding the Reports •

Sessions Overview

This view shows a summary of the "sessions" associated with the data.

For details of what sessions mean, see **What does Sawmill measure?**.



Sessions overview

Statistics for **01/Mar/2005 - 07/Mar/2005**, 7 days Date Filter Filter Refresh

	All days	Average per day
Total accesses	4,453	636.14
Total sessions	1,803	257.57
Sessions by one-time users	1,072	-
Sessions by repeat users	731	-
Median sessions per user	1.00	-
Total session users	1,294	184.86
One-time users	1,072	-
Repeat users	222	-
Two-time users	140	-
Three-time users	34	-
Four-time users	15	-
Five-time users	14	-
Six+-time users	19	-
Total duration of all sessions	5d 05:11:26	-
Average session duration	00:04:09	-

• User Guide Index

The Session Views

The Session Overview report is a summary of the main session views.

- **Entry and Exit pages:** These are the first and last pages of every session.
- **Paths through a Page:** Lets you select a single page on your site and see those pages that were visited prior to hitting that page, and those visited after. This gives you all routes to and from a given point in your site.
- **Session Paths:** Shows all the sessions (and the pages they hit) in a single expandable view.
- **Session Pages:** These are every occurrence of each page in any session. They are calculated by tracking how long it was from the hit on one page to the hit on the next page in that session (exit pages are considered to have zero time spent per page), and tabulating the results for all pages giving the time spent per page.
- **Session Users:** This lists all the users on your site and the number and duration of their visits.
- **Individual Sessions:** This lists all the sessions on your site.

- User Guide Index

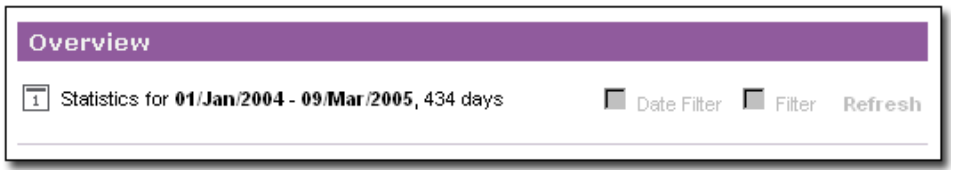
The Report Bar

• User Guide Index • An Overview of Sawmill • Reports • Filters • Understanding the Reports •

The Report Bar

The Report Bar shows the **Global Filters**, **Date/Time Filters** and **Zoom Filters** that are currently active.

Unfiltered Report: This is an example of what the Report Bar looks like for an unfiltered 'Overview' report:

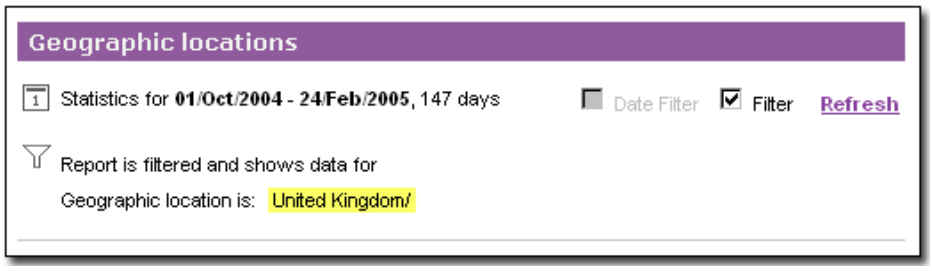


The icon indicates the date range the report covers. The "Date Filter" and "Filter" check boxes are greyed out because there are no date or global filters active.

Date/Time Filter: Here is an example of what the Report Bar looks like on the 'Overview' report with one of the **Date/Time Filters** in place (week filter):

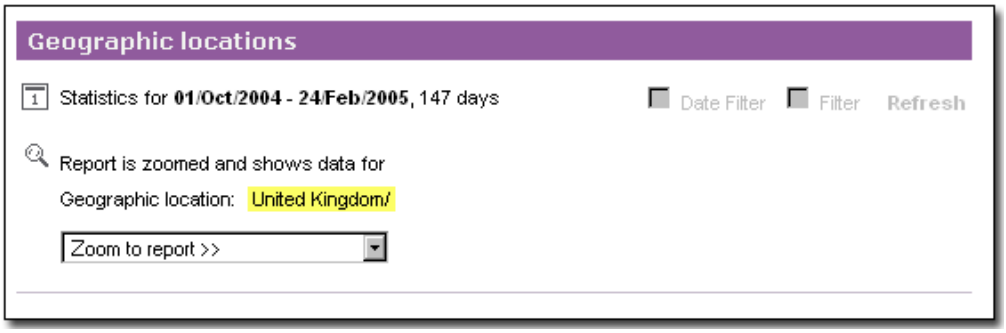
The icon indicates the date range the report covers and the "Date Filter" check box is no longer greyed out. You can check and uncheck this box and hit refresh to enable/disable the filter easily. The "Filter" check box remains greyed out because there is no active global filter.

Geographic location filter: This is an example of what the Report Bar looks like on the 'Geographic Locations' report with one of the **Global Filters** in place (geographic location filter of 'United Kingdom/'):



The icon indicates the date range the report covers, the icon indicates the global filter is in place and the "Filter" check box is no longer greyed out and you can check and uncheck this box and hit refresh to enable/disable the filter easily. The "Date Filter" check box remains greyed out because there is no active date filter.

Zoom filter: Here is an example of what the Report Bar looks like on the 'Geographic Locations' report with one of the **Zoom Filters** in place (geographic location filter of 'United Kingdom/'):



The icon indicates the date range the report covers, the icon indicates the zoom filter that is in place. Under the Zoom filter you can see **The Zoom To Menu**. This displays the reports you can Zoom to which will be filtered by the active Zoom filter. The "Date Filter" and "Filter" check boxes are greyed out because there are no date or global filters active.

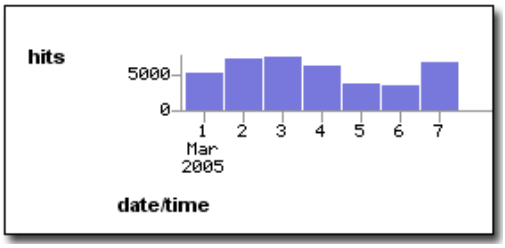
The zoom filters are discarded once you click on one of the reports in **The Report Menu**.

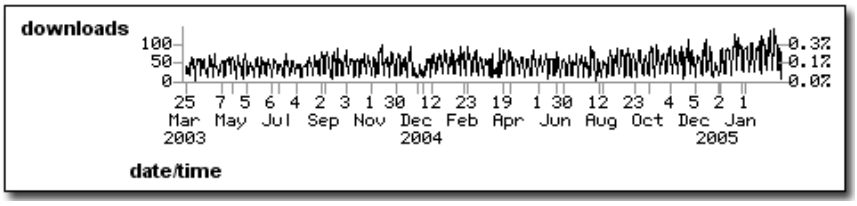
- **User Guide Index**

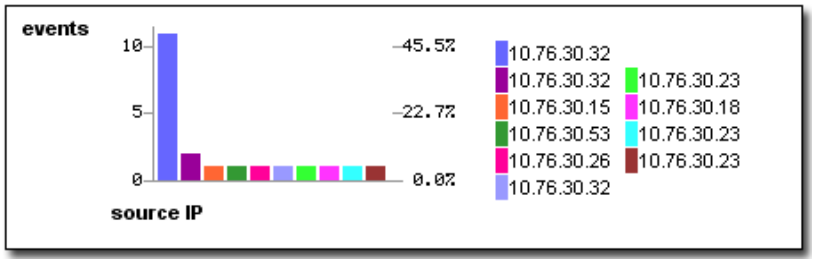
The Report Graph

The Report Graph

There are a variety of graphs that are visible in the reports, depending on the data in that report. Your Sawmill Administrator can control if a graph is visible, and which field is graphed. Below are some samples:





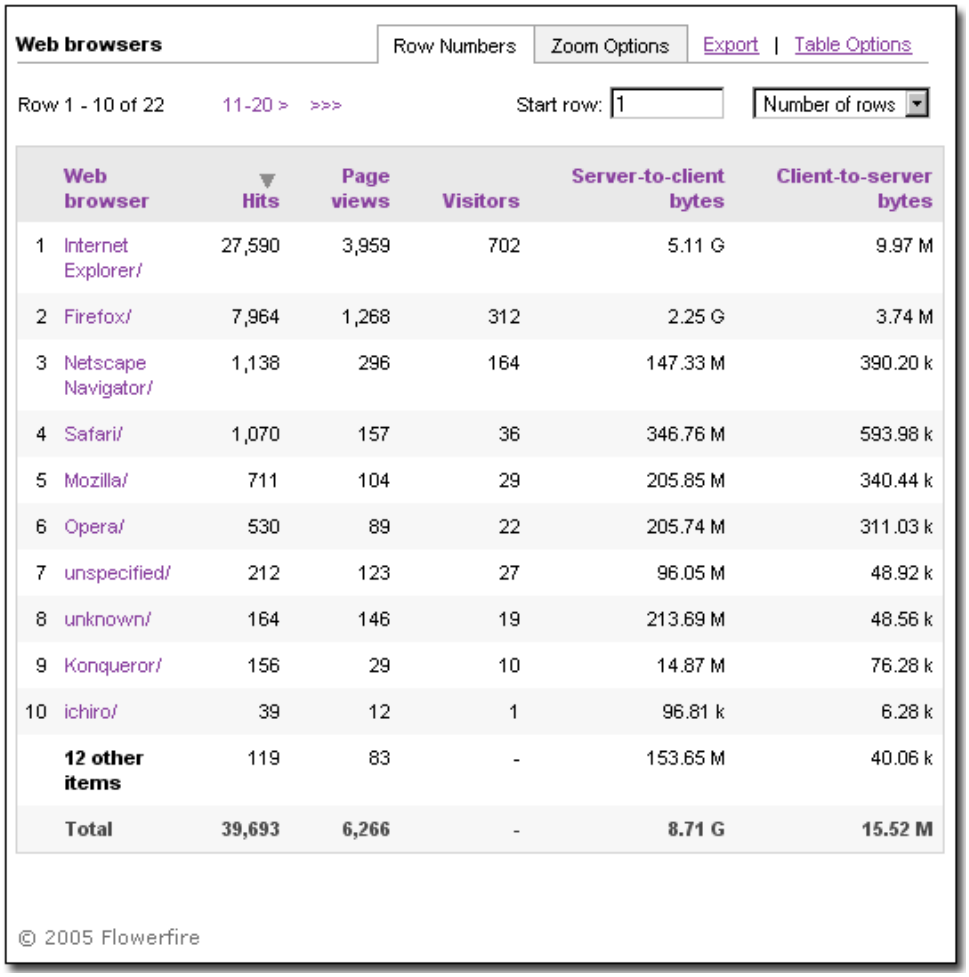


The Report Table

The Report Table

The Report Table contains the main information of the report. It displays one row per item, with the sum of hits/page views, ie. aggregated numerical values in columns next to it. How this table appears is controlled by the **Table Options**.

An example of the Web Browsers table is shown below.



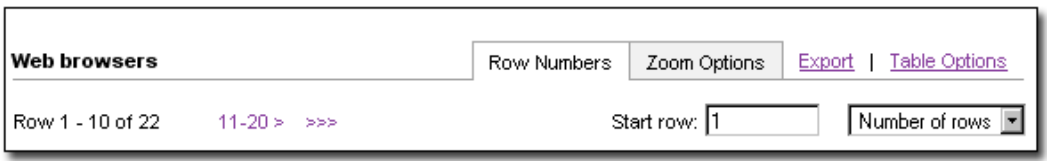
Web browsers Row Numbers | Zoom Options | Export | Table Options

Row 1 - 10 of 22 11-20 > >>> Start row: 1 Number of rows

	Web browser	Hits	Page views	Visitors	Server-to-client bytes	Client-to-server bytes
1	Internet Explorer/	27,590	3,959	702	5.11 G	9.97 M
2	Firefox/	7,964	1,268	312	2.25 G	3.74 M
3	Netscape Navigator/	1,138	296	164	147.33 M	390.20 k
4	Safari/	1,070	157	36	346.76 M	593.98 k
5	Mozilla/	711	104	29	205.85 M	340.44 k
6	Opera/	530	89	22	205.74 M	311.03 k
7	unspecified/	212	123	27	96.05 M	48.92 k
8	unknown/	164	146	19	213.69 M	48.56 k
9	Konqueror/	156	29	10	14.87 M	76.28 k
10	ichiro/	39	12	1	96.81 k	6.28 k
	12 other items	119	83	-	153.65 M	40.06 k
	Total	**39,693**	**6,266**	-	**8.71 G**	**15.52 M**

In some tables there are URLs (see the **Referrers** table for an example) and if these URLs are full; i.e., they contain the whole address like this - 'http://www.somedomain.com', then there will be a small arrow to the left of each line in the table that can be used to view that URL. Clicking the arrow will launch a new browser window, or new tab if you are using tabbed browsing, with that URL.

Just above the table is a bar that contains several different controls:



Web browsers Row Numbers | Zoom Options | Export | Table Options

Row 1 - 10 of 22 11-20 > >>> Start row: 1 Number of rows

- The **Row Numbers** detail (selected by default) can be used to change the starting row and the number of rows that are displayed in the table. There are also "paging" buttons that allow you to "page" through tables with multiple entries.
- The **Zoom Options** detail allows you to pre-select the view that will show when you use **Zoom Filters**.

- The **Export** link can be used to export data from the table.
- The **Table Options** link can be used to change which columns are visible, the sort order, and other aspects of the report. The sort order of any table can also be changed by clicking a column name; click once to sort by that column, or again to sort in reverse.

User Guide Index

Row Numbers

Row Numbers

The **Row Numbers** section above, from left to right shows you:

- The current row numbers displayed.
- The Paging function - The paging is done by selecting the next row set (<1-10 or 11-20>) or by skipping to the start or end by clicking (<<< or >>>).
- The Start row box - This allows you to enter and directly jump to any row and list from there.
- The Number of rows drop down menu - This allows you to see more rows than the default for this table.

- User Guide Index

Export

The Export Function

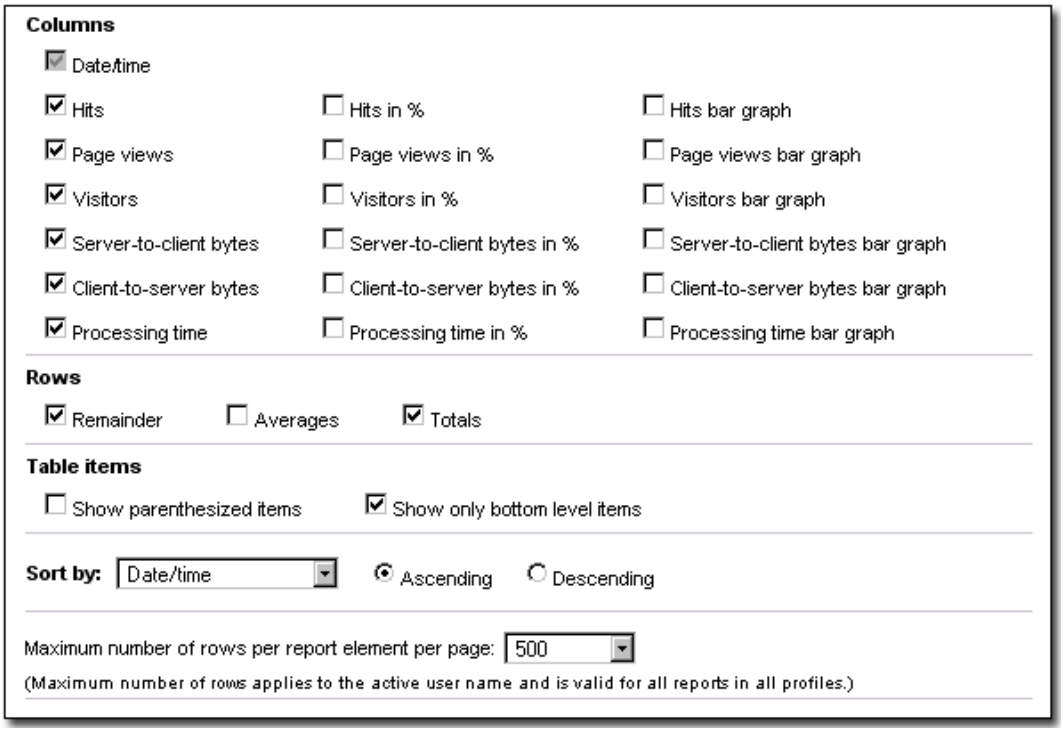
You can export data from Sawmill by clicking on the Export link from the table. A window will open and you can download the report from there. For large tables, you will see a "Generating Report" dialog first, then the window below will appear.

- User Guide Index

Table Options

Table Options

The Table Options controls which elements of the statistics are visible. Most elements are optional, and you can turn them on or off by selecting them from here.



Examples include the columns of tables; e.g., the hits and bandwidth rows, the special rows of tables; e.g., the total and averages rows, the pie charts, and more. The list of available elements changes depending on the view, the database structure, and other configuration options.

Most of the elements are self-explanatory, but others are explained here:

- **Show only bottom level items:** When this special element is "on", it changes the way the data is structured. Instead of showing the number of hits on a directory, this shows all of the files in the directory individually, with the number of hits on each.

On

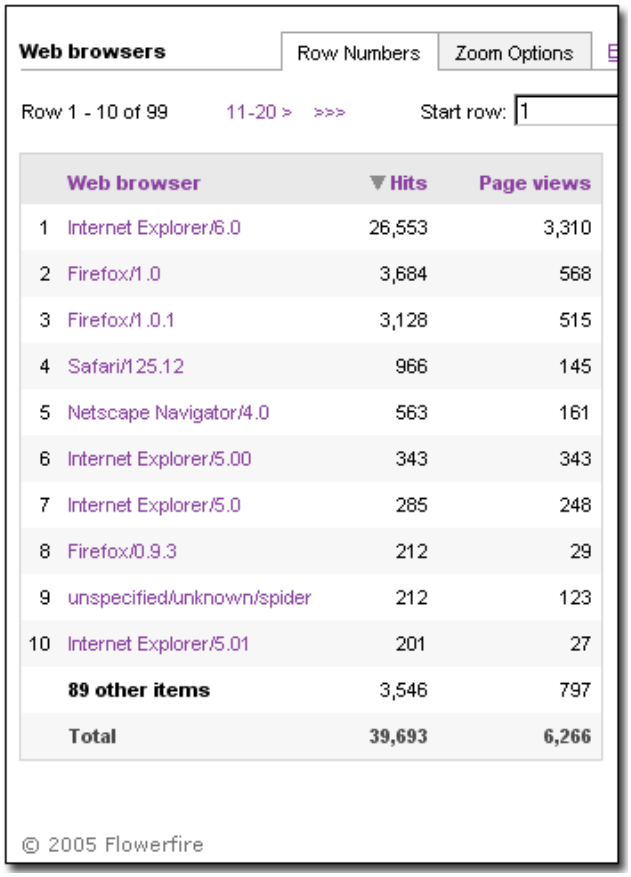
Off

Web browsers | Row Numbers | Zoom Optic

Row 1 - 10 of 22 11-20 > >>> Start row: 1

	Web browser	Hits	Page views
1	Internet Explorer/	27,590	3,959
2	Firefox/	7,964	1,268
3	Netscape Navigator/	1,138	296
4	Safari/	1,070	157
5	Mozilla/	711	104
6	Opera/	530	89
7	unspecified/	212	123
8	unknown/	164	146
9	Konqueror/	156	29
10	ichiro/	39	12
	12 other items	119	83
	Total	**39,693**	**6,266**

© 2005 Flowerfire



Web browsers | Row Numbers | Zoom Options | E

Row 1 - 10 of 99 11-20 > >>> Start row: 1

	Web browser	Hits	Page views
1	Internet Explorer/6.0	26,553	3,310
2	Firefox/1.0	3,684	568
3	Firefox/1.0.1	3,128	515
4	Safari/125.12	966	145
5	Netscape Navigator/4.0	563	161
6	Internet Explorer/5.00	343	343
7	Internet Explorer/5.0	285	248
8	Firefox/0.9.3	212	29
9	unspecified/unknown/spider	212	123
10	Internet Explorer/5.01	201	27
	89 other items	3,546	797
	Total	**39,693**	**6,266**

© 2005 Flowerfire

In the example above, we see the left table has neatly collected all the various browser versions together to give an overall total for that browser. On the right, all the versions are listed.

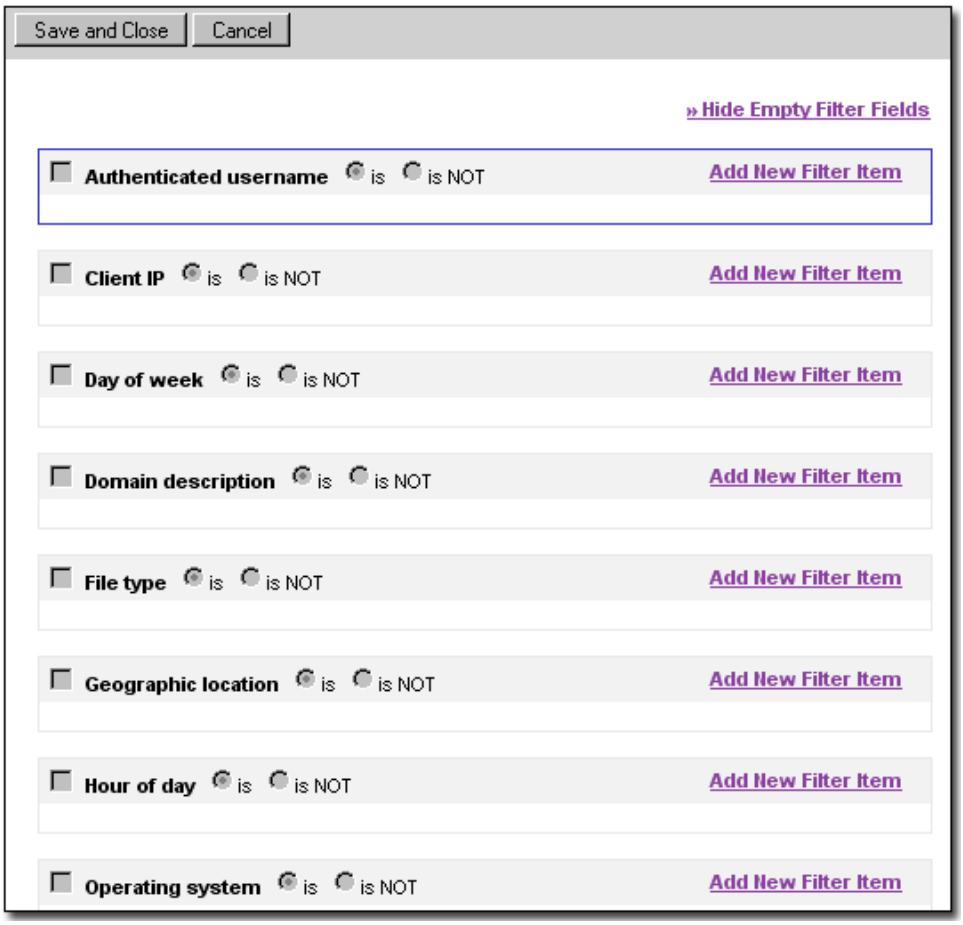
- **Show parenthesized items:** When this is shown, the items in the table that have parentheses (brackets) around them are shown. These include (empty) - where a field is empty and (internal referrer) where the referrer for a particular hit is the same site as the hit. There are some others.
- **Maximum number of rows:** This menu lets you control the number of rows displayed per table item per page. The default is 500 and if there are more than 500 items for a given table item, they will not be displayed.

User Guide Index

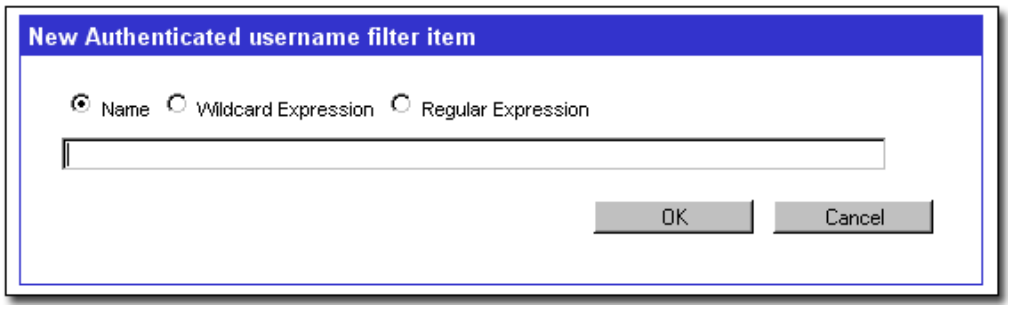
Global Filters

Global Filters

The Global Filters Editor lets you add, change, and remove filters from any field available in the database. To get to the Filter Editor, click on the  Icon in **The Report Toolbar**.



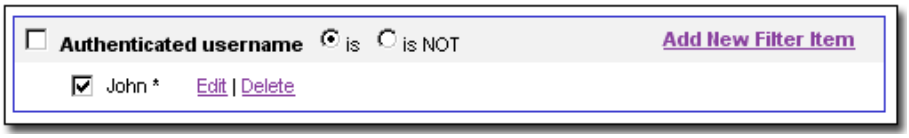
To filter one of the report views, click the **"Add New Filter Item"** link next to the field you want to filter. You will see this:



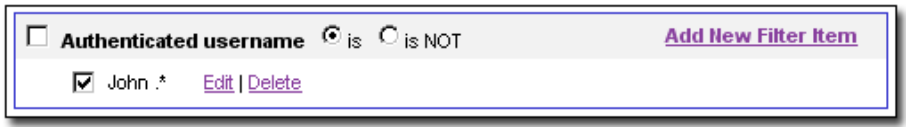
The three radio buttons are the three options for the way in which Sawmill can match what you want to see, to the report tables. Select the one you want to use, and type in the search string and click the OK button. These are examples of what you will see:

The **"Name"** filter will match exactly what you type in, so if you type in 'John Jones' as a Username, Sawmill will look for that string, and show you all the matches for it in the report table:

The **"Wildcard Expression"** filter will match using the standard Windows wildcard characters (* and ? amongst others), so if you type in 'John *' as a Username, Sawmill will return all usernames that start with 'John ':



The **"Regular Expression"** filter will match using UNIX Regular Expressions. Regular Expressions are more powerful and flexible than Wildcard Expressions and there are many online and printed resources on the subject. The topic of Regular Expressions is outside the scope of this User Guide, but try searching the internet for examples, or contact your Sawmill Administrator. In the following example, we are using 'John .*' as a match:



Once you have entered the search string and clicked the OK button, you can:

- Choose to turn this new filter on (or off if it is currently on) with the checkbox in the top left hand corner.
- Choose to use an 'is' or 'is NOT' match by selecting the radio buttons at the top.
- Add more filters.

The Global Filters Editor will save your filters even when they are not enabled, so that you can come back the next time you look at this profile and enable these filters again, without having to keep re-entering them. You can add multiple filters and enable some, none or all of them for any view. You can edit the current filters by clicking the 'edit' link and delete them with the 'delete' link.

Once complete, click on the "Save and Close" button to save your filter. Once you have Global Filters active, you will see the active filters in **The Report Bar** and the Filter checkbox becomes active. You can enable and disable all filters from here. NOTE: To enable the Global Filters from the report bar, the filters will need to be active from within the Global Filter Editor.

- **User Guide Index**

Date/Time Filters

Date/Time Filters

Date/Time filters are controlled by **The Calendar** and **The Date Range Selector**. They remain active on all Reports until they are removed by Clicking 'Show All' from within the Calendar.

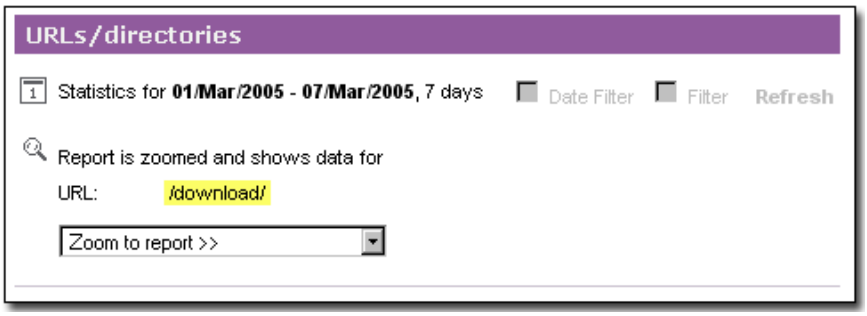
- **User Guide Index**

Zoom Filters

Zoom Filters

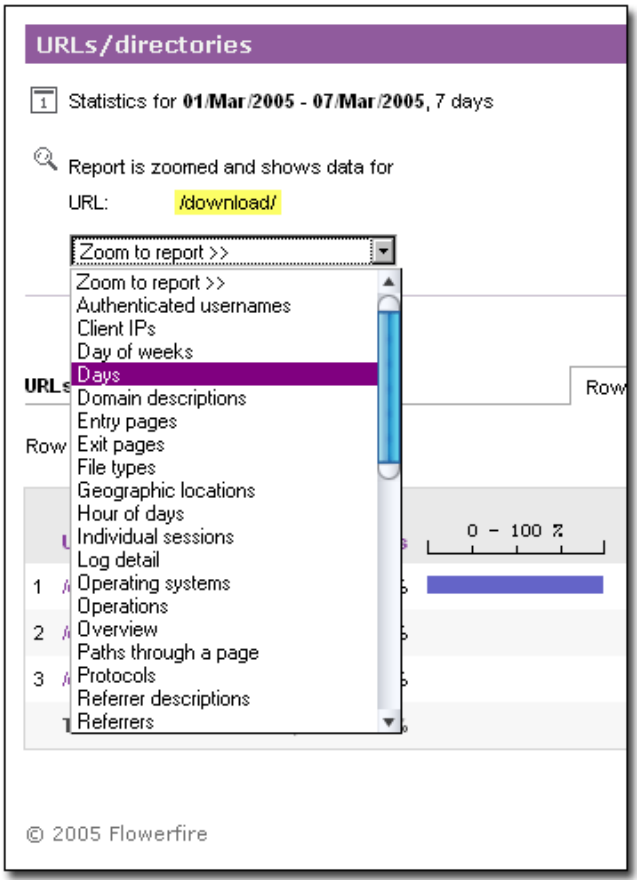
Zoom Filters are temporary filters that allow you to zoom about a report and see more detailed information. They are activated by clicking on a table item. The Zoom Display (part of **The Report Bar**) shows what you have zoomed in on.

For instance, if you've clicked on item '/download/' in the 'URLs/Directories' table, you will see the report bar display the Zoom filter like this:



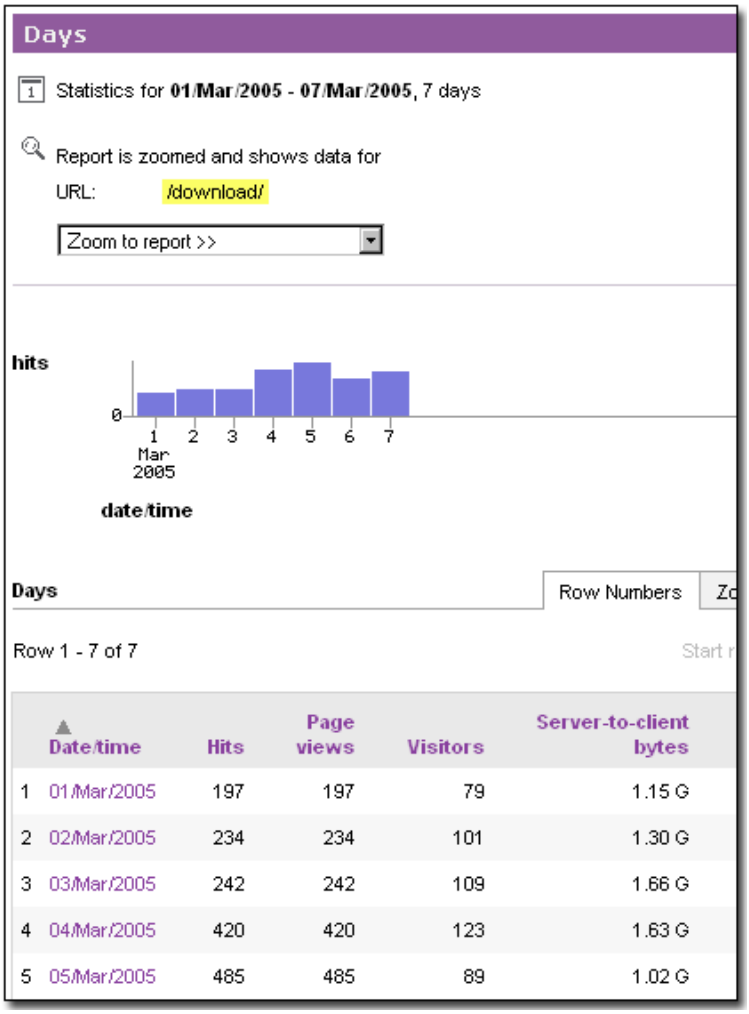
Zoom filters disappear when you click a new view in **The Report Menu**.

The Zoom To Menu shows the name of the report that will be displayed when you zoom. For instance, you can see that the 'Zoom to' Menu appears (in the screenshot below) when you select the /downloads/ directory in the table. By selecting a new Report View from the menu, you will Zoom into that new view and filter it by the selection '/download/'.



This can be useful if you want to break down each item in a table by some other report view, for instance, here we will see

what traffic looks like by day, for the directory '/download/'. To do this, choose 'URLs/Directories' view from the Report Menu, then click on '/download/' in the table and then 'Zoom to' the Days view.



	Date/time	Hits	Page views	Visitors	Server-to-client bytes
1	01/Mar/2005	197	197	79	1.15 G
2	02/Mar/2005	234	234	101	1.30 G
3	03/Mar/2005	242	242	109	1.66 G
4	04/Mar/2005	420	420	123	1.63 G
5	05/Mar/2005	485	485	89	1.02 G

You can change the Zoom menu view repeatedly, and filter each view from the Zoom to Menu by the '/download/' directory. You could also use **Global Filters** to add a report wide filter rather than using the Zoom Filters.

- **User Guide Index**

What does Sawmill measure?

Web Site Metrics

Sawmill can count web log traffic in several ways. Each way is counted independently of the others, and each has its own advantages in analyzing your traffic. The different types are:

Hits

Hits are events on your website. A single web page can be made up of many elements - it might have 10 images and a Flash movie (making 12 elements including the page). Each element would translate into an event. So if you had just one person visit only this one page, Sawmill would show 12 hits on your site. Hits could also be that a page was downloaded, or a PDF, or it could be just an image (one of many) downloaded on your front page. So if there are 5000 events for a single day, then Sawmill will report 5000 hits. Your Sawmill Administrator can help you remove traffic that you do not want to see in your reports.

Page views

A page view is a request to load a single page on a website. Page views correspond to hits on pages. For instance, a hit on /index.html is followed by 10 hits on image files, 1 style sheet, and 2 JavaScript files, that appear in the page. This will count as a single page view and 14 hits -- only the landing on /index.html will count in the Page Views total. By default, page views are all hits that are not GIF, JPEG, PNG, CCS, JS, and a few others. Your Sawmill Administrator can configure Sawmill for your website and the types of files you have.

Visitors

Visitors correspond roughly to the total number of people who visited the site (see **How Sawmill counts Visitors**). Every visitor is tracked by their unique IP address. By default, Sawmill defines visitors to be "unique hosts" -- a hit is assumed to come from a different visitor if it comes from a different hostname (client IP). For example, if a single person visits the site and looks at 100 pages, that will count as 100 page views, but only one visitor. This is not the most accurate way of measuring visitors due to the effects of web caches and proxy servers. Some web servers can track visitors using cookies, and if your web logs contain this information, Sawmill can use it instead of hostnames (contact your Sawmill Administrator for more information). It is also possible for Sawmill to use WebNibbler™ Visitor Tracking. Contact your Sawmill Administrator, or see **The Sawmill Web Site** for details.

Bandwidth

Bandwidth is the total number of bytes transferred. Bandwidth is tracked for every event that occurs, whether it is accepted as a "hit" or as a "page view". For log formats that track both inbound and outbound bandwidth, Sawmill can report both simultaneously.

Referrers

Referrers are "where visitors came from". For instance, if a visitor first does a search on Google and then clicks on your link in the search results then, when that visitor arrives at your site, the referrer is Google for that session and Sawmill will report Google as the referring web site in the Referrers view. For more detail, see **Where did my visitors come from?**

Sessions

Several of Sawmill's reports deal with "session" information, including the **The Session Overview** and the "paths (clickstreams)" report. Sessions are similar to visitors, except that they can "time out." When a visitor visits the site, and then leaves, and comes back later, it will count as two sessions, even though it's only one visitor. To reduce the effect of caches that look like very long sessions, Sawmill also discards sessions longer than a specified time. To change the timeout interval, contact your Sawmill Administrator.

How Sawmill counts Visitors

• User Guide Index • An Overview of Sawmill • Reports • Filters • Understanding the Reports •

The Visitor Totals

Question: When I add up the number of visitors on each day of the month, and I compare it to the total number of visitors for the month, they're not equal. Also, why doesn't the sum of visitors on subpages/subdirectories add up to the total for the directory, and why doesn't the sum of visitors on subdomains add up to the total for the domain?

Answer: "Visitors" in Sawmill's terminology refers to unique visitors (see **What does Sawmill measure?**).

Therefore:

- The total hits in a month is equal to the sum of the hits on the days of the month.
- The total bandwidth for a month is equal to the sum of the bandwidth on the days of the month.
- The total page views for a month is equal to the sum of the page views for each day of the month.

BUT

- The total number of visitors in a month is not usually equal to the sum of the visitors on the days of the month.

To explain these differences, suppose you have a website where only one person ever visits the site, but that person visits it every day. For every day of the month, you will have a single visitor but for the month you will have a single visitor as well, because visitors are unique visitors, and there was only one unique visitor in the entire month.

If what you're really looking for is "visits" rather than "visitors" (so each visit will count once, even if it's the same visitor coming back over and over), then that's what Sawmill calls "sessions," and you can get information about them in **The Session Overview** and **How Sawmill calculates Sessions**.

• User Guide Index

How Sawmill calculates Sessions

How Does Sawmill Compute Session Information?

Sawmill computes session information by tracking the page, date/time, and visitor id, which is usually the hostname (client IP), for each page view. When a session view is requested, it processes all of these page views at the time of the request.

Sawmill groups the hits into initial sessions based on the visitor id-- it starts by assuming that each visitor contributed one session. It sorts the hits by date so it has a click-by-click record of the movement of each visitor.

Then it splits the sessions, using the session timeout interval. This is set to 30 minutes by default but your Sawmill Administrator can change this. Since most websites are accessible without logging in, there is no way for Sawmill to know the real time that a user leaves the site; it can only guess by assuming that if they didn't click anything on the site for 30 minutes, they must have finished and left. By splitting the sessions up in this way we are counting more accurately the number of sessions a given visitor has had on the website. It is not a perfect science, but over time, as long as the method of measuring remains the same, a trend can be found. This splitting process increases the number of sessions that Sawmill counts, resulting in possibly more than one session per visitor.

Next, Sawmill discards sessions over 2 hours long (again this can be changed by the Sawmill Administrator). Most web sessions are considerably shorter than that, so there's a good chance that any really long session is actually caused by multiple visitors using the same proxy server to visit the site. These look like one long session because all of the hits seem to come from the proxy server (the same IP address). Sawmill rejects these because there is no way to tell which hits were from a particular visitor. If you're using visitor cookies to track unique visitors, this will not be a problem as your visitor count will not be covered by the hostname (Client IP) of the visitor.

Finally, Sawmill discards sessions based on the **Filters**.

After that, Sawmill is ready to generate the statistics reports.

How Sawmill calculates Durations

How Sawmill Calculates "Durations"

User 'A' visits a current affairs/news website and spends one minute browsing and then settles on a news item, spending seven minutes reading it then browsing for another two minutes and then reading the next news item for a further eight minutes, then exiting by closing the browser. Here is the time line:

- 11/04/2005 16:12 - arrive at homepage and navigate to a news item
- 11/04/2005 16:13 - read news item 1
- 11/04/2005 16:20 - navigate to next news item
- 11/04/2005 16:22 - read news item 2
- 11/04/2005 16:30 - exit (close down browser - no further activity for 30 mins)

With the session timeout interval of 30 minutes (this can be changed by the Sawmill administrator) Sawmill will count 1 session with a 10 minute duration, with the 2 news item pages having durations of 7 & 8 minutes respectively.

This calculation shows the problem of duration counting in all log analysis software. Since there is no reporting of the exact time the user left the site (there is no log entry for the 16.30 'exit' as closing the browser does not leave any record in the server logs), we have no way of knowing when the user exited, so we count from the last known time, which in this case, is the start of reading the second news item, 16:22.

Applying what you have learned to your web site

Using the information Sawmill reports to you, you can track and improve the effectiveness of your site, to bring more visitors and increase your sales.

Sawmill's Reports Will Help You Answer These Important Questions:

What is bringing traffic to the site?

Check the **Referrers** and Search Engines reports. These two reports list which web sites referred your visitors to your site and also lists the Search Engines which were used to find your site.

Which ads are effective?

A good approach is to use special URLs in your ad campaigns so that you understand which ads are bringing traffic to your site. For each ad, use the URL of your home page but add a question mark and some information about the ad. For instance, if you run an online advertising campaign in Europe on the search engine Google, you might use the URL "**www.mydomain.com/homepage.html?source=google&ad=euro1**". Then in Sawmill, you can look at the Pages or URL's view and filter out pages that do not contain "source=google" by using the **Global Filters** and then Sawmill will show you the traffic generated by your Google ads. See **Tracking Ad Campaigns** for an example.

Which search engines are working?

Check the Search Engines report for how much traffic is generated by each search engine. Those without any entries could be where to start a new campaign. Do some searches on the Search Engine yourself to see if or where your entry appears. If you are spending more on one search engine but it is producing less traffic, you will need to determine why.

What are people searching for in the search engines-- should you change the text of your pages to get higher placement?

If you review the Search Phrases report, this will tell you what visitors used in the search engine to find your site. Compare this to the Words you use in your Web Pages, consider using http "meta" tags on your pages with some keywords - your Web Developer will be able to help with the use of meta tags.

Why are people leaving the site once they get there?

Here are some questions that you can ask as you review the reports: Is there a specific page which is driving them away? Can it be improved to keep them? Are they leaving as soon as they get to the top page? If so, maybe the site needs to be more attractive or interesting. Are they searching on your site for something, not finding it, and then leaving? Maybe you should add that to what you are offering. Are they getting to the pricing page and then leaving? You might need to reconsider your pricing and make it more competitive.

Who buys from your site?

If you're selling something, which domains place orders? Should you be focusing your marketing on that domain? Is the site easy to navigate? Are the paths people taking reasonable? Are they getting lost, or getting in loops? Maybe the site should be streamlined so that people can get to the pages they want more easily.

What content works or doesn't?

Are visitors looking at the pages you didn't spend much time working on, or ignoring the pages you thought were the most important? Maybe you should reconsider your priorities, and work on the content people want.

Do we have enough bandwidth, and a fast enough server?

Sawmill can show you the total hits and bandwidth usage of your sites, so you can decide when it's time to upgrade your connection or buy a new server. By upgrading your connection or server before it becomes a problem, you can keep your visitors happy.

Sawmill Can Help You Solve Important Problems.

Here are some scenarios where Sawmill can help:

You see in the statistics that numerous hits came from Spain yesterday. Want to know what they were looking at? Just click on the Geographic Locations view and then click on Spain in that table, this 'Zooms' you in on hits from Spain. From **The Zoom To Menu** switch to the "Pages" report, and you see all the pages hit by people from Spain, in two clicks.

You're trying to see if a particular page got a hit yesterday. The statistics show the top twenty pages, but you want to see *all* the pages; the page you're looking for only got one hit, or none, so it'll be way down in the ranking. You can do one of three things:

- Choose the **Row Numbers** feature to 'page' to the last row in the table and 'page' back through the table until you find it.
- Click on 'All Rows' from the 'number of rows' drop down menu and the table will expand to show *all* the pages that ever got any hits, even if there are 500,000 of them.

You want to see which paths visitors took through the site, click-by-click. At each page along the way you want to see which visitors left, and where the other ones went next. Sawmill's "session paths" report lets you see the complete click-by-click details of *every* visit. Starting with the entry pages, you can click to expand any part of the data, to see whatever you need to see about your visitors and the paths they took.

You want to see a graph of the web site traffic for last November. You want to see the bandwidth for November. Actually you just want to see November 16th. Wow, look at that spike around 3pm; where did all those hits come from? Sawmill can do all this and much more. This is how you'll think when you use Sawmill. Look at the November hits graph, wonder about the bandwidth; choose "bandwidth bar graph" from the menu. You have another graph showing bandwidth where there used to be just a hits graph. See a lot of hits on the 16th? Click the November 16 bar in the graph and the graph changes to show only traffic on November 16, hour-by-hour. Wondering where that spike came from on some particular hour? Three clicks and you can see the traffic for that hour only, broken down by domain, so you can see who it was. Or broken down by page, so you can see what they looked at. Or broken down by referring URL or domain, so you can see which ad, search engine, or web page brought you all that traffic!

- **User Guide Index**

All Options

Command Line Options

Option name	Description
Session ID	Internal option used to track sessions in the graphical interface Command-line name: `command_line.session_id` Command-line shortcut: `si`
Action	The command-line action to perform Command-line name: `command_line.action` Command-line shortcut: a
Generate HTML report files to folder	Generate HTML report files into a folder Command-line name: `command_line.generate_html_to_directory` Command-line shortcut: `ghtd`
Field name	The name of a database field Command-line name: `command_line.field_name` Command-line shortcut: `fn`
Start web server	This option controls whether Sawmill starts its built-in web server when it starts (whether it runs in web server mode) Command-line name: `command_line.web_server` Command-line shortcut: `ws`
Generate report	Generate a report Command-line name: `command_line.generate_report` Command-line shortcut: `gr`
Statistics viewing password	The password required to view the statistics for this profile, passed on the command line Command-line name: `command_line.password` Command-line shortcut: `clp`
Profile to use	The name of the profile to use for this command Command-line name: `command_line.profile` Command-line shortcut: p
Command-line output types	The types of command-line output to generate Command-line name: `command_line.verbose` Command-line shortcut: v
Master process ID	Process ID of the master web server thread (used internally) Command-line name: `command_line.master_process_id` Command-line shortcut: `mpi`
Display page	The interface page to display Command-line name: `command_line.display_page` Command-line shortcut: `dp`
Report name	The name of the report to generate Command-line name: `command_line.report_name` Command-line shortcut: `rn`
Return address	The return address of an email message Command-line name: `command_line.return_address` Command-line shortcut: `rna`
Recipient address	The recipient address of an email message Command-line name: `command_line.recipient_address` Command-line shortcut: `rca`

SMTP server	The SMTP server to use to send email Command-line name: `command_line.smtp_server` Command-line shortcut: `ss`
Report email subject	The email subject to use when sending a report by email Command-line name: `command_line.report_email_subject` Command-line shortcut: `res`
Merge database directory	Directory of database to merge into this one Command-line name: `command_line.merge_database_directory` Command-line shortcut: `mdd`
Report filters	The filters to use when generating a report Command-line name: `command_line.filters` Command-line shortcut: `f`
Date filter	The date filter to use when generating a report Command-line name: `command_line.date_filter` Command-line shortcut: `df`
Date breakdown	Zooms to the date which is specified in "Date filter" Command-line name: `command_line.date_breakdown` Command-line shortcut: `db`
Update to version	The version to update to, using web update Command-line name: `command_line.update_to_version` Command-line shortcut: `utv`
Expand session paths	This option controls whether session paths should be expanded while generating a session paths report Command-line name: `command_line.expand_paths` Command-line shortcut: `ep`
Generate PDF friendly files	Generate HTML report files in a PDF friendly format Command-line name: `command_line.generate_pdf_friendly_files` Command-line shortcut: `gpff`
Zoom value	The zoom value to zoom into a hierarchical report. Command-line name: `command_line.zoom_value` Command-line shortcut: `zv`
Ending row	This option controls the ending row of a generated or exported report. Command-line name: `command_line.ending_row` Command-line shortcut: `er`
Total row	Adds a total row when used with export_csv_table. Command-line name: `command_line.total_row` Command-line shortcut: `tr`
Output filename	Exports a report to the specified filename when used with export_csv_table. Command-line name: `command_line.output_filename` Command-line shortcut: `of`
End of line	Specifies the end of line marker in a CSV file. Command-line name: `command_line.end_of_line` Command-line shortcut: `eol`

Preferences

Option name	Description
Never look up IP numbers using domain name server	Whether to ever try to look up hostnames of IP-numbered hosts Command-line name: `preferences.miscellaneous.never_look_up_ip_numbers` Command-line shortcut: `nluin`

Only look up IP numbers for log entries	Look up IP numbers only when they appear in logs, not for local server or remote browsing computer Command-line name: `preferences.miscellaneous.only_look_up_log_ip_numbers` Command-line shortcut: `olulin`
Logout URL	The URL to go to on logout; if empty, goes to login screen Command-line name: `preferences.miscellaneous.logout_url` Command-line shortcut: `lu`
Temporary files lifespan	Amount of time to keep temporary files before deleting them (in seconds) Command-line name: `preferences.miscellaneous.temporary_files_lifespan` Command-line shortcut: `tfl`
Language	The language module to use to generate language-specific text Command-line name: `preferences.miscellaneous.language` Command-line shortcut: `l`
Charset	The HTML charset to use when displaying pages Command-line name: `preferences.miscellaneous.charset` Command-line shortcut: `c`
Prompt for trial tier	Whether to prompt for Professional/Enterprise switch during trial period Command-line name: `preferences.miscellaneous.prompt_for_trial_tier` Command-line shortcut: `pftt`
Security Mode	The level of security to use Command-line name: `preferences.security.security_mode` Command-line shortcut: `sm`
Trusted hosts	The hostnames of computers which are "trusted," and do not need to enter passwords Command-line name: `preferences.security.trusted_hosts` Command-line shortcut: `th`
Show full operating system details in errors	Show full operating system version details in the text of error messages Command-line name: `preferences.security.show_full_operating_system_details_in_errors` Command-line shortcut: `sfosdie`
Authentication command line	The command line to run to authenticate users Command-line name: `preferences.security.authentication_command_line` Command-line shortcut: `acl`
Default permissions	The permissions Sawmill uses when creating a file or folder (chmod-style) Command-line name: `preferences.security.default_permissions`
Database file permissions	The permissions Sawmill uses when creating a file as part of a database Command-line name: `preferences.security.database_file_permissions`
Database folder permissions	The permissions Sawmill uses when creating a folder as part of a database Command-line name: `preferences.security.database_directory_permissions`
Profile file permissions	The permissions Sawmill uses when creating a profile file Command-line name: `preferences.security.profile_permissions`
Profile folder permissions	The permissions Sawmill uses when creating a folder containing profile files Command-line name: `preferences.security.profiles_directory_permissions`
Temporary profile file permissions	The permissions Sawmill uses when creating a temporary profile file Command-line name: `preferences.security.temporary_profile_permissions`
Default profile file permissions	The permissions Sawmill uses when creating the default profile file Command-line name: `preferences.security.default_profile_permissions`
Password file permissions	The permissions Sawmill uses when creating the password file Command-line name: `preferences.security.password_file_permissions`
Image file permissions	The permissions Sawmill uses when creating an image file Command-line name: `preferences.security.image_file_permissions`

Image folder permissions	The permissions Sawmill uses when creating a folder containing image files Command-line name: `preferences.security.image_directory_permissions`
Server folder permissions	The permissions Sawmill uses when creating the server folder Command-line name: `preferences.security.server_directory_permissions`
LogAnalysisInfo folder permissions	The permissions Sawmill uses when creating the LogAnalysisInfo folder Command-line name: `preferences.security.log_analysis_info_directory_permissions`
Show only the profiles matching REMOTE_USER	Whether to show only the profiles whose names start with the value of REMOTE_USER Command-line name: `preferences.security.show_only_remote_user_profiles` Command-line shortcut: `sorup`
Administrative REMOTE_USER	The value of REMOTE_USER which marks that user as administrator Command-line name: `preferences.security.administrative_remote_user` Command-line shortcut: `aru`
LogAnalysisInfo folder location	A folder where Sawmill can store profiles and other information Command-line name: `preferences.server.log_analysis_info_directory` Command-line shortcut: `laid`
Temporary folder	A folder on the web server running Sawmill as a CGI program, from which images can be served Command-line name: `preferences.server.temporary_directory_pathname` Command-line shortcut: `tdp`
Temporary folder URL	The URL of a folder on the web server running Sawmill as a CGI program, from which images can be served Command-line name: `preferences.server.temporary_directory_url` Command-line shortcut: `tdu`
Web server port	The port to listen on as a web server Command-line name: `preferences.server.web_server_port` Command-line shortcut: `wsp`
Maximum simultaneous tasks	Maximum number of simultaneous web tasks (threads of execution) that Sawmill will perform Command-line name: `preferences.server.maximum_number_of_threads` Command-line shortcut: `mnot`
CGI folder	The folder containing Sawmill, relative to the server root Command-line name: `preferences.server.cgi_directory` Command-line shortcut: `cd`
Maximum CPU usage	Percent of CPU time to use while processing log data Command-line name: `preferences.server.maximum_cpu_usage_percent` Command-line shortcut: `mcup`
Web server IP address	The IP address to run Sawmill's web server on Command-line name: `preferences.server.server_hostname` Command-line shortcut: `sh`
Cookie domain	The domain to use in cookies sent to the browser Command-line name: `preferences.server.cookie_domain`

Profile Options

Option name	Description
Automatically update database when older than	Automatically update the database when the statistics are viewed and the database has not been updated in this many seconds Command-line name: `database.options.automatically_update_when_older_than` Command-line shortcut: `auwot`
Database folder	The location of the database Command-line name: `database.options.database_directory` Command-line shortcut: `dd`

Prompt before erasing database	True if Sawmill should prompt before erasing a non-empty database Command-line name: `database.options.prompt_before_erasing_database` Command-line shortcut: `pbed`
Build all indices simultaneously	Build all indices simultaneously after processing log data, for better performance Command-line name: `database.tuning.build_all_indices_simultaneously` Command-line shortcut: `bais`
Build all cross-reference tables simultaneously	Build all cross-reference tables simultaneously after processing log data, for better performance Command-line name: `database.tuning.build_all_xref_tables_simultaneously` Command-line shortcut: `baxts`
Build indices during log processing	Build indices on the fly while log data is read, rather than in a separate stage Command-line name: `database.tuning.build_indices_during_log_processing` Command-line shortcut: `bidlp`
Build cross-reference tables and indices simultaneously	Build cross-reference tables and indices simultaneously after processing log data, for better performance Command-line name: `database.tuning.build_xref_tables_and_indices_simultaneously` Command-line shortcut: `bxtais`
Build cross-reference tables during log processing	Build cross-reference tables on the fly while log data is read, rather than in a separate stage Command-line name: `database.tuning.build_xref_tables_during_log_processing` Command-line shortcut: `bxtdlp`
Build indices in threads	Build indices in threads, and merge them at the end Command-line name: `database.tuning.build_indices_in_threads` Command-line shortcut: `biit`
Build indices in memory	Build indices in memory, rather than using memory-mapped files on disk Command-line name: `database.tuning.build_indices_in_memory` Command-line shortcut: `biim`
Build cross-reference tables in threads	Build cross-reference tables in threads, and merge them at the end Command-line name: `database.tuning.build_xref_tables_in_threads` Command-line shortcut: `bxtit`
Expansion factor for database table	Factor by which a hash table expands when necessary Command-line name: `database.tuning.hash_table_expansion_factor` Command-line shortcut: `htef`
Initial size of database table	Initial size of a database hash table Command-line name: `database.tuning.hash_table_starting_size` Command-line shortcut: `htss`
Surplus factor for database table	Number of times larger a hash table is than its contents Command-line name: `database.tuning.hash_table_surplus_factor` Command-line shortcut: `htsf`
List cache size	Maximum memory used by the list cache Command-line name: `database.tuning.list_cache_size` Command-line shortcut: `lcs`
Maximum main table segment size to merge	Maximum size of main table segment to merge; larger segments will be copied Command-line name: `database.tuning.maximum_main_table_segment_merge_size` Command-line shortcut: `mmtsms`
Maximum main table segment size	The maximum size of one segment of main database table Command-line name: `database.tuning.maximum_main_table_segment_size` Command-line shortcut: `mmtss`
Maximum xref segment size to merge	Maximum size of a cross-reference table segment to merge; large segments will be copied Command-line name: `database.tuning.maximum_xref_segment_merge_size` Command-line shortcut: `mxsms`
Maximum cross-reference table segment size	The maximum size of one segment of a cross-reference database table Command-line name: `database.tuning.maximum_xref_table_segment_size` Command-line shortcut: `mxtss`

Allow newlines inside quotes	Allow newlines (return or line feed) inside quotes, in log lines Command-line name: `log.format.allow_newlines_inside_quotes` Command-line shortcut: `aniq`
Apache log format description string	A string which describes the log format, Apache-style Command-line name: `log.format.apache_description_string` Command-line shortcut: `ads`
Autodetect lines	Number of lines to examine for this log format while auto-detecting Command-line name: `log.format.autodetect_lines` Command-line shortcut: `al`
Autodetect regular expression	A regular expression used to auto-detect the log format Command-line name: `log.format.autodetect_regular_expression` Command-line shortcut: `are`
Autodetect expression	An expression used to auto-detect the log format Command-line name: `log.format.autodetect_expression` Command-line shortcut: `ae`
Blue Coat log format description string	A string which describes the log format, Blue Coat style Command-line name: `log.format.blue_coat_description_string` Command-line shortcut: `bcds`
Format is Common Log Format	Log format is similar to Common Log Format Command-line name: `log.format.common_log_format` Command-line shortcut: `clf`
Log field separator	The character or string that separates one log field from the next Command-line name: `log.format.field_separator` Command-line shortcut: `fs`
Date format	Format of dates in the log data Command-line name: `log.format.date_format` Command-line shortcut: `ldf`
Default log date year	The year to use, e.g., 2004, if the date format in the log data has no year information Command-line name: `log.format.default_log_date_year` Command-line shortcut: `dldy`
Log data format	The format of the log data Command-line name: `log.format.format_label` Command-line shortcut: `fl`
Global date regular expression	A regular expression which, if matched in the log data, determines the date for all subsequent entries Command-line name: `log.format.global_date_regular_expression` Command-line shortcut: `gdre`
Global date filename regular expression	A regular expression which, if matched in the log filename, determines the date for all entries in that log file Command-line name: `log.format.global_date_filename_regular_expression` Command-line shortcut: `gdfre`
Ignore format lines	Ignore format lines in the log data Command-line name: `log.format.ignore_format_lines` Command-line shortcut: `ifl`
Ignore quotes	Ignore quotes in log data Command-line name: `log.format.ignore_quotes` Command-line shortcut: `iq`
Parse log only with log parsing filters	Use only the parsing filters to parse the log (and not the log format regexp, index/subindex, etc.) Command-line name: `log.format.parse_only_with_filters` Command-line shortcut: `powf`
Log data format regular expression	A regular expression describing the log format Command-line name: `log.format.parsing_regular_expression` Command-line shortcut: `pre`

Time format	Format of times in the log data Command-line name: `log.format.time_format` Command-line shortcut: `ltf`
Treat square brackets as quotes	Treat square brackets as quotes Command-line name: `log.format.treat_brackets_as_quotes` Command-line shortcut: `tbaq`
Treat apostrophes (') as quotes	Treat apostrophes (') as quotes Command-line name: `log.format.treat_apostrophes_as_quotes` Command-line shortcut: `taaq`
Allow empty log source	True if Sawmill should allow databases to be created from log sources which contain no data Command-line name: `log.processing.allow_empty_log_source` Command-line shortcut: `aels`
Date offset	The number of hours to add to each date in the log file Command-line name: `log.processing.date_offset` Command-line shortcut: `do`
Log entry pool size	The number of log entries Sawmill can work on simultaneously Command-line name: `log.processing.log_entry_pool_size` Command-line shortcut: `eps`
Log reading block size	Size in bytes of the blocks which are read from the log Command-line name: `log.processing.read_block_size` Command-line shortcut: `rbs`
Skip processed files on update (by pathname)	Skip files which have already been processed, judging by their pathnames, during a database update or add operation Command-line name: `log.processing.skip_processed_filenames_on_update` Command-line shortcut: `spfod`
Thread data block size	Command-line name: `log.processing.thread_data_block_size` Command-line shortcut: `tdbs`
Log processing threads	The number of simultaneous threads to use to process log data Command-line name: `log.processing.threads` Command-line shortcut: `lpt`
Convert log data charset	Command-line name: `log.processing.convert_log_data_charset` Command-line shortcut: `cldc`
Convert log data from charset	The charset to convert from, when converting input log data Command-line name: `log.processing.convert_log_data_from_charset` Command-line shortcut: `cldfc`
Convert log data to charset	The charset to convert to, when converting input log data Command-line name: `log.processing.convert_log_data_to_charset` Command-line shortcut: `cldtc`
Actions email address (es)	The address(es) that Sawmill should send email to whenever an action completes; e.g., the database is built Command-line name: `network.actions_email_address` Command-line shortcut: `aea`
DNS Server	The hostname or IP address of the DNS server to use to look up IP addresses in the log data Command-line name: `network.dns_server` Command-line shortcut: `ds`
DNS timeout (seconds)	Amount of time to wait for DNS response before timing out Command-line name: `network.dns_timeout` Command-line shortcut: `dt`
IP Numbers Cache File	The file in which to cache IP numbers after they're looked up Command-line name: `network.ip_numbers_cache_file` Command-line shortcut: `incf`

Look up IP numbers using domain nameserver (DNS)	Whether to look up IP numbers using a domain nameserver (DNS), to try to compute their hostnames Command-line name: `network.look_up_ip_numbers` Command-line shortcut: `luin`
Look up IP numbers before filtering	Whether to look up IP numbers before filtering (rather than after). Command-line name: `network.look_up_ip_numbers_before_filtering` Command-line shortcut: `luinbf`
Maximum Simultaneous DNS Lookups	The maximum number of IP addresses that Sawmill will attempt to lookup at the same time Command-line name: `network.maximum_simultaneous_dns_lookups` Command-line shortcut: `msdl`
Running Sawmill URL	The URL of a running version of Sawmill, used to insert live links into HTML email Command-line name: `network.running_statistics_url` Command-line shortcut: `rsu`
Secondary DNS Server	The hostname or IP address of the DNS server to use to look up IP addresses in the log data, if the primary DNS server fails Command-line name: `network.secondary_dns_server` Command-line shortcut: `sds`
Support email address	The email address where bug and error reports should be sent Command-line name: `network.support_email_address` Command-line shortcut: `sea`
Use TCP to communicate with DNS servers	True if Sawmill should use TCP (rather than the more standard UDP) to communicate with DNS servers Command-line name: `network.use_tcp_for_dns` Command-line shortcut: `utfd`
Number thousands divider	A divider to separate thousands in displayed numbers Command-line name: `output.number_thousands_divider` Command-line shortcut: `ntd`
Number of seconds between progress pages	The number of seconds between progress pages Command-line name: `output.progress_page_interval` Command-line shortcut: `ppi`
Convert export charset	 Command-line name: `output.convert_export_charset` Command-line shortcut: `cec`
Convert export from charset	The charset to convert from, when converting a final exported CSV file Command-line name: `output.convert_export_from_charset` Command-line shortcut: `cefc`
Convert export to charset	The charset to convert to, when converting a final exported CSV file Command-line name: `output.convert_export_to_charset` Command-line shortcut: `cetc`
Allow viewers to rebuild/ update database	Allow all statistics viewers to rebuild/update the database Command-line name: `security.allow_viewers_to_rebuild` Command-line shortcut: `avtr`
Cache reports	True if reports should be cached for faster repeat display Command-line name: `statistics.miscellaneous.cache_reports` Command-line shortcut: `cr`
Log entry name	The word to use to describe a log entry Command-line name: `statistics.miscellaneous.entry_name` Command-line shortcut: `en`
First weekday	The first weekday of the week (0=Sunday, 1=Monday, ...) Command-line name: `statistics.miscellaneous.first_weekday` Command-line shortcut: `fw`
Hidden views URL	The URL to link view buttons to when the views are not visible Command-line name: `statistics.miscellaneous.hidden_views_url` Command-line shortcut: `hvu`

Marked weekday	The weekday which appears marked in calendar months displays (0=Sunday, 1=Monday, ...) Command-line name: `statistics.miscellaneous.marked_weekday` Command-line shortcut: `mw`
Maximum session duration (seconds)	The maximum duration of a session; longer sessions are discarded from the session information Command-line name: `statistics.miscellaneous.maximum_session_duration` Command-line shortcut: `msd`
Footer text	HTML code to place at the bottom of statistics pages Command-line name: `statistics.miscellaneous.page_footer` Command-line shortcut: `pf`
Footer file	An HTML file whose contents go at the bottom of statistics pages Command-line name: `statistics.miscellaneous.page_footer_file` Command-line shortcut: `pff`
Header text	HTML code to place at the top of the statistics pages Command-line name: `statistics.miscellaneous.page_header` Command-line shortcut: `ph`
Header file	An HTML file whose contents go at the top of the statistics pages Command-line name: `statistics.miscellaneous.page_header_file` Command-line shortcut: `phf`
Page frame command	A command which is executed to generate HTML to frame Sawmill's statistics Command-line name: `statistics.miscellaneous.page_frame_command` Command-line shortcut: `pfc`
Show page links	Shows table items which starts with "http://" as a link. Command-line name: `statistics.miscellaneous.show_http_link` Command-line shortcut: `shl`
Root URL of log data server	The root URL of the server being analyzed Command-line name: `statistics.miscellaneous.server_root` Command-line shortcut: `sr`
Session timeout (seconds)	The interval after which events from the same user are considered to be part of a new session Command-line name: `statistics.miscellaneous.session_timeout` Command-line shortcut: `st`
User agent for email	Specifies the target user agent when sending emails. Command-line name: `statistics.miscellaneous.user_agent_for_emails` Command-line shortcut: `uafe`
User agent for report files	Specifies the target user agent when generating report files. Command-line name: `statistics.miscellaneous.user_agent_for_files` Command-line shortcut: `uaff`
Expand paths greater than this	The number of sessions through a path that causes the path to be expanded with "expand all" or in offline (static) statistics Command-line name: `statistics.sizes.expand_paths_greater_than` Command-line shortcut: `epgt`
Maximum continuous text length	Specifies the maximum number of characters per table item per line. Command-line name: `statistics.sizes.table_cell.maximum_continuous_text_length` Command-line shortcut: `mctl`
Maximum continuous text length offset	Specifies the minimum number of characters to break the last table item line. Command-line name: `statistics.sizes.table_cell.maximum_continuous_text_length_offset` Command-line shortcut: `mctlo`
Maximum text length	Specifies the maximum number of characters per table item. Command-line name: `statistics.sizes.table_cell.maximum_text_length` Command-line shortcut: `mtl`

Maximum continuous text length	Specifies the maximum number of characters per line in a session path and path through a page report.. Command-line name: `statistics.sizes.session_path.maximum_continuous_text_length` Command-line shortcut: `spmctl`
Maximum continuous text length offset	Specifies the minimum number of characters to break the last line in a session path and path through a page report. Command-line name: `statistics.sizes.session_path.maximum_continuous_text_length_offset` Command-line shortcut: `spmctlo`
Maximum text length	Specifies the maximum number of characters of page names in the session path and path through a page report. Command-line name: `statistics.sizes.session_path.maximum_text_length` Command-line shortcut: `spmtl`

Documentation for "Profile to use"

Short description

The name of the profile to use for this command

Long description

This specifies a profile which is to be used for the current command-line command. This is typically the first option on any command line that deals with a particular profile, e.g., you might use '-p myconfig -a bd' to rebuild a database for profile myconfig. More generally, this can be used in conjunction with **Action** and other options to build or update or expire databases from the command line, or to generate HTML files. In CGI or web server mode, this is used internally to manage profiles, and should generally not be changed.

If this option is a full pathname of an existing file, that file is read as a profile file; otherwise, Sawmill treats it as the name of a profile in the profiles subfolder of the LogAnalysisInfo folder. If that doesn't exist either, Sawmill scans all profiles in that directory to see if the label of any profile matches the specified value, and uses that profile if it matches. See **Configuration Files**.

Command line usage

Command line name:	`command_line. profile`
Command line shortcut:	`p`
Command line syntax:	`-p` *value*
Default value:	

All Options

Documentation for "Action"

Short description

The command-line action to perform

Long description

This specifies the command-line action to perform with the specified profile. The HTML interface takes care of setting this option for you as necessary, but you will need to set it manually when using the command line interface. Possible modes are:

- **build_database** (or bd): This builds or rebuilds the database from the log data, erasing any data already in the database.
- **update_database** (or ud): This adds the log data to the database, while also leaving any existing data in the database.
- **process_logs** (or pl): This processes all the log data in the log source, and generates comma-separated output for each line accepted by the filters. It does not modify or create a database. This is useful for converting log data to CSV.
- **remove_database_data** (or rdd): This expires all data from the database which is in the filter set specified by **Report filters**.
- **rebuild_cross_reference_tables** (or rcrt): This rebuilds the cross-reference tables of the database from the main table (without processing any log data). It is much faster than rebuilding the database. It can be useful if you have modified the cross-reference table settings and want to update the cross-reference tables to reflect the new settings, but don't want to rebuild the database.
- **rebuild_database_indices** (or rdi): This rebuilds the indices of the main table.
- **rebuild_database_hierarchies** (or rdh): This rebuilds the hierarchy tables of the database.
- **merge_database** (or md): This merges the contents of a database (specified with **Merge database directory**) into the current database. After it completes, the current profile's database will contain all the information it contained prior to the merge, plus the information in the added database.
- **generate_all_report_files** (or garf): This generates HTML statistics pages for all reports, and the associated images, into the folder specified by **Generate HTML report files to folder**. The files and images are linked properly, so the HTML can be browsed directly from the resulting folder. This allows statistics to be browsed "off-line," without having to run Sawmill to generate each page.
- **generate_report_files** (or grf): This generates HTML statistics pages for a particular report (specified by **Report name**), and the associated images, into the folder specified by **Generate HTML report files to folder**. The files and images are linked properly, so the HTML can be browsed directly from the resulting folder. This allows one report to be browsed "off-line," without having to run Sawmill to generate each page.
- **send_report_by_email** (or srbe): This sends a statistical report using HTML email. The report is sent to **Recipient address** with return address **Return address** using **SMTP server**. The report to send is specified by **Report name**.
- **export_csv_table** (or ect): This exports a view table as CSV text. The report to export is specified by **Report name**, and is written to the standard output stream, so this is useful only in command-line mode. See also **Output filename**, **Ending row**, **Total row** and **End of line**
- **print_values** (or pv): This displays (to the command line console) the numerical field values for a particular filter set.
- **print_subitems** (or ps): This displays (to the command line console) the subitem hierarchy for the database field specified with -fn option.
- **print_items** (or pi): This displays (to the command line console) all item values for the database field specified with -fn option.
- **list_profiles** (or lp): This displays (to the command line console) a list of the internal names of all profiles. These names can be used for command-line options that call for profile names.
- **list_reports** (or lr): This displays (to the command line console) a list of the report in the specified profile (specified

with -p *profilename*). These names can be used for command-line options that call for report names (like -rn).

- **list_log_fields** (or llf): This displays (to the command line console) a list of the internal names of the log fields in the specified profile (specified with -p *profilename*). These names can be used for log filters.
- **list_database_fields** (or ldf): This displays (to the command line console) a list of the internal names of the database fields in the specified profile (specified with -p *profilename*). These names can be used for report filters.
- **print_database_statistics** (or pds): This displays statistics on the database for a profile (specified with -p *profilename*). It is useful for tuning and debugging memory and disk usage.
- **convert_70_database** (or c70d): This updates an existing MySQL database created by Sawmill 7.0 to use the new layout used by version 7.1. This is required if you want to continue to use your existing MySQL database after upgrading to Sawmill 7.1 or later. It applies only to MySQL databases; no conversion is required for internal databases.
- **recreate_profile** (or rp): This recreates profile (specified with -p *profilename*). It deletes the profile, and recreates it using the options originally used to create it. This destroys any changes to the profile which have been made since it was created; e.g., if new log filters were added manually, they will be deleted. This rewinds the profile to the state it was in just after it was created. This also incorporates any change to the log format plug-in into the profile, so this is very useful during log format plug-in authoring, to repeatedly create the profile until the plug-in is working, without having to go through the web interface each time.
- **update_to_version** (or utv): This updates the Sawmill installation to a newer version (version number is specified with **Update to version**. **This is new and highly experimental. For maximum safety, use the existing downloads page to download new versions instead of using this feature. If you do use this, back up your LogAnalysisInfo folder first!**

Command line usage

Command line name:	`command_line.action`
Command line shortcut:	`a`
Command line syntax:	`-a` *value*
Default value:	

All Options

Documentation for "Header text"

Short description

HTML code to place at the top of the statistics pages

Long description

This specifies the HTML text to appear at the top of the statistics pages. If both this and **Header file** are specified, both will appear, and this will appear first. See also **Header file**, **Footer text**, **Footer file**, and **Page frame command**.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.page_header`
Command line shortcut:	`ph`
Command line syntax:	`-ph` *value*
Default value:	

All Options

Documentation for "Database folder"

Short description

The location of the database

Long description

This is the pathname of a database on disk. If this option is blank, Sawmill will store the database in a folder with the same name as the profile, in the Databases folder in the LogAnalysisInfo folder.

Information from log files is stored in this database, and when reports are generated, they are generated using the information in the database. See **Databases**.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.options.database_directory`
Command line shortcut:	`dd`
Command line syntax:	`-dd` *value*
Default value:	

All Options

DOCUMENTATION

Documentation for "Date format"

Short description

Format of dates in the log data

Long description

This controls the expected format of date fields in the log data. Possible formats are:

- **auto**: This handles a range of date formats automatically. Four-digit integers are treated as years, and one-or-two digit integers are treated as months or days (months are assumed to preceded days). This can be used in many cases where the exact date format is not known in advance. However, it does not support day-before-month dates.
- **mm/dd/yy**; example: 04/21/00.
- **mm/dd/yyyy**; example: 04/21/2000
- **dd/mm/yyyy**; example: 21/04/2000
- **dd/mm/yy**; example: 21/04/00.
- **ddmmmyy**; example: 21Apr00.
- **dd/mmm/yy**; example: 21/Apr/00.
- **dmmmyyyy**; example: 21Apr2000, 4Dec1998.
- **dd/mmm/yyyy**; example: 21/Apr/2000.
- **mmm/dd/yyyy**; example: Apr/21/2000.
- **mmmmm/dd/yyyy**; example: April/21/2000, "December 21, 2002", "January 5 2001" (any dividers allowed).
- **yyyy/mmm/dd**; example: 2000/Apr/21.
- **m/d/yy**; same as mm/dd/yy, but leading zeros in month and day may be omitted; examples: 4/21/00, 12/4/98, 11/23/02.
- **m/d/y**; same as mm/dd/yy, but leading zeros in month, day, and year may be omitted; examples: 4/21/0, 12/4/98, 11/23/2.
- **d/m/y**; same as dd/mm/yy, but leading zeros in month, day, and year may be omitted; examples: 21/4/0, 4/12/98, 23/11/2.
- **m/d/yyyy**; same as mm/dd/yyyy, but leading zeros in month and day may be omitted; examples: 4/21/2000, 12/4/1998, 11/23/2002.
- **d/m/yyyy**; same as dd/mm/yyyy, but leading zeros in month and day may be omitted; examples: 21/4/2000, 4/12/1998, 23/11/2002.
- **d/m/yy**; same as dd/mm/yy, but leading zeros in month and day may be omitted; examples: 21/4/00, 4/12/98, 23/11/02.
- **mmdd**; example: 0421; year is assumed to be 2002.
- **mm/dd**; example: 04/21; year is assumed to be 2002.
- **mmm dd**; example: Apr 21; year is assumed to be 2002.

- **dd/mmm/yyyy:hh:mm:ss**; example: 21/Apr/1998:08:12:45; colon between date and time may be a space instead.
- **mm/dd/yyyy hh:mm:ss**; example: 04/21/1998 08:12:45.
- **mmm dd hh:mm:ss yyyy**; example: Apr 21 08:12:45 1998. Optionally, a time zone can be specified before the year; i. e., Apr 03 15:57:15 PST 2002.
- **yyyy-mm-dd**; example: 1998-04-21.
- **yyyy/mm/dd**; example: 1998/04/21.
- **yyyy/m/d**; example: 1998/4/21.
- **yyyymmdd**; example: 19980421.
- **yyyymmddhhmmss**; example: 19980421081245.
- **yymmdd-hhmmss**; example: 980421-081245.
- **m/d/yy h:mm**; example: 4/21/98 8:12.
- **seconds_since_jan1_1970**; the number of seconds since January 1, 1970, possibly with a decimal point; example: 887395356.086578.
- **TAI64N**; TAI64N format; example @400000003c675d4000fb2ebc.

The divider between items does not need to be a slash--it can be any single character; for instance if your log format uses 04-21-2000 as its date, you can choose mm/dd/yyyy and it will work.

- **auto**: Automatic detection of date. Currently this supports only formats where the year is four digits or is two-digits and comes last, and the month is three-letter English or is numerical and comes before the day.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.date_format`
Command line shortcut:	`ldf`
Command line syntax:	`-ldf` *value*
Default value:	auto

All Options

DOCUMENTATION

Documentation for "Time format"

Short description

Format of times in the log data

Long description

This controls the expected format of time fields in the log data. Possible formats are:

- **auto**: This handles most time formats automatically. Times are assumed to be in H:M:S format, where H is the hours, M is the minutes, and S is the seconds; any of them can be one or two digits. The dividers can be any non-numerical values-- they don't have to be colons. If PM appears after the time, it is assumed to be a 12-hour PM time. This can be used in many cases where the exact date format is not known in advance.
- **hh:mm:ss**; example: 18:04:23.
- **h:mm:ss**; same as hh:mm:ss except that the leading 0 on the hour may be an omitted example: 8:12:45.
- **h:m:s**; same as hh:mm:ss except that the leading 0 on the hour, minute, or second may be an omitted example: 8:12:45, 12:8:15, 1:5:9.
- **dd/mmm/yyyy:hh:mm:ss**; example: 21/Apr/1998:08:12:45.
- **mmm dd hh:mm:ss yyyy**; example: Apr 21 08:12:45 1998.
- **h:mm:ss AM/PM**; examples: 9:32:45 AM, 12:34:22 PM.
- **h:mm:ss GMT**; examples: 9:32:45 GMT, 18:34:22 GMT.
- **h:mm**; example: 18:04, 9:32.
- **hhmm**; example: 1804.
- **hhmmss**; example: 180423.
- **yyyymmddhhmmss**; example: 19980421081245.
- **yymmdd-hhmmss**; example: 980421-081245.
- **m/d/yy h:mm**; example: 4/21/98 8:12.
- **seconds_since_jan1_1970**; the number of seconds since January 1, 1970, possibly with a decimal point; example: 887395356.086578.
- **TAI64N**; TAI64N format; example @400000003c675d4000fb2ebc.

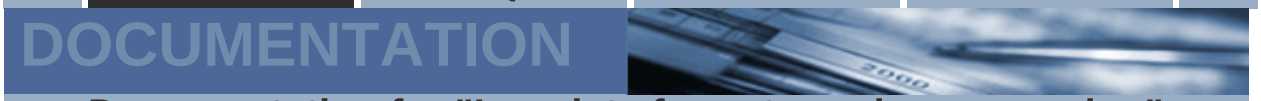
Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.time_format`

Command line shortcut:	`ltf`
Command line syntax:	`-ltf` *value*
Default value:	auto

All Options



Documentation for "Log data format regular expression"

Short description

A regular expression describing the log format

Long description

This is a regular expression which specifies the log format. You only need to change this option if you are creating your own log file format. It is a regular expression (see **Regular Expressions**) which matches an entire log entry (one line of the log), and contains a parenthesized substring for each log field. For each line that matches the regular expression, the part of the line which matches the first substring will become log field 1, the part of the line matching the second substring will be log field 2, etc. Log lines which do not match this expression will be rejected.

For example, consider the following regular expression:

```
^([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*)$
```

Each parenthesized part matches any sequence not containing a space, and the four parts must be separated by spaces. The leading ^ and trailing $ ensure that the whole line must be matched. This expression matches a line which contains four fields, separated by spaces, where the fields do not contain spaces. The first field will be put into the first log field, the second into the second log field, the third into the third log field, and the fourth into the fourth log field.

Some log formats cannot be processed using a single regular expression, either because they have peculiar field layout, or because their fields span several lines, or for some other reason. Usually it is still possible to process the log files with Sawmill, but the log format description file will need to include log parsing filters with more complicated parsing logic. For one example of that, see the Raptor file in the log_formats subfolder of the LogAnalysisInfo folder.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.parsing_regular_expression`
Command line shortcut:	`pre`
Command line syntax:	`-pre` *value*
Default value:	

All Options

Documentation for "Look up IP numbers using domain nameserver (DNS)"

Short description

Whether to look up IP numbers using a domain nameserver (DNS), to try to compute their hostnames

---

Long description

When this is true or checked, Sawmill attempts to look up the full domain name of IPs which appear in the log as IP numbers ("reverse DNS lookup"), using the DNS server specified by the **DNS Server** and **Secondary DNS Server** options. The lookup is performed as the log data is read, so if you change this option, you will need to rebuild the database to see the effects. Looking up the IP numbers provides a more human-readable format for the IP hosts, but requires a network access as frequently as once per line, so it can take much longer than leaving them as IP numbers. There are several ways to improve the performance of DNS lookup. The most important is to make sure Sawmill has a fast network connection to your DNS server; you can usually do this by running Sawmill on your web server (as a CGI program, if necessary), rather than on your desktop system. It may also be faster to configure the logging server to perform the domain name lookups, rather than having Sawmill do it. See also **Never look up IP numbers using domain name server** and **Look up IP numbers before filtering**.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`network.look_up_ip_numbers`
Command line shortcut:	`luin`
Command line syntax:	`-luin` *boolean*
Default value:	false

All Options

Documentation for "DNS timeout (seconds)"

Short description

Amount of time to wait for DNS response before timing out

Long description

This option controls the amount of time Sawmill waits for a response from a DNS (domain nameserver) when attempting to look up an IP number (see **Look up IP numbers using domain nameserver (DNS)**) during log processing. The value is in seconds; so a value of 30 means that Sawmill will give up after waiting 30 seconds for a response. Setting this to a low value may speed up your log processing, but fewer of your IP numbers will be resolved successfully.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`network.dns_timeout`
Command line shortcut:	`dt`
Command line syntax:	`-dt` *integer*
Default value:	5
Minimum value	0

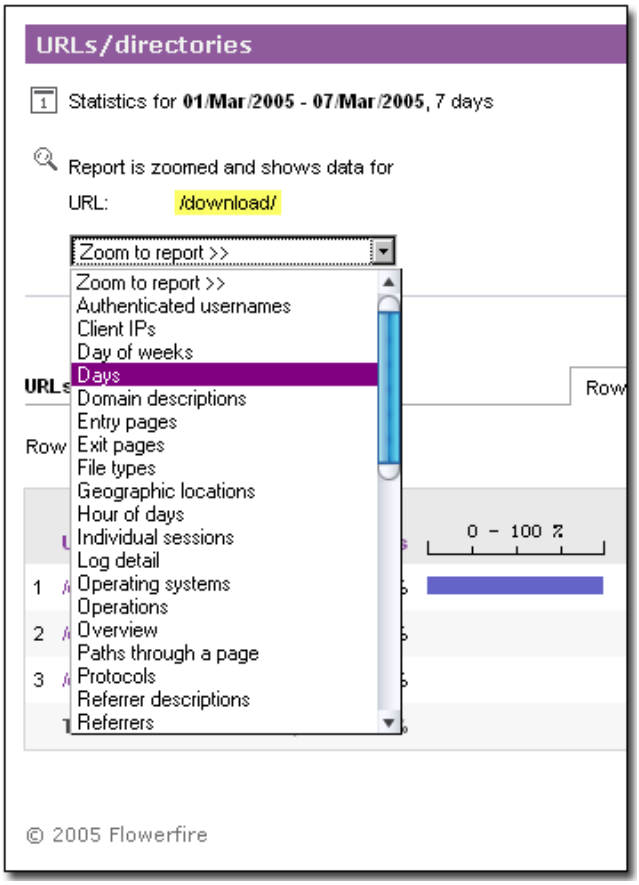
All Options

The Zoom To Menu

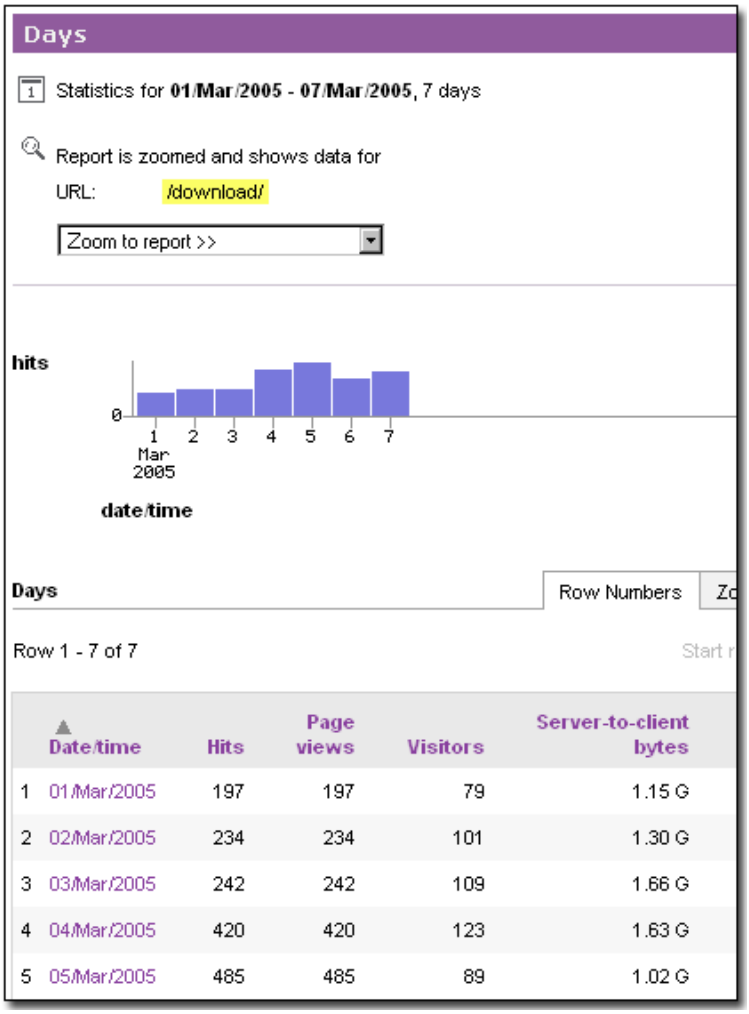
The Zoom to Menu

The Zoom to Menu shows the names of the reports that can be displayed when you Zoom.

For instance, when we select the /downloads/ directory in the pages and directories view, you can see that the 'Zoom to' Menu appears. By selecting a new Report View from the menu, you will Zoom into that new view and filter it by the selection '/download/'.



This can be useful if you want to break down each item in a table by some other report view. For instance, here we will see what traffic looked like by day, for the directory '/download/'. To do this, we choose 'URLs/Directories' view from the Report Menu, then click on '/download/' in the table and then 'Zoom to' the Days view.



	Date/time	Hits	Page views	Visitors	Server-to-client bytes
1	01/Mar/2005	197	197	79	1.15 G
2	02/Mar/2005	234	234	101	1.30 G
3	03/Mar/2005	242	242	109	1.66 G
4	04/Mar/2005	420	420	123	1.63 G
5	05/Mar/2005	485	485	89	1.02 G

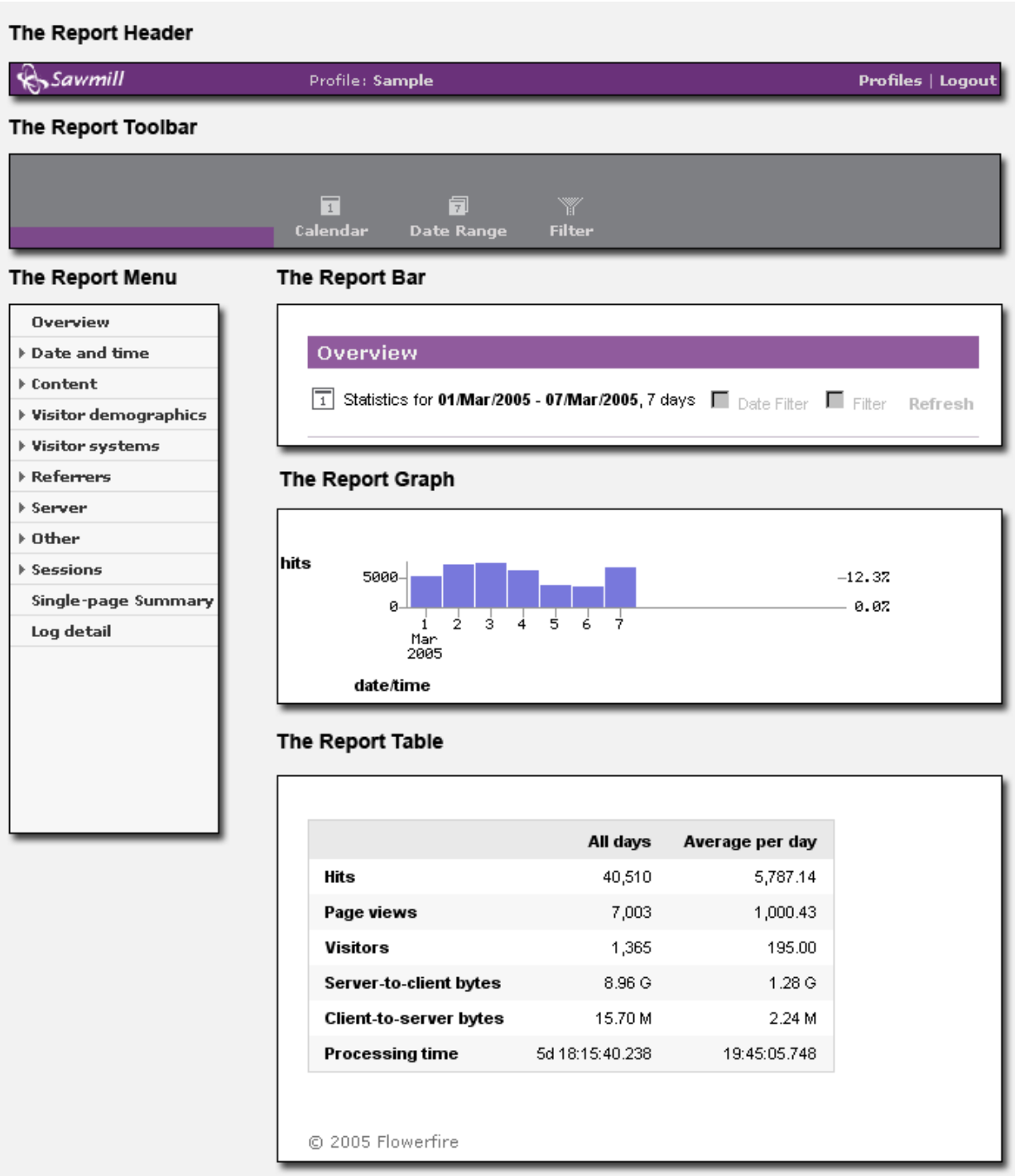
You can change the Zoom menu view repeatedly, and filter each view from the Zoom to Menu by the '/download/' directory.
You could also use **Global Filters** to add a report wide filter rather than using the Zoom Filters.

- **User Guide Index**

The Report

The Report

Each section of this image clicks through to further information about that section.

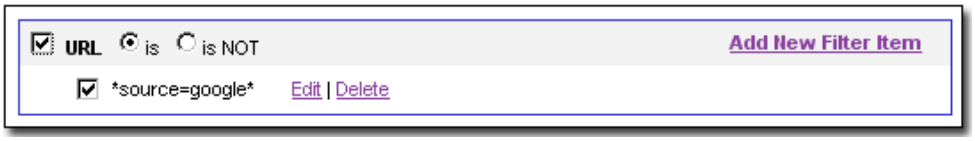


Tracking Ad Campaigns

Once you have set up the URLs in your campaign (by encoding the extra parameters on the end of the URL to your home page) you can use Sawmill to track what traffic is coming in from them.

1. Login to Sawmill.
2. Click "Show Reports".
3. Click on the Pages (or URL's) view from **The Report Menu**.
4. Select the "Filter" Option from **The Report Toolbar**.
5. Scroll down the window until you see the "Page" or "URL" field and click "**Add New Filter Item**".
6. Click the "Wildcard Expression" radio button.
7. Enter "***source=google***" into the text box (notice the ' * ' at the start and the end) and click OK.

8. Check the box turning this filter on (left of the field name, URL in this example).



9. Click "Save and Close" to close the Filter Editor.
10. Sawmill will apply this filter and refresh the report.

Documentation for "Session ID"

Short description

Internal option used to track sessions in the graphical interface

---

Long description

This is an internal option used to track sessions in the graphical interface.

Command line usage

Command line name:	`command_line. session_id`
Command line shortcut:	`si`
Command line syntax:	`-si` *value*
Default value:	

All Options

Documentation for "Generate HTML report files to folder"

Short description

Generate HTML report files into a folder

Long description

This option is used when **Action** is **generate_report_files** or **generate_all_report_files** (in command line usage). Sawmill generates statistics pages into this folder. This option determines what folder the files are generated in.

Command line usage

Command line name:	`command_line. generate_html_to_directory`
Command line shortcut:	`ghtd`
Command line syntax:	`-ghtd` *value*
Default value:	

All Options

Documentation for "Field name"

Short description

The name of a database field

Long description

This specifies the name of a database field. It is used in a variety of command-line contexts where a database field name is required, including print_values, print_subitems, and print_items (see **Action**).

Command line usage

Command line name:	`command_line.field_name`
Command line shortcut:	`fn`
Command line syntax:	`-fn` *value*
Default value:	

All Options

Documentation for "Start web server"

Short description

This option controls whether Sawmill starts its built-in web server when it starts (whether it runs in web server mode)

Long description

This controls whether Sawmill starts its built-in web server. When this option is checked (true), Sawmill starts a web server on the IP address specified by **Web server IP address**, and port specified by **Web server port**, *unless* it detects that it is running as a CGI program under another web server, in which case it responds as a CGI program instead, and does not start the server. When this option is unchecked (false), Sawmill never starts the web server, *unless* it is run from the command line with no parameters, or it is run as a GUI program under MacOS or Windows.

Command line usage

Command line name:	`command_line. web_server`
Command line shortcut:	`ws`
Command line syntax:	`-ws` *boolean*
Default value:	false

All Options

Documentation for "Generate report"

Short description

Generate a report

Long description

This generates a report, given the report ID. This is used internally to handle the generation of HTML reports.

Command line usage

Command line name:	`command_line. generate_report`
Command line shortcut:	`gr`
Command line syntax:	`-gr` *value*
Default value:	

All Options

Documentation for "Statistics viewing password"

Short description

The password required to view the statistics for this profile, passed on the command line

---

Long description

This option lets you use a direct URL to your statistics even if they are password-protected. Just include this option in the URL (clp+password) and Sawmill will treat it as if you had entered the password in the prompt field, and will bypass the prompt field and take you to the statistics.

Command line usage

Command line name:	`command_line. password`
Command line shortcut:	`clp`
Command line syntax:	`-clp` *value*
Default value:	

All Options

Documentation for "Command-line output types"

Short description

The types of command-line output to generate

Long description

This controls the types of debugging output generated during a command-line action. This option is a sequence of letters, each representing a particular type of command-line output. If the letter corresponding to a type is present in the sequence, that type of output will be generated; if it is not present, that type of output will not be generated. The types, and their corresponding letters, are:

- **e**: Error message output.
- **g**: Generate Sawmill logo (banner) output.
- **b**: Built-in web server basic output.
- **P**: Progress indicator (command line and web).
- **w**: Built-in web server debugging output.
- **f**: Filter debugging output.
- **p**: Log parsing debugging output.
- **i**: Database I/O debugging output.
- **d**: Database access debugging output.
- **D**: Detailed database access debugging output.
- **s**: Statistics generation debugging output.
- **l**: Log reading debugging output.
- **a**: Administrative debugging output.
- **m**: Language module debugging output.
- **n**: DNS debugging output.
- **N**: Detailed DNS debugging output.
- **t**: Network debugging output.
- **q**: SQL query debugging.
- **Q**: Detailed SQL query debugging.
- **o**: Add a timestamp to every output line.
- **c**: Log information about inter-process communications.
- **u**: Add the process ID (PID) to each line, in curly brackets.

For instance, a value of **`ew`** will show only error messages and basic web server output. A value of **`elbwfpidDslamnNtqQuo`** will show all possible output.

In CGI mode or web server mode, the output will be sent to a file in the Output folder of the LogAnalysisInfo folder; the file will be named Output-profilename, where profilename is the name of the profile. In command line mode (on UNIX and Windows), the output will be sent to the standard output stream.

Command line usage

Command line name:	`command_line.verbose`
Command line shortcut:	`v`
Command line syntax:	`-v` *value*
Default value:	beP

All Options

Documentation for "Master process ID"

Short description

Process ID of the master web server thread (used internally)

Long description

This is the process ID (pid) of the main web server process. This is used internally to recognize when the main process exits, so the subordinate process knows to exit too.

Command line usage

Command line name:	`command_line. master_process_id`
Command line shortcut:	`mpi`
Command line syntax:	`-mpi` *value*
Default value:	

All Options

Documentation for "Display page"

Short description

The interface page to display

Long description

This tells Sawmill which page to display in its HTML interface. This is generally used internally to deliver pages of the user interface to the browser.

Command line usage

Command line name:	`command_line. display_page`
Command line shortcut:	`dp`
Command line syntax:	`-dp` *value*
Default value:	docs.option

All Options

Documentation for "Report name"

Short description

The name of the report to generate

Long description

This specifies the name of a report to generate. This is used as a parameter in various command-line actions, including export_csv_table, email_report, and generate_report_files (see **Action**).

Command line usage

Command line name:	`command_line. report_name`
Command line shortcut:	`rn`
Command line syntax:	`-rn` *value*
Default value:	

All Options

Documentation for "Return address"

Short description

The return address of an email message

Long description

This specifies the return address of an email message. It is used when sending a report by email with send_report_by_email (see **Action**).

Command line usage

Command line name:	`command_line. return_address`
Command line shortcut:	`rna`
Command line syntax:	`-rna` *value*
Default value:	

All Options

Documentation for "Recipient address"

Short description

The recipient address of an email message

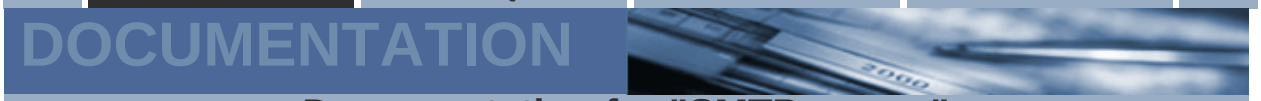
Long description

This specifies the recipient address for an email message. It is used when sending a report by email with send_report_by_email (see **Action**).

Command line usage

Command line name:	`command_line. recipient_address`
Command line shortcut:	`rca`
Command line syntax:	`-rca` *value*
Default value:	

All Options



Documentation for "SMTP server"

Short description

The SMTP server to use to send email

Long description

This specifies the SMTP server to use to send an email message. It is used when sending a report by email with send_report_by_email (see **Action**).

Command line usage

Command line name:	`command_line. smtp_server`
Command line shortcut:	`ss`
Command line syntax:	`-ss` *value*
Default value:	

All Options

Documentation for "Report email subject"

Short description

The email subject to use when sending a report by email

Long description

This specifies the email subject to use when sending a report in an email message. It is used when sending a report by email with send_report_by_email (see **Action**).

Command line usage

Command line name:	`command_line. report_email_subject`
Command line shortcut:	`res`
Command line syntax:	`-res` *value*
Default value:	Sawmill Report

All Options

Documentation for "Merge database directory"

Short description

Directory of database to merge into this one

Long description

This specifies the database directory for a database to merge into the current database. This is used together with "-a md" to add the contents of a second database to the current database. The second database must have the exact same structure as the first-- the easiest way to ensure that is to use the same profile file to build both.

Command line usage

Command line name:	`command_line. merge_database_directory`
Command line shortcut:	`mdd`
Command line syntax:	`-mdd` *value*
Default value:	

All Options

Documentation for "Report filters"

Short description

The filters to use when generating a report

Long description

This specifies the filters to use when generating a report; i.e., it filters out all data *not* matching this expression, so only part of the data is reported. This can be used in the Report Filter field of a report or a report element, when editing a report, it can be used from the command line.

The value of this option is an expression using a subset of the **The Configuration Language** syntax (see examples below). Only a subset of the language is available for this option. Specifically, the option can use:

- **within**: e.g. "(page within '/directory')" or "(date_time within '__/Jan/2004 __:__:__')"
- **<, >, <=, >=**: for date/time field only, e.g. "(date_time < '01/Jan/2004 00:00:00')"
- **and**: between any two expressions to perform the boolean "and" of those expressions
- **or**: between any two expressions to perform the boolean "or" of those expressions
- **not**: before any expression to perform the boolean "not" of that expression
- **matches**: wildcard matching, e.g. "(page matches '/index.*')"
- **matches_regexp**: regular expression matching, e.g. "(page matches_regexp '^/index\\..*$')"

Date/time values are always in the format dd/mmm/yyyy hh:mm:ss; underscores are used as wildcards, to match any value in that position. For instance, '15/Feb/2003 __:__:__' refers to a single day, and '__/Feb/2003 __:__:__' refers to a month, a '__/___/2003 __:__:__' refers to a year.

Examples

NOTE: To use these examples in a command line, or in the Extra Options of the Scheduler, use

-f "*filter*"

where *filter* is one of the examples below, e.g.,

-f "(date_time within '__/Feb/2005 __:__:__')"

Use double quotes (") around the entire filter expression; use single quotes (') within the filter if necessary.

Example: To show only events from February, 2005:

(date_time within '__/Feb/2005 __:__:__')

Example: To show only events within the page directory /picts/:

(page within '/picts/')

Example: To show only events from February, 2004, *and* within the page directory /picts/:

((date_time within '__/Jan/2004 __:__:__') and (page within '/picts/'))

Example: To show only events from February 4, 2004 through February 10, 2004:

((date_time >= '04/Jan/2004 00:00:00') and (date_time <'11/Jan/2004 00:00:00'))

Example: To show only events in the past 30 days:

(date_time >= (now() - 30*24*60*60))

Example: To show only events with source port ending with 00:

(source_port matches '*00')

Example: To show only events with source port ending with 00, *or* with destination port not ending in 00:

((source_port matches '*00') or not (destination_port matches '*00'))

Example: To show only events with server_response 404, and on pages whose names contain three consecutive digits:

((server_response inside '404') and (page matches_regexp '[0-9][0-9][0-9]'))

Command line usage

Command line name:	`command_line. filters`
Command line shortcut:	`f`
Command line syntax:	`-f` *value*
Default value:	

All Options

Documentation for "Date filter"

Short description

The date filter to use when generating a report

Long description

This specifies the date filters to use when generating a report. It is similar to the **Report filters** option, but with a few differences:

- It is intended only for date filtering (use **Report filters** for other types of filtering).
- It supports a wider range of syntax options (see below).
- It is displayed attractively in the "Date Filter" section of the report (filter specified with **Report filters** will appear below that, and will appear as literal filter expressions).

This option is most often used to create easy date or date range filters when generating reports from the command line.

The format of the value is either a date in the format yyyy, mmm/yyyy, d/mmm/yyyy (leading 0 is optional on day), or a date range D1-D2 where D1 and D2 use the same format. Examples:

```
2004
2001-2004
Jan/2004
Jan/2004-April/2005
10/Jan/2004
10/Jan/2004-15/May/2005
```

Command line usage

Command line name:	`command_line. date_filter`
Command line shortcut:	`df`
Command line syntax:	`-df` *value*
Default value:	

All Options

Documentation for "Date breakdown"

Short description

Zooms to the date which is specified in "Date filter"

Long description

This option is used in combination with the "Date filter". If the option is specified then the generated report will be zoomed to the date which is specified in "Date filter".

Command line usage

Command line name:	`command_line. date_breakdown`
Command line shortcut:	`db`
Command line syntax:	`-db` *boolean*
Default value:	false

All Options

Documentation for "Update to version"

Short description

The version to update to, using web update

Long description

This specifies the version number to update to, using the web update feature. This feature is still experimental. When used in conjunction with -a ui, this provides a command-line method of updating an existing Sawmill installation to a newer version.

Command line usage

Command line name:	`command_line. update_to_version`
Command line shortcut:	`utv`
Command line syntax:	`-utv` *value*
Default value:	

All Options

Documentation for "Expand session paths"

Short description

This option controls whether session paths should be expanded while generating a session paths report

---

Long description

This option controls whether session paths should be expanded while generating a session paths report. If this option is true, then all path segments with more than **Expand paths greater than this** will be expanded. If this option is false, only those segments which have been clicked will be expanded.

Command line usage

Command line name:	`command_line. expand_paths`
Command line shortcut:	`ep`
Command line syntax:	`-ep` *boolean*
Default value:	false

All Options

Documentation for "Generate PDF friendly files"

Short description

Generate HTML report files in a PDF friendly format

Long description

This option is used when **Action** is **generate_report_files** or **generate_all_report_files** (in command line usage). Sawmill generates statistics pages in a PDF friendly format by omitting the frameset, adding a table of contents page and by modifying specific style parameters.

Command line usage

Command line name:	`command_line. generate_pdf_friendly_files`
Command line shortcut:	`gpff`
Command line syntax:	`-gpff` *boolean*
Default value:	false

All Options

Documentation for "Zoom value"

Short description

The zoom value to zoom into a hierarchical report.

Long description

This option is used to specify the zoom value to zoom into a hierarchical report, i.e., -rn location -zv "United States/" will zoom the report location to "United States".

Command line usage

Command line name:	`command_line. zoom_value`
Command line shortcut:	`zv`
Command line syntax:	`-zv` *value*
Default value:	

All Options

Documentation for "Ending row"

Short description

This option controls the ending row of a generated or exported report.

Long description

This option controls the ending row of a generated or exported report. Ending row is a global setting and overrides existing ending row values in every report and report element.

Command line usage

Command line name:	`command_line. ending_row`
Command line shortcut:	`er`
Command line syntax:	`-er` *integer*
Default value:	

All Options

Documentation for "Total row"

Short description

Adds a total row when used with export_csv_table.

Long description

This option adds a total row to the exported data-set when it is used with **Action** **export_csv_table**.

Command line usage

Command line name:	`command_line.total_row`
Command line shortcut:	`tr`
Command line syntax:	`-tr` *boolean*
Default value:	false

All Options

Documentation for "Output filename"

Short description

Exports a report to the specified filename when used with export_csv_table.

Long description

This option exports a report to the specified filename when it is used with **Action** **export_csv_table**.

Command line usage

Command line name:	`command_line. output_filename`
Command line shortcut:	`of`
Command line syntax:	`-of` *value*
Default value:	

All Options

Documentation for "End of line"

Short description

Specifies the end of line marker in a CSV file.

Long description

This option specifies the end of line marker in a CSV file when it is used with **Action** **export_csv_table**.

Command line usage

Command line name:	`command_line.end_of_line`
Command line shortcut:	`eol`
Command line syntax:	`-eol` *value*
Default value:	

All Options

Documentation for "Never look up IP numbers using domain name server"

Short description

Whether to ever try to look up hostnames of IP-numbered hosts

---

Long description

When this is true (checked), Sawmill will never attempt to look up hostnames from IP numbers; it will use IP numbers for everything. When this is false (unchecked), it will attempt to look up the local hostname when it starts a web server, and it will attempt to look up the hostname of any host which accesses it by HTTP, and it will look up the hostname of any host it encounters in the logs (if **Look up IP numbers using domain nameserver (DNS)** is true). This option is useful if there is no local Domain Name Server (for instance, if the computer running Sawmill is not connected to a network and is not itself running a DNS).

Command line usage

Command line name:	`preferences.miscellaneous.never_look_up_ip_numbers`
Command line shortcut:	`nluin`
Command line syntax:	`-nluin` *boolean*
Default value:	false

All Options

Documentation for "Only look up IP numbers for log entries"

Short description

Look up IP numbers only when they appear in logs, not for local server or remote browsing computer

---

Long description

When this is true (checked), Sawmill will look up the hostnames of IP numbers using DNS only when they appear in a log file and **Look up IP numbers using domain nameserver (DNS)** is on. When this is false (unchecked), Sawmill will still look up numbers in log files, but will also look up the hostname of the computer Sawmill is running on, and the hostnames of computers using Sawmill through web browsers. This option is useful because when it is true, Sawmill will never do any network access, so it can be run on a computer with a dial-up connection without having to be dialed in. When this option is false, Sawmill will perform a DNS lookup when it first starts and when other computers access it, so it will have to be permanently connected to the Internet (or using a DNS server on your local network).

Command line usage

Command line name:	`preferences.miscellaneous.only_look_up_log_ip_numbers`
Command line shortcut:	`olulin`
Command line syntax:	`-olulin` *boolean*
Default value:	true

All Options

Documentation for "Logout URL"

Short description

The URL to go to on logout; if empty, goes to login screen

Long description

This specifies the URL that Sawmill sends you to when you log out of Sawmill. If this option is blank, it will send you to the Sawmill login screen.

Command line usage

Command line name:	`preferences.miscellaneous.logout_url`
Command line shortcut:	`lu`
Command line syntax:	`-lu` *value*
Default value:	

All Options

Documentation for "Temporary files lifespan"

Short description

Amount of time to keep temporary files before deleting them (in seconds)

Long description

This option controls the amount of time, in seconds, Sawmill keeps temporary files before deleting them. Temporary files include temporary profiles (used to browse statistics) and temporary images (used to embed images in statistics pages). Setting this to a high number will ensure that temporary images are around as long as they are needed, but will use more disk space.

Command line usage

Command line name:	`preferences.miscellaneous.temporary_files_lifespan`
Command line shortcut:	`tfl`
Command line syntax:	`-tfl` *integer*
Default value:	3600
Minimum value	0

All Options

Documentation for "Language"

Short description

The language module to use to generate language-specific text

---

Long description

This option specifies the language module to use. Language modules contain rules for translating from Sawmill's internal text variables to what actually appears in generated HTML pages and other output. Language modules are contained in the languages subfolder of the LogAnalysisInfo folder, in a folder named after the language (for instance, English modules are in a folder called "english"). Modules are in several pieces:

1. *lang_stats.cfg*: The text of statistics pages
2. *lang_options.cfg*: The text of the option names and descriptions.
3. *lang_admin.cfg*: The text of the administrative pages.
4. *lang_messages.cfg*: The text of error messages and other messages.

The module is split into pieces to allow for partial implementation. For instance, by implementing only the small lang_stats module, you can provide support for a particular language for statistics browsing, without having to spend a considerable amount of time to fully translate the entire Sawmill interface.

Command line usage

Command line name:	`preferences.miscellaneous.language`
Command line shortcut:	`l`
Command line syntax:	`-l` *value*
Default value:	english

All Options

Documentation for "Charset"

Short description

The HTML charset to use when displaying pages

Long description

This option specifies the HTML charset, e.g. UTF-8, to use when displaying pages in the web interface.

Command line usage

Command line name:	`preferences.miscellaneous.charset`
Command line shortcut:	`c`
Command line syntax:	`-c` *charset*
Default value:	UTF-8

All Options

Documentation for "Prompt for trial tier"

Short description

Whether to prompt for Professional/Enterprise switch during trial period

---

Long description

This option controls whether the Professional/Enterprise switch is shown during the trial period. This is useful for hiding the Professional/Enterprise switch if only the Professional or Enterprise features are available during the trial period.

Command line usage

Command line name:	`preferences.miscellaneous.prompt_for_trial_tier`
Command line shortcut:	`pftt`
Command line syntax:	`-pftt` *boolean*
Default value:	true

All Options

Documentation for "Security Mode"

Short description

The level of security to use

Long description

Sawmill provides a number of security features to prevent unauthorized access to your profiles, or to your system. The Security Mode is one of these; see also **Security**.

The security mode cannot be set from the web GUI-- it can only be set by modifying the Sawmill preferences.cfg file (in the LogAnalysisInfo folder) with a text editor. The security mode may be one of the following:

- **`browse_and_modify`**. This is the default. This mode allows web users to create new profiles, and modify existing profiles. It provides the full power of the Sawmill web interface from any web browser. It relies on the Sawmill password for security; users who have the password can create profiles, and modify existing profiles. Users who do not have the password can make temporary modifications, during browsing, to existing profiles, but they cannot modify the secure options. Secure options are those which cause files on the server to be read or written in any way; examples include **Header file**.
- **`browse_only`**. This mode adds an additional layer of security beyond what is provided by the password, by preventing users from creating or modifying profiles, even if they know the password. It allows the user to browse existing profiles, and nothing more. In this mode, profile options can be modified by directly modifying the **Configuration Files**, or by running another installation of Sawmill in Browse and Modify mode, and copying the profile.

Either of the options are secure enough to protect your system from malicious users, because all require the password before any profiles may be created or modified, and before any secure options may be changed (changes to the non-secure options cannot harm your system). If you are highly concerned about security, you may want to set the security mode to Browse Only, to prevent even password-equipped users from doing any damage.

Command line usage

Command line name:	`preferences.security.security_mode`
Command line shortcut:	`sm`
Command line syntax:	`-sm` *value*
Default value:	browse_and_modify

All Options

Documentation for "Trusted hosts"

Short description

The hostnames of computers which are "trusted," and do not need to enter passwords

---

Long description

This is a list of the hostnames of computers which are *trusted*. Hostnames should be separated from each other by spaces. Any browsing host which contains any of the listed hostnames as part of its hostname will be trusted, so entire subdomains can be trusted by entering the domain. Example:

trusted.host.com 206.221.233.20 .trusteddomain.edu

Browsers from these hosts will not be required to enter any passwords-- they will be automatically validated. Use this option with caution--it simplifies the use of Sawmill by eliminating all password screens for the administrative host, but can potentially be a security hole, if someone uses or spoofs the administrative machine without permission.

If you are connecting from a trusted host, it may be difficult to remove that trusted host using the web interface, because Sawmill will refuse to allow you administrative access to change the trusted host, because your host will no longer be trusted. One solution to this is to modify the preferences.cfg file (in the LogAnalysisInfo folder) manually, with a text editor, to remove the trusted host. Another solution is to connect from another system, log in normally, and remove the trusted host that way.

Command line usage

Command line name:	`preferences.security.trusted_hosts`
Command line shortcut:	`th`
Command line syntax:	`-th` *value*
Default value:	

All Options

Documentation for "Show full operating system details in errors"

Short description

Show full operating system version details in the text of error messages

---

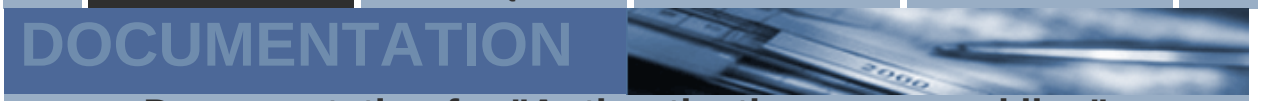
Long description

This controls whether Sawmill displays the full operating system version details in error message. It is useful for Sawmill to do this because this helps to debug problems when they are reported. However, full operating system details could be of use to someone attempting to gain unauthorized access to your server, since it would allow them to determine if you are running a vulnerable version of the operating system. This should not be an issue if you keep your operating system up to date, but if you'd rather that this information not be public, you should turn this option off.

Command line usage

Command line name:	`preferences.security.show_full_operating_system_details_in_errors`
Command line shortcut:	`sfosdie`
Command line syntax:	`-sfosdie` *boolean*
Default value:	true

All Options



Documentation for "Authentication command line"

Short description

The command line to run to authenticate users

Long description

This specifies a command line that Sawmill will run when it authenticates users. The command line program must accept two parameters: the username and the entered password. The command line must print the names of the profiles that the user is permitted to access, one name per line. A printed value of *ADMIN* means that the user is an administrator, and may access any profile, as well as accessing the administrative interface (any other response, and the administrative interface will not be available). A printed value of *FAILED* means that the username/password authentication failed.

If this option is blank, Sawmill will use the users.cfg file (in LogAnalysisInfo) to authenticate users.

Command line usage

Command line name:	`preferences.security.authentication_command_line`
Command line shortcut:	`acl`
Command line syntax:	`-acl` *value*
Default value:	

All Options

Documentation for "Default permissions"

Short description

The permissions Sawmill uses when creating a file or folder (chmod-style)

Long description

This specifies the file permissions to use when creating files and folders which do not fall into any other permissions category. This is a UNIX-style chmod value, a 3- or 4-digit octal number (see **File/Folder Permissions**).

Command line usage

Command line name:	`preferences.security.default_permissions`
Command line shortcut:	`(none)`
Command line syntax:	`-preferences.security.default_permissions` *permissions*
Default value:	644

All Options

Documentation for "Database file permissions"

Short description

The permissions Sawmill uses when creating a file as part of a database

---

Long description

This specifies the file permissions to use when creating files as part of a database. This is a UNIX-style chmod value, a 3- or 4-digit octal number (see **File/Folder Permissions**).

Command line usage

Command line name:	`preferences.security.database_file_permissions`
Command line shortcut:	`(none)`
Command line syntax:	`-preferences.security.database_file_permissions` *permissions*
Default value:	644

All Options

Documentation for "Database folder permissions"

Short description

The permissions Sawmill uses when creating a folder as part of a database

---

Long description

This specifies the file permissions to use when creating a folder as part of a database. This is a UNIX-style chmod value, a 3- or 4-digit octal number (see **File/Folder Permissions**).

Command line usage

Command line name:	`preferences.security.database_directory_permissions`
Command line shortcut:	`(none)`
Command line syntax:	`-preferences.security.database_directory_permissions` *permissions*
Default value:	1755

All Options

Documentation for "Profile file permissions"

Short description

The permissions Sawmill uses when creating a profile file

Long description

This specifies the file permissions to use when creating a profile file. This is a UNIX-style chmod value, a 3- or 4-digit octal number (see **File/Folder Permissions**).

Command line usage

Command line name:	`preferences.security.profile_permissions`
Command line shortcut:	`(none)`
Command line syntax:	`-preferences.security.profile_permissions` *permissions*
Default value:	644

All Options

Documentation for "Profile folder permissions"

Short description

The permissions Sawmill uses when creating a folder containing profile files

---

Long description

This specifies the file permissions to use when creating a folder containing profile files. This is a UNIX-style chmod value, a 3- or 4-digit octal number (see **File/Folder Permissions**).

Command line usage

Command line name:	`preferences.security.profiles_directory_permissions`
Command line shortcut:	`(none)`
Command line syntax:	`-preferences.security.profiles_directory_permissions` *permissions*
Default value:	1755

All Options

Documentation for "Temporary profile file permissions"

Short description

The permissions Sawmill uses when creating a temporary profile file

---

Long description

This specifies the file permissions to use when creating a temporary profile file. This is a UNIX-style chmod value, a 3- or 4-digit octal number (see **File/Folder Permissions**).

Command line usage

Command line name:	`preferences.security.temporary_profile_permissions`
Command line shortcut:	`(none)`
Command line syntax:	`-preferences.security.temporary_profile_permissions` *permissions*
Default value:	700

All Options

Documentation for "Default profile file permissions"

Short description

The permissions Sawmill uses when creating the default profile file

---

Long description

This specifies the file permissions to use when creating the default profile file. This is a UNIX-style chmod value, a 3- or 4-digit octal number (see **File/Folder Permissions**).

Command line usage

Command line name:	`preferences.security.default_profile_permissions`
Command line shortcut:	`(none)`
Command line syntax:	`-preferences.security.default_profile_permissions` *permissions*
Default value:	644

All Options

Documentation for "Password file permissions"

Short description

The permissions Sawmill uses when creating the password file

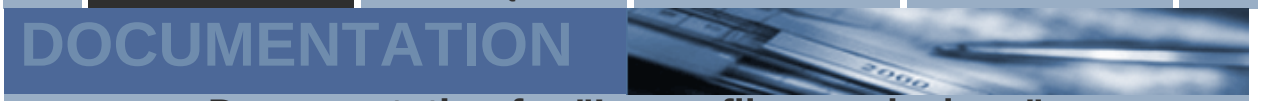
Long description

This specifies the file permissions to use when creating the password file. This is a UNIX-style chmod value, a 3- or 4-digit octal number (see **File/Folder Permissions**).

Command line usage

Command line name:	`preferences.security.password_file_permissions`
Command line shortcut:	`(none)`
Command line syntax:	`-preferences.security.password_file_permissions` *permissions*
Default value:	600

All Options



Documentation for "Image file permissions"

Short description

The permissions Sawmill uses when creating an image file

Long description

This specifies the file permissions to use when creating an image file. This is a UNIX-style chmod value, a 3- or 4-digit octal number (see **File/Folder Permissions**).

Command line usage

Command line name:	`preferences.security.image_file_permissions`
Command line shortcut:	`(none)`
Command line syntax:	`-preferences.security.image_file_permissions` *permissions*
Default value:	666

All Options

Documentation for "Image folder permissions"

Short description

The permissions Sawmill uses when creating a folder containing image files

---

Long description

This specifies the file permissions to use when creating a folder containing image files. This is a UNIX-style chmod value, a 3- or 4-digit octal number (see **File/Folder Permissions**).

Command line usage

Command line name:	`preferences.security.image_directory_permissions`
Command line shortcut:	`(none)`
Command line syntax:	`-preferences.security.image_directory_permissions` *permissions*
Default value:	755

All Options

Documentation for "Server folder permissions"

Short description

The permissions Sawmill uses when creating the server folder

Long description

This specifies the file permissions to use when creating the server folder. This is a UNIX-style chmod value, a 3- or 4-digit octal number (see **File/Folder Permissions**).

Command line usage

Command line name:	`preferences.security.server_directory_permissions`
Command line shortcut:	`(none)`
Command line syntax:	`-preferences.security.server_directory_permissions` *permissions*
Default value:	755

All Options

Documentation for "LogAnalysisInfo folder permissions"

Short description

The permissions Sawmill uses when creating the LogAnalysisInfo folder

---

Long description

This specifies the file permissions to use when creating the LogAnalysisInfo folder. This is a UNIX-style chmod value, a 3- or 4-digit octal number (see **File/Folder Permissions**).

Command line usage

Command line name:	`preferences.security.log_analysis_info_directory_permissions`
Command line shortcut:	`(none)`
Command line syntax:	`-preferences.security.log_analysis_info_directory_permissions` *permissions*
Default value:	1777

All Options

Documentation for "Show only the profiles matching REMOTE_USER"

Short description

Whether to show only the profiles whose names start with the value of REMOTE_USER

Long description

When this is true (checked), the main profile list will show only the profile, if any, whose names start with the value of REMOTE_USER followed by a dash (-). For instance, if REMOTE_USER is "tom," the Main Menu will show profiles named "tom-access," "tom-referrer," or "tom-stats," but will not show "bob-access," "tom access," or "tom." When this is false (unchecked), or if REMOTE_USER is empty (undefined), or if REMOTE_USER is equal to the value of **Administrative REMOTE_USER**, then all profiles will appear in the Main Menu. REMOTE_USER is a web server CGI variable which contains the username of the user who logged in through an authentication screen; e.g., htaccess or realms authentication. This option provides a simple mechanism for hiding users' profiles from each other, provided Sawmill is run in a section of the site protected by username/password authentication. For instance, you can run Sawmill in CGI mode, protect the Sawmill directory using authentication, turn on this option, and send the CGI URL to your users, so they will be able to log in to Sawmill with web server authentication, and they will only be able to see their own profiles. This option is only useful in CGI mode, and should not be turned on in web server mode (if it is turned on, it will make all profiles invisible), unless you are *also* running in CGI mode.

Command line usage

Command line name:	`preferences.security.show_only_remote_user_profiles`
Command line shortcut:	`sorup`
Command line syntax:	`-sorup` *boolean*
Default value:	false

All Options

Documentation for "Administrative REMOTE_USER"

Short description

The value of REMOTE_USER which marks that user as administrator

---

Long description

This option specifies the username which, when present in the REMOTE_USER environment variable in CGI mode, marks that user as the administrator. This can be used to easily integrate web server authentication with the authentication used by Sawmill, so Sawmill uses information passed in the REMOTE_USER environment variable to determine which user is logged in. See **Show only the profiles matching REMOTE_USER**.

Command line usage

Command line name:	`preferences.security.administrative_remote_user`
Command line shortcut:	`aru`
Command line syntax:	`-aru` *value*
Default value:	administrator

All Options

Documentation for "LogAnalysisInfo folder location"

Short description

A folder where Sawmill can store profiles and other information

---

Long description

This specifies a local folder where Sawmill can store profiles, databases, preferences, and other information. This folder must exist and be writable by Sawmill, or must be in a folder which is writable by Sawmill (so Sawmill can create it). If this option is empty, Sawmill assumes that the folder is named LogAnalysisInfo, and is found in the same folder as Sawmill. If a file named LogAnalysisInfoDirLoc exists in the same folder as Sawmill, the contents of that file are used as the pathname of this folder, and this option is ignored. If the environment variable LOGANALYSISINFODIR is set, its value is used instead, and this option is ignored.

Command line usage

Command line name:	`preferences.server.log_analysis_info_directory`
Command line shortcut:	`laid`
Command line syntax:	`-laid` *value*
Default value:	

All Options

Documentation for "Temporary folder"

Short description

A folder on the web server running Sawmill as a CGI program, from which images can be served

---

Long description

This specifies a folder on the web server which is running Sawmill as a CGI program. This folder will be used to serve the images which will be embedded in Sawmill's HTML pages. This folder must be accessible to Sawmill as a local folder on the machine running Sawmill, and must also be accessible through a web browser by connecting to the web server running Sawmill. In other words, it must be inside the root folder of the web server running Sawmill. The folder specified by this option must match the URL specified by **Temporary folder URL**. In other words, the value specified here, which is a pathname local to the machine running Sawmill (see **Pathnames**), must refer to the same folder which is specified by the URL in **Temporary folder URL**. It may be specified either as a full pathname, or as a pathname relative to the folder containing Sawmill (e.g. ../html/sawmill, if your server is UNIX and your site's root directory is called html and is next to cgi-bin). See **The Temporary Folder**.

Command line usage

Command line name:	`preferences.server.temporary_directory_pathname`
Command line shortcut:	`tdp`
Command line syntax:	`-tdp` *value*
Default value:	

All Options

Documentation for "Temporary folder URL"

Short description

The URL of a folder on the web server running Sawmill as a CGI program, from which images can be served

---

Long description

This specifies the URL of a folder on the web server which is running Sawmill as a CGI program. This folder will be used to serve the images which will be embedded in Sawmill's HTML pages. This folder must be accessible to Sawmill as a local folder on the machine running Sawmill, and must also be accessible through a web browser by connecting to the web server running Sawmill. Therefore, it must be inside the root folder of the web server running Sawmill. The URL specified by this option must match the folder specified by **Temporary folder**. In other words, the value specified here, which is specified by URL, must refer to the same folder which is specified by local pathname in **Temporary folder**. See **The Temporary Folder**.

Command line usage

Command line name:	`preferences.server.temporary_directory_url`
Command line shortcut:	`tdu`
Command line syntax:	`-tdu` *value*
Default value:	

All Options

Documentation for "Web server port"

Short description

The port to listen on as a web server

Long description

This specifies the port Sawmill should listen on when it runs as a web server. See **Start web server**.

Command line usage

Command line name:	`preferences.server.web_server_port`
Command line shortcut:	`wsp`
Command line syntax:	`-wsp` *integer*
Default value:	8987
Maximum value	65535
Minimum value	0

All Options

Documentation for "Maximum simultaneous tasks"

Short description

Maximum number of simultaneous web tasks (threads of execution) that Sawmill will perform

---

Long description

This specifies the maximum number of simultaneous tasks (threads of execution) that Sawmill will perform at a time, in web server mode. When a user attempts to use the built-in web server, Sawmill will check if there are already this many threads or connections actively in use. If there are, Sawmill will respond with a "too busy" page. Otherwise, the connection will be allowed. This prevents Sawmill from becoming overloaded if too many people try to use it at the same time, or if one user works it too hard (for instance, by rapidly and repeatedly clicking on a view button in the statistics).

Command line usage

Command line name:	`preferences.server.maximum_number_of_threads`
Command line shortcut:	`mnot`
Command line syntax:	`-mnot` *integer*
Default value:	8
Minimum value	1

All Options

Documentation for "CGI folder"

Short description

The folder containing Sawmill, relative to the server root

Long description

This is the folder containing the Sawmill CGI program, relative to the root of the web server. This should be as it appears in a URL; forward slashes (/) should separate subfolders. It should begin and end with a forward slash (/), unless it is empty (i.e. Sawmill is in the root folder). For instance, if the Sawmill CGI program is inside the "sawmill" folder, which is inside the "scripts" folder of your web server, then this should be /scripts/sawmill/. This is used in the rare cases when Sawmill needs to build a full (non-relative) URL for itself, ex.

http://myhost.com/cgi-bin/sawmill

Sawmill can automatically compute all parts of the URL except the CGI folder part ("/cgi-bin/" above); this option specifies that part.

Command line usage

Command line name:	`preferences.server.cgi_directory`
Command line shortcut:	`cd`
Command line syntax:	`-cd` *value*
Default value:	/cgi-bin/

All Options

Documentation for "Web server IP address"

Short description

The IP address to run Sawmill's web server on

Long description

This specifies the IP address Sawmill should run its web server on. Sawmill uses all available IPs by default, but if you want to have Sawmill's web server bind only to a specific IP, you can set this option. Sawmill uses the IP address you specify here as the IP address the server runs on.

Command line usage

Command line name:	`preferences.server.server_hostname`
Command line shortcut:	`sh`
Command line syntax:	`-sh` *value*
Default value:	

All Options

Documentation for "Cookie domain"

Short description

The domain to use in cookies sent to the browser

Long description

This specifies domain to use in cookies sent to the browser

Command line usage

Command line name:	`preferences.server.cookie_domain`
Command line shortcut:	`(none)`
Command line syntax:	`-preferences.server.cookie_domain` *value*
Default value:	

All Options

Documentation for "Automatically update database when older than"

Short description

Automatically update the database when the statistics are viewed and the database has not been updated in this many seconds

---

Long description

This controls whether Sawmill automatically updates the database when the statistics are viewed. When this option is disabled, Sawmill never updates or creates the database unless you manually tell it to. When this option is enabled, it specifies the number of seconds old a database can be before it is automatically updated when the statistics are viewed. For instance, if the value is 3600, Sawmill will automatically update the database when the statistics are viewed if it has not updated the database in the past hour (3600 seconds = 1 hour). If this value is 86400, Sawmill will only update if the database has not been updated in the past day (86400 seconds = 1 day). Regardless of the setting of this option, Sawmill will build the database from scratch when the statistics are viewed if the database has never been built before.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.options.automatically_update_when_older_than`
Command line shortcut:	`auwot`
Command line syntax:	`-auwot` *integer*
Default value:	0
Minimum value	0

All Options

Documentation for "Prompt before erasing database"

Short description

True if Sawmill should prompt before erasing a non-empty database

---

Long description

This controls whether Sawmill will prompt for verification before erasing an existing disk-based database. If this option is true, and Rebuild Database is clicked, Sawmill will ask if you really want to destroy all data in the database. If this option is false, Sawmill will erase the database without asking, and rebuild it from the log data.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.options.prompt_before_erasing_database`
Command line shortcut:	`pbed`
Command line syntax:	`-pbed` *boolean*
Default value:	true

All Options

Documentation for "Build all indices simultaneously"

Short description

Build all indices simultaneously after processing log data, for better performance

---

Long description

This option affects the stage of log processing when indices are rebuilt. This option only has an effect if **Build indices during log processing** is false. If this option is true, Sawmill will scan through the main database table just once during the index rebuilding stage, building all indices simultaneously. If this option is false, Sawmill will build each index separately, scanning through the main table once per index. Turning this option on can greatly speed up index building by combining all the table scans into one, but will use much more memory, since all indices will need to be in memory at the same time. See also **Build cross-reference tables and indices simultaneously**.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.build_all_indices_simultaneously`
Command line shortcut:	`bais`
Command line syntax:	`-bais` *boolean*
Default value:	false

All Options

Documentation for "Build all cross-reference tables simultaneously"

Short description

Build all cross-reference tables simultaneously after processing log data, for better performance

---

Long description

This option affects the stage of log processing when cross-reference tables are rebuilt. This option only has an effect if **Build cross-reference tables during log processing** is false. If this option is true, Sawmill will scan through the main database table just once during the cross-reference rebuilding stage, building all cross-reference tables simultaneously. If this option is false, Sawmill will build each cross-reference table separately, scanning through the main table once per cross-reference table. Turning this option on can greatly speed up cross-reference building by combining all the table scans into one, but will use much more memory, since all cross-reference tables will need to be in memory at the same time. See also **Build cross-reference tables and indices simultaneously**.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.build_all_xref_tables_simultaneously`
Command line shortcut:	`baxts`
Command line syntax:	`-baxts` *boolean*
Default value:	false

All Options

Documentation for "Build indices during log processing"

Short description

Build indices on the fly while log data is read, rather than in a separate stage

---

Long description

This option affects the stages of log processing when indices are built. When this option is true, indices are kept in memory during log processing, and are incrementally updated on the fly as new log lines are processed. When this option is false, indices are updated in a single stage after all log data has been processed. Turning this option on can speed database building because it eliminates the need to re-read the main database table after processing log data, but can require much more memory, because all indices must be kept in memory while log data is processed.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.build_indices_during_log_processing`
Command line shortcut:	`bidlp`
Command line syntax:	`-bidlp` *boolean*
Default value:	false

All Options

Documentation for "Build cross-reference tables and indices simultaneously"

Short description

Build cross-reference tables and indices simultaneously after processing log data, for better performance

---

Long description

This option affects the stages of log processing when cross-reference tables indices are rebuilt. This option only has an effect if **Build indices during log processing** and **Build cross-reference tables during log processing** are false. If this option is true, Sawmill will combine the index-building and cross-reference table building stages of log processing into one, scanning through the main database table once and building both indices and cross-reference tables. If this option is false, Sawmill will build indices and cross-reference tables separately, scanning through the main table twice. Turning this option on can speed up index and cross-reference table building by combining the two table scans into one, but will use more memory, since both the cross-reference tables and the indices will need to be in memory at the same time.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.build_xref_tables_and_indices_simultaneously`
Command line shortcut:	`bxtais`
Command line syntax:	`-bxtais` *boolean*
Default value:	false

All Options

Documentation for "Build cross-reference tables during log processing"

Short description

Build cross-reference tables on the fly while log data is read, rather than in a separate stage

---

Long description

This option affects the stages of log processing when cross-reference tables are built. When this option is true, cross-reference tables are kept in memory during log processing, and are incrementally updated on the fly as new log lines are processed. When this option is false, cross-reference tables are updated in a single stage after all log data has been processed. Turning this option on can speed database building because it eliminates the need to re-read the main database table after processing log data, but can require much more memory, because all cross-reference tables must be kept in memory while log data is processed.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.build_xref_tables_during_log_processing`
Command line shortcut:	`bxtdlp`
Command line syntax:	`-bxtdlp` *boolean*
Default value:	false

All Options

Documentation for "Build indices in threads"

Short description

Build indices in threads, and merge them at the end

Long description

This option affects multi-processor database builds. When this option is true, each thread (processor) builds the indices for its part of the database separately, and they are merged in a final stage to create the indices for the main database. When this option is false, threads do not build indices; the indices are built in the final stage from the main table (which is merged from the threads' main tables). If your system has fast disk I/O, it is generally best to leave this on, to spend as much time as possible using all processors. But if disk I/O is slow, the I/O contention between processes may slow both threads down to the point that using multiple processors is actually slower than using one.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.build_indices_in_threads`
Command line shortcut:	`biit`
Command line syntax:	`-biit` *boolean*
Default value:	false

All Options

Documentation for "Build indices in memory"

Short description

Build indices in memory, rather than using memory-mapped files on disk

---

Long description

When this option is true, database indices are held completely in memory during database builds. When this option is false, database indices are mapped to files on the disk. Keeping the indices in memory can increase the performance of the index building part of database builds, sometimes by a factor of 3x or more, but requires enough memory to hold the indices (which can exceed 1G in some cases).

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.build_indices_in_memory`
Command line shortcut:	`biim`
Command line syntax:	`-biim` *boolean*
Default value:	true

All Options

Documentation for "Build cross-reference tables in threads"

Short description

Build cross-reference tables in threads, and merge them at the end

---

Long description

This option affects multi-processor database builds. When this option is true, each thread (processor) builds the cross-reference tables for its part of the database separately, and they are merged in a final stage to create the cross-reference tables for the main database. When this option is false, threads do not build cross-reference tables; the cross-reference tables are built in the final stage from the main table (which is merged from the threads' main tables). If your system has fast disk I/O, it is generally best to leave this on, to spend as much time as possible using all processors. But if disk I/O is slow, the I/O contention between processes may slow both threads down to the point that using multiple processors is actually slower than using one.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.build_xref_tables_in_threads`
Command line shortcut:	`bxtit`
Command line syntax:	`-bxtit` *boolean*
Default value:	true

All Options

Documentation for "Expansion factor for database table"

Short description

Factor by which a hash table expands when necessary

Long description

This controls the factor by which the database hash table, an internal table used to store information in the database, expands when necessary. A factor of 2 means that the database table will double in size when it needs more space, while 10 means that the database table size will increase by a factor of 10. Setting this to a higher value will eliminate the need for some internal data shuffling, and will speed processing a bit; however, it will also use more memory and disk space.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.hash_table_expansion_factor`
Command line shortcut:	`htef`
Command line syntax:	`-htef` *integer*
Default value:	2

All Options

Documentation for "Initial size of database table"

Short description

Initial size of a database hash table

Long description

This controls the initial size of the database hash table, an internal table used to store information in the database. Setting this to a higher value will eliminate the need for some internal data shuffling, and will speed processing a bit; however, it will also use a bit more memory.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.hash_table_starting_size`
Command line shortcut:	`htss`
Command line syntax:	`-htss` *integer*
Default value:	4096

All Options

Documentation for "Surplus factor for database table"

Short description

Number of times larger a hash table is than its contents

Long description

This controls the amount of surplus space maintained in the database hash table, an internal table used to store information in the database. Setting this to a higher value will increase database access speed, but will use more memory. This value represents the proportion of space in the table that should remain free; when that space fills up, the table is expanded by **Expansion factor for database table**. A value of 1 means that at least 10% of the table will always be free, a value of 2 means that at least 20% is free, and so on, up to a value of 9, where at least 90% of the table is kept free at all times. With a value of 1, the same table size will hold 9 times more data than with a value of 9, so the data section of your database (which is often the largest part) will be one-ninth the size with a value of 1 than it would be with a value of 9. However, lower values slow down database building and accessing slightly.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.hash_table_surplus_factor`
Command line shortcut:	`htsf`
Command line syntax:	`-htsf` *integer*
Default value:	5

All Options

Documentation for "List cache size"

Short description

Maximum memory used by the list cache

Long description

This option specifies the maximum memory used by the list cache. The list cache is used when tracking unique item lists (e.g. visitors) or database indices, to improve performance when lists get very large. Normally, lists are stored in a form that uses minimal memory, but does not allow items to be added quickly to the list in some situations. When a list appears to be slow, it is moved to the list cache, and expanded into a high-memory-usage, high-performance format. At the end of the operation, it is compacted into the low-memory-usage format again. When the cache is full, the least-used cached lists are compacted. Setting this option higher will use more memory during database cross-reference group building and index building, but will allow more lists to be kept in the fast-access format -- this usually improves performance, sometimes dramatically.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.list_cache_size`
Command line shortcut:	`lcs`
Command line syntax:	`-lcs` *bytes*
Default value:	100M

All Options

Documentation for "Maximum main table segment size to merge"

Short description

Maximum size of main table segment to merge; larger segments will be copied

---

Long description

This option specifies the maximum size of a main table segment that will be merged while merging databases. If a segment is smaller than this, the merge will be done by adding each entry to the existing final segment of the main database table; if there are more than this number of entries, the merge will be done by copying the entire table and indices to the main database, creating a new segment. Copying is faster, but since it creates a new segment it fragments the database, slowing queries slightly. Therefore, setting this to a high value will improve the query performance of the final database, at a cost in log processing performance.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.maximum_main_table_segment_merge_size`
Command line shortcut:	`mmtsms`
Command line syntax:	`-mmtsms` *bytes*
Default value:	10M

All Options

Documentation for "Maximum main table segment size"

Short description

The maximum size of one segment of main database table

Long description

This determines the maximum size of one segment of the main database table. Segments are files stored in the database directory; Sawmill prefers to leave the entire table in a single file, but operating system limitations sometimes make that impossible. So when the table exceeds this size, it is split into multiple files, each smaller than this size. This reduces performance somewhat, but allows arbitrarily large datasets to be represented in a database. If you set this higher than the operating system allows, you will get errors when processing very large datasets (10 million lines of log data corresponds roughly to 1GB of main table, depending on the database structure and other factors).

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.maximum_main_table_segment_size`
Command line shortcut:	`mmtss`
Command line syntax:	`-mmtss` *bytes*
Default value:	100M

All Options

Documentation for "Maximum xref segment size to merge"

Short description

Maximum size of a cross-reference table segment to merge; large segments will be copied

---

Long description

This option specifies the maximum size of a cross-reference table segment which will be merged during a database merge operation; e.g., at the end of a multiprocessor database build. Segments large than this will be copied to the main database, and will form their own segments; segments smaller than this will be merged into the main database. Copies can be much faster than merges, but result in a more segmented main database, making queries slower. Therefore, setting this to a high value will improve the query performance of the final database, at a cost in log processing performance.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.maximum_xref_segment_merge_size`
Command line shortcut:	`mxsms`
Command line syntax:	`-mxsms` *bytes*
Default value:	10M

All Options

Documentation for "Maximum cross-reference table segment size"

Short description

The maximum size of one segment of a cross-reference database table

---

Long description

This determines the maximum size of one segment of a cross-reference database table. Segments are files stored in the database directory; Sawmill prefers to leave the entire table in a single file, but operating system limitations sometimes make that impossible. So when the table exceeds this size, it is split into multiple files, each smaller than this size. This reduces performance significantly, but allows arbitrarily large datasets to be represented in a database. If you set this higher than the operating system allows, you will get errors when processing very large datasets. Most operating systems can handle files up to 2 GB in size; a setting of 1 GB should be safe in most cases, and should prevent segmentation for all but the largest datasets.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`database.tuning.maximum_xref_table_segment_size`
Command line shortcut:	`mxtss`
Command line syntax:	`-mxtss` *bytes*
Default value:	100M

All Options

Documentation for "Allow newlines inside quotes"

Short description

Allow newlines (return or line feed) inside quotes, in log lines

---

Long description

This controls whether newline characters, returns or line feeds, are permitted inside quotes in log data. When this option is true, and a log line starts a quoted section but does not close it Sawmill will continue with the next line, looking for the closing quote there (or on a later line). The resulting "line" of log data will be two or more lines long, and some field values may have returns or linefeeds in them. When this option is false (unchecked), Sawmill will assume that unclosed quotes in a line are errors or formatting problems, and will treat the final (unclosed) quoted section as though there were a closing quote at the end of the line. This option should generally be left off, since turning it makes it possible that a small log data corruption can render the entire rest of the file unprocessed. But if your log data does contain entries with newlines inside quotes (as some CSV data does), then you will need to turn this option on.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.allow_newlines_inside_quotes`
Command line shortcut:	`aniq`
Command line syntax:	`-aniq` *boolean*
Default value:	false

All Options

Documentation for "Apache log format description string"

Short description

A string which describes the log format, Apache-style

Long description

This option describes the log format, in Apache log format description string style. This is intended for use as a quick way of using a custom Apache format--you can copy the format string from an Apache configuration file (or another file that uses Apache style format strings), and Sawmill will set up the log fields and format regular expressions for you. This option overrides **Log data format regular expression**.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.apache_description_string`
Command line shortcut:	`ads`
Command line syntax:	`-ads` *value*
Default value:	

All Options

Documentation for "Autodetect lines"

Short description

Number of lines to examine for this log format while auto-detecting

---

Long description

This specified the number of lines to compare using **Autodetect regular expression** while autodetecting this log format (this option is used in log format plug-ins). For instance, if this is 10, only the first 10 lines will be checked against the regular expression; if it is 100, the first 100 lines will be checked. If any line matches, the format will be considered a match.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.autodetect_lines`
Command line shortcut:	`al`
Command line syntax:	`-al` *integer*
Default value:	20

All Options

Documentation for "Autodetect regular expression"

Short description

A regular expression used to auto-detect the log format

Long description

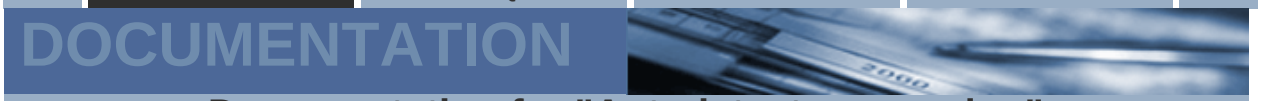
This is a regular expression which is used to auto-detect the log format. This option appears in the log format plug-in for a supported log format (plug-ins are in the log_formats folder of LogAnalysisInfo). A log file matches the format if any of the first few lines of the log file (the number of lines is specified by **Autodetect lines**) match this regular expression. See also **Log data format regular expression**, which is a similar option serving a different purpose. **Log data format regular expression** is used during log reading to separate out log fields, and does not affect auto-detection; this option is used *only* during format auto-detection, and does not affect log reading. See also **Autodetect expression**.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.autodetect_regular_expression`
Command line shortcut:	`are`
Command line syntax:	`-are` *value*
Default value:	

All Options



Documentation for "Autodetect expression"

Short description

An expression used to auto-detect the log format

Long description

This is an expression (written in the internal language) which is used to auto-detect the log format. This option appears in the log format plug-in for a supported log format (plug-ins are in the log_formats folder of LogAnalysisInfo). A log file matches the format if any of the first few lines of the log file (the number of lines is specified by **Autodetect lines**) result in a value of true for this expression (the log line is in volatile.log_data_line). See also **Log data format regular expression**, which is a similar option serving a different purpose. **Log data format regular expression** is used during log reading to separate out log fields, and does not affect auto-detection; this option is used *only* during format auto-detection, and does not affect log reading. See also **Autodetect regular expression**.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.autodetect_expression`
Command line shortcut:	`ae`
Command line syntax:	`-ae` *value*
Default value:	

All Options

Documentation for "Blue Coat log format description string"

Short description

A string which describes the log format, Blue Coat style

Long description

This option describes the log format, in Blue Coat custom log format description string style. This is intended for use as a quick way of using a custom Blue Coat format--you can copy the format string from the Blue Coat configuration interface, and Sawmill will set up the log fields and format regular expressions for you. This option overrides **Log data format regular expression**.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.blue_coat_description_string`
Command line shortcut:	`bcds`
Command line syntax:	`-bcds` *value*
Default value:	

All Options

Documentation for "Format is Common Log Format"

Short description

Log format is similar to Common Log Format

Long description

This option should be set when the log format is a Common Log Format (CLF), one of a collection of similar log formats which, among other attributes, have the date/time field in brackets, and the user field right before the bracketed date/time field. This option turns on a special work-around which is necessary for certain CLF files where the usernames contain spaces. Because CLF does not quote the username field, spaces in the username field can cause the rest of the fields to be offset, causing strange results. This option causes the field before the date/time field to be combined with any apparently separate fields, until a left-square-bracket ([) is found. This effectively allows the username field to contain spaces in CLF format. This option should be left off for any non-CLF log formats.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.common_log_format`
Command line shortcut:	`clf`
Command line syntax:	`-clf` *boolean*
Default value:	false

All Options

Documentation for "Log field separator"

Short description

The character or string that separates one log field from the next

---

Long description

This specifies the character or string which separates one log field from another in a log entry. For instance, if this is "," then log fields are comma-separated; if it is "==--==", then the fields are separate from each other by ==--==. This option only affects index/subindex style parsing of log data-- it does not affect parsing if **Parse log only with log parsing filters** is true or if **Log data format regular expression** is specified.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.field_separator`
Command line shortcut:	`fs`
Command line syntax:	`-fs` *value*
Default value:	

All Options

Documentation for "Default log date year"

Short description

The year to use, e.g., 2004, if the date format in the log data has no year information

---

Long description

This option is used if the log date format (**Date filter**) is one of the few formats which does not include year information. Sawmill will use this option's value as the year. For instance, if the date in the log is "May 7" and this option's value is 2004, then Sawmill will assume that the log entry is for May 7, 2004. The value of this option should be a four-digit integer between 1970 and 2030, or 'thisyear' --if the value of this option is 'thisyear' (without quotes), Sawmill will fill in the current year, the year the log data is *processed* in, as the year.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.default_log_date_year`
Command line shortcut:	`dldy`
Command line syntax:	`-dldy` *value*
Default value:	thisyear

All Options

Documentation for "Log data format"

Short description

The format of the log data

Long description

This specifies the name of the log format of the log data. When this appears in a log format description file, it defines the name of the format being described. When this appears in a profile, it has no effect other than providing the name of the log format plug-in used to create the profile. Sawmill sets this option a new profile is created.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.format_label`
Command line shortcut:	`fl`
Command line syntax:	`-fl` *value*
Default value:	

All Options

Documentation for "Global date regular expression"

Short description

A regular expression which, if matched in the log data, determines the date for all subsequent entries

---

Long description

This option is a regular expression (see **Regular Expressions**) which is used to extract a "global date" from log data. A global date is a date that appears in log data, usually in the header, and specifies the date for all subsequent log entries. Usually, this is used when the log entries do not contain date information at all, but if they do, this overrides them. When this option is not empty, every line of the log file is checked against this regular expression, and if it matches, the parenthesized section is remembered as the "global date". From then on, or until another global date line is found, the date field of any accepted log entry is replaced by the global date value. If this option is empty, it is not used.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.global_date_regular_expression`
Command line shortcut:	`gdre`
Command line syntax:	`-gdre` *value*
Default value:	

All Options

Documentation for "Global date filename regular expression"

Short description

A regular expression which, if matched in the log filename, determines the date for all entries in that log file

---

Long description

This option is a regular expression (see **Regular Expressions**) which is used to extract a "global date" from the name of the log file. A global date is a date that applies to all logs that appear in a file. Usually, this is used when the log entries do not contain date information at all, but if they do, this overrides them. When this option is not empty, the filename of every log processed is checked against this regular expression, and if it matches, the parenthesized section is remembered as the "global date". From then on, or until another global date filename is found, the date field of any accepted log entry is replaced by the global date value. If this option is empty, it is not used.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.global_date_filename_regular_expression`
Command line shortcut:	`gdfre`
Command line syntax:	`-gdfre` *value*
Default value:	

All Options

Documentation for "Ignore format lines"

Short description

Ignore format lines in the log data

Long description

This controls whether Sawmill ignores format lines in the log data. Format lines are lines starting with #Format, format=, or !! LOG_FORMAT, which appear in the log data, usually in a header, and describe the format of the log data on the following lines. Generally, you want to leave this option off, so Sawmill will understand log format changes if they occur in the middle of the log data. However, if you have defined custom log fields, you need to turn this on, or the field changes will be lost when format lines are encountered.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.ignore_format_lines`
Command line shortcut:	`ifl`
Command line syntax:	`-ifl` *boolean*
Default value:	false

All Options

Documentation for "Ignore quotes"

Short description

Ignore quotes in log data

Long description

This controls whether quotes (double or single) should be treated specially in log data. If this option is checked (false), quotes are treated the same as any other character. If this option is unchecked, quotes are treated specially; a quoted value containing a field divider will be treated as a single field value-- the quotes will override the field divider, and prevent it from marking the end of the field. See also **Treat square brackets as quotes**.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.ignore_quotes`
Command line shortcut:	`iq`
Command line syntax:	`-iq` *boolean*
Default value:	false

All Options

Documentation for "Parse log only with log parsing filters"

Short description

Use only the parsing filters to parse the log (and not the log format regexp, index/subindex, etc.)

Long description

This controls whether the log format regular expression (**Log data format regular expression**) option and the index/subindex settings of the log fields have any effect. This option is set in the log format plug-in to determine what type of parsing is used, and it should generally not be changed. When this is false, the (**Log data format regular expression**) option will be used to parse the log, or index/subindex options will be used if the (**Log data format regular expression**) option is empty. When this is true, only the log parsing filters (log.parsing_filters in the profile) will be used to parse the log data.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.parse_only_with_filters`
Command line shortcut:	`powf`
Command line syntax:	`-powf` *boolean*
Default value:	false

All Options

Documentation for "Treat square brackets as quotes"

Short description

Treat square brackets as quotes

Long description

This controls whether square brackets ([and]) should be treated the same as quotes (") when they are encountered in a log entry. For some log formats; e.g., Common Access Log Format, it is convenient to think of square brackets as a special kind of quote; whatever they contain is treated as a single field.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.treat_brackets_as_quotes`
Command line shortcut:	`tbaq`
Command line syntax:	`-tbaq` *boolean*
Default value:	false

All Options

Documentation for "Treat apostrophes (') as quotes"

Short description

Treat apostrophes (') as quotes

Long description

This controls whether apostrophes (') should be treated the same as quotes (") when they are encountered in a log entry. Some log formats include literal apostrophes in the data, and do not intend them to be treated as quotes; for these log formats, this option should be set to false.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.format.treat_apostrophes_as_quotes`
Command line shortcut:	`taaq`
Command line syntax:	`-taaq` *boolean*
Default value:	true

All Options

Documentation for "Allow empty log source"

Short description

True if Sawmill should allow databases to be created from log sources which contain no data

---

Long description

This option controls whether Sawmill complains if the log source is empty when the database is built or rebuilt. If this option is false, Sawmill will generate an error if there is no data in the log source during a (re)build. If this is true, Sawmill will not complain, but will just create a database containing no data. An empty log source is often a sign of an error in the log source, so it is usually best to leave this option off. But in a multi-user environment, some sites may have no log data at all, and in that case, this can be turned on to allow for error-free rebuilds of all databases. Sawmill never generates an error if there is no (new) log data during a database update; this affects only "from scratch" (re)builds.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.processing.allow_empty_log_source`
Command line shortcut:	`aels`
Command line syntax:	`-aels` *boolean*
Default value:	false

All Options

Documentation for "Date offset"

Short description

The number of hours to add to each date in the log file

Long description

This specifies the number of hours to add to the dates in the log file. A positive value causes that many hours to be added to every date in the log as it is read, and a negative value causes hours to be subtracted. For instance, if your log data is in GMT (as some formats are, including some W3C-based formats) but your time zone is GMT-8 (i.e. Pacific Standard Time), then you should enter -8 here. A value of zero leaves the dates unchanged. Fractional hours are allowed; e.g., 9.5.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.processing.date_offset`
Command line shortcut:	`do`
Command line syntax:	`-do` *value*
Default value:	0

All Options

Documentation for "Log entry pool size"

Short description

The number of log entries Sawmill can work on simultaneously

Long description

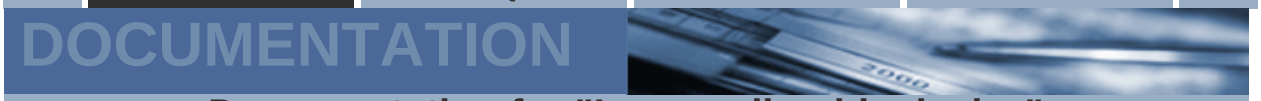
This controls the number of log entries Sawmill can work on at a time. Increasing this value may improve performance of DNS lookup. However, it will also use more memory.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.processing.log_entry_pool_size`
Command line shortcut:	`eps`
Command line syntax:	`-eps` *integer*
Default value:	1000
Minimum value	1

All Options



Documentation for "Log reading block size"

Short description

Size in bytes of the blocks which are read from the log

Long description

This controls the size in bytes of the blocks which are read from the log data. Sawmill reads the log data in chunks, processing each chunk completely before continuing to the next. Larger settings will reduce the number of disk accesses, potentially speeding processing time, but will also require the specified number of bytes of memory.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.processing.read_block_size`
Command line shortcut:	`rbs`
Command line syntax:	`-rbs` *bytes*
Default value:	100k
Minimum value	1

All Options

Documentation for "Skip processed files on update (by pathname)"

Short description

Skip files which have already been processed, judging by their pathnames, during a database update or add operation

---

Long description

This controls whether Sawmill uses the pathname of log files to determine if the files have already been added to the database. If this option is checked (true), then Sawmill will skip over any log files in the log source if it has already added a file with that name to the database. This can speed processing, especially when using FTP, because Sawmill does not have to download or process the file data and use it's more sophisticated checking mechanism to see if the data has been processed. However, it will not work properly if you have log files in your log source which are growing from update to update, or if you have log files with the same name which contain different data. If this option is off, Sawmill will handle those situations correctly, but it will have to download and examine the log data of all files to do it.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.processing.skip_processed_filenames_on_update`
Command line shortcut:	`spfod`
Command line syntax:	`-spfod` *boolean*
Default value:	false

All Options

Documentation for "Thread data block size"

Short description

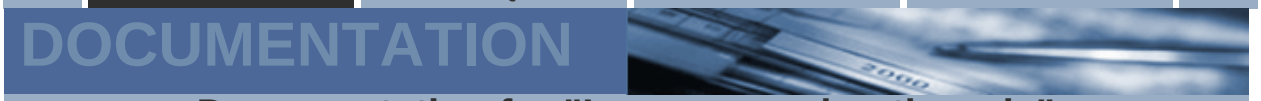
Long description

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.processing.thread_data_block_size`
Command line shortcut:	`tdbs`
Command line syntax:	`-tdbs` *bytes*
Default value:	1M
Minimum value	1

All Options



Documentation for "Log processing threads"

Short description

The number of simultaneous threads to use to process log data

---

Long description

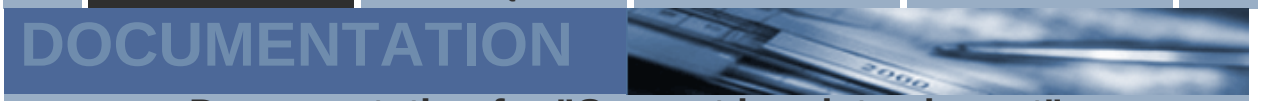
This specifies the number of threads of execution to use to process log data. The threads will execute simultaneously, each processing a portion of the log data, and at the end of processing, their results will be merged into the main database. On systems with multiple processors, using one thread per processor can result in a significant speedup of using a single thread.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.processing.threads`
Command line shortcut:	`lpt`
Command line syntax:	`-lpt` *integer*
Default value:	0
Minimum value	0

All Options



Documentation for "Convert log data charset"

Short description

Long description

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.processing.convert_log_data_charset`
Command line shortcut:	`cldc`
Command line syntax:	`-cldc` *boolean*
Default value:	false

All Options

Documentation for "Convert log data from charset"

Short description

The charset to convert from, when converting input log data

Long description

If this option is not empty, it will be used together with **Convert log data to charset** to convert the log data from the log source to a different charset from the one it is currently in. This option specifies the charset the log data is in to begin with; **Convert export to charset** specifies the charset that the log data will be in after conversion; e.g., the charset that will be seen by log filters and in the database.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.processing.convert_log_data_from_charset`
Command line shortcut:	`cldfc`
Command line syntax:	`-cldfc` *value*
Default value:	

All Options

Documentation for "Convert log data to charset"

Short description

The charset to convert to, when converting input log data

Long description

If this option is not empty, it will be used together with **Convert log data from charset** to convert the log data from the log source to a different charset from the one it is currently in. **Convert log data from charset** specifies the charset the log data is in to begin with; this option specifies the charset that the log data will be in after conversion; e.g., the charset that will be seen by log filters and in the database.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`log.processing.convert_log_data_to_charset`
Command line shortcut:	`cldtc`
Command line syntax:	`-cldtc` *value*
Default value:	

All Options

Documentation for "Actions email address(es)"

Short description

The address(es) that Sawmill should send email to whenever an action completes; e.g., the database is built

---

Long description

This specifies the address or addresses Sawmill should send email to whenever an action occurs, for instance when the database finishes rebuilding, updating, expiring, or when HTML files are done being generated. If this option is non-empty, Sawmill will send a brief description of what it just finished doing, using the SMTP server specified by **SMTP server**. Multiple recipients may be specified with commas, e.g., "user1@mydomain.com,user2@mydomain.com,user3@mydomain.com". If this option is empty, Sawmill will not send email.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`network.actions_email_address`
Command line shortcut:	`aea`
Command line syntax:	`-aea` *value*
Default value:	

All Options

Documentation for "DNS Server"

Short description

The hostname or IP address of the DNS server to use to look up IP addresses in the log data

---

Long description

This specifies the DNS server to use when looking up IP addresses in the log data (when **Look up IP numbers using domain nameserver (DNS)** is true). This can be either a hostname or an IP address of the DNS server. If this option is empty, and Sawmill is running on a UNIX-type operating system, it will use the system's default primary DNS server. On all other platforms (including Windows), this option must be set when **Look up IP numbers using domain nameserver (DNS)** is true.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`network.dns_server`
Command line shortcut:	`ds`
Command line syntax:	`-ds` *value*
Default value:	

All Options

Documentation for "IP Numbers Cache File"

Short description

The file in which to cache IP numbers after they're looked up

---

Long description

This specifies a file where Sawmill should store a database of IP numbers it has looked up in the past (see **Look up IP numbers using domain nameserver (DNS)**). When Sawmill looks up an IP number, it will look in this cache first, to see if it has already found the hostname for that IP number (or if it has already determined that the hostname cannot be found). If it finds the IP number in the cache stored in this file, it will use that hostname, rather than performing the reverse DNS lookup again. This can greatly improve the speed of converting IP numbers to hostnames, especially when the same log is analyzed again.

This option can be either a full pathname of a file, in which case that file will be used, or a single filename, in which case the file will be created inside the LogAnalysisInfo folder.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`network.ip_numbers_cache_file`
Command line shortcut:	`incf`
Command line syntax:	`-incf` *value*
Default value:	IPNumberCache

All Options

Documentation for "Look up IP numbers before filtering"

Short description

Whether to look up IP numbers before filtering (rather than after).

Long description

NOTE: As of Sawmill 6.1 (which is when we added our fast asynchronous DNS lookup algorithm), this option has no effect. In a later version, we may add it back. Here's what it *used* to do:

When this is true (checked), Sawmill performs IP number lookup (reverse DNS) before running the filters (see **Using Log Filters** and **Look up IP numbers using domain nameserver (DNS)**). When this is false (unchecked), Sawmill performs IP number lookup after running the filters. Note that this does not determine *whether* Sawmill performs DNS (**Look up IP numbers using domain nameserver (DNS)** does that); it determines *when*. Setting this to true allows filtering based on hostnames, even when the hosts are in IP number format in the log data. Setting this to false saves time, by looking up only those IP numbers which correspond to entries which are accepted by the filters.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`network.look_up_ip_numbers_before_filtering`
Command line shortcut:	`luinbf`
Command line syntax:	`-luinbf` *boolean*
Default value:	false

All Options

Documentation for "Maximum Simultaneous DNS Lookups"

Short description

The maximum number of IP addresses that Sawmill will attempt to lookup at the same time

---

Long description

This specifies the maximum number of IP addresses that will be looked up simultaneously. Setting this to a high value may increase DNS lookup performance, but if you set it too high, you may exceed operating system limitations, and the log processing may fail.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`network.maximum_simultaneous_dns_lookups`
Command line shortcut:	`msdl`
Command line syntax:	`-msdl` *integer*
Default value:	10
Maximum value	100
Minimum value	1

All Options

Documentation for "Running Sawmill URL"

Short description

The URL of a running version of Sawmill, used to insert live links into HTML email

---

Long description

This specifies the URL of a running copy of Sawmill. The URL may be something like http://www.flowerfire.com:8987/if Sawmill is running in web server mode, or it may be http://www.domainname.com/cgi-bin/sawmill if Sawmill is running in CGI mode. The URL is used to embed "live" links in HTML email; for instance, it allows your HTML email to include tables of items which, when clicked, open a web browser and display more information on that item (as they would if the table were in a normal live Sawmill report). If this option is empty, links will not appear in HTML email.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`network.running_statistics_url`
Command line shortcut:	`rsu`
Command line syntax:	`-rsu` *value*
Default value:	

All Options

Documentation for "Secondary DNS Server"

Short description

The hostname or IP address of the DNS server to use to look up IP addresses in the log data, if the primary DNS server fails

---

Long description

This specifies a secondary DNS server to use when looking up IP addresses in the log data (when **Look up IP numbers using domain nameserver (DNS)** is true). This can be either a hostname or an IP address of the DNS server. If this option is empty, and Sawmill is running on a UNIX-type operating system, it will use the system's default secondary DNS server. On all other platforms (including Windows), this option must be set when **Look up IP numbers using domain nameserver (DNS)** is true. This is used only if the primary DNS server (**DNS Server**) does not respond.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`network.secondary_dns_server`
Command line shortcut:	`sds`
Command line syntax:	`-sds` *value*
Default value:	

All Options

Documentation for "Support email address"

Short description

The email address where bug and error reports should be sent

Long description

This option specifies the email address where bug reports should be sent when a Sawmill user clicks the "Report It" button on an error message. If this is blank, it will go to the software vendor's support address (support@flowerfire.com). That's fine for some situations, especially if the reporting user is the Sawmill administrator, but for ISPs and other multi-client and multi-user installations, most of the errors will be configuration issues that the software vendor can't do anything about, and that the reporting user can't fix (because they don't have administrative access). For multi-client licensing setups, this should be set to the email address of the Sawmill administrator, who can fix the problems as they occur.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`network.support_email_address`
Command line shortcut:	`sea`
Command line syntax:	`-sea` *value*
Default value:	

All Options

Documentation for "Use TCP to communicate with DNS servers"

Short description

True if Sawmill should use TCP (rather than the more standard UDP) to communicate with DNS servers

---

Long description

This specifies whether Sawmill should use the TCP protocol when communicating with DNS servers. DNS servers more commonly communicate using UDP, and UDP is generally faster, but in some cases it may be preferable to use TCP instead. For instance, if your DNS server is accessible only by TCP due to its configuration or network location.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`network.use_tcp_for_dns`
Command line shortcut:	`utfd`
Command line syntax:	`-utfd` *boolean*
Default value:	false

All Options

Documentation for "Number thousands divider"

Short description

A divider to separate thousands in displayed numbers

Long description

This option specifies the value to separate thousands in the displayed number. For instance, if this option is empty, a number may be displayed as 123456789. If the value of this option is a comma (,), the number will be 123,456,789. If it's a period (.), the number will be 123,456,789. If it's a space, the number will be 123 456 789. This can be used to localize number divisions. If this option is empty the value of lang_stats.number.thousands_divider will be used; i.e., leave this value empty to use the current language's default divider.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`output.number_thousands_divider`
Command line shortcut:	`ntd`
Command line syntax:	`-ntd` *value*
Default value:	

All Options

Documentation for "Number of seconds between progress pages"

Short description

The number of seconds between progress pages

Long description

This controls the number of seconds which elapse between the progress pages or command-line progress indicators, which appear when the progress display is enabled (see **Command-line output types**).

The "progress" (p) option (see **Command-line output types**) controls whether a progress indicator will appear during long operations (like reading a large log file). When Sawmill is used from the command line, this option causes it to show a single-line text progress indicator. There isn't enough room on a single 80-character line to show all the information that's shown on a graphical progress page, but Sawmill shows the most important parts:

```
G:[@@@@@@@@@@ ]47% 643779e E00:20:42 R00:20:01 25M/1976k
```

The first character (G in this case) is the first letter of the full description of the current operation, as it would appear in the graphical view. For instance, in this case the G stands for "Getting data by FTP." Other common operations are "(R)eading data" (from a local file or command) and "(E)rasing database."

The section in brackets is a progress meter, which gradually fills as the task progresses, and is completely full at the end. The percentage after it is the percentage of to task that is now complete. If Sawmill cannot determine the length of the task (for instance, if it's processing gzipped log data, or bzipped log data, or log data from a command), then it will not show anything in the bar area, and it will show ??% as the percentage.

The next section (643779e above) is the number of log entries that Sawmill has processed.

The next section (E00:20:42 above) is the time elapsed since processing began, in hours:minutes:seconds format. That is followed by the estimated time remaining, (R00:20:01 above), in the same format. If Sawmill cannot determine the length of the task, the time remaining will be R??:??:??.

The last two numbers (25 MB/1976 KB above) are the memory used by the database (25 MB in this case), and the disk space used by the database (1976 KB in this case). Note that this is just the memory used by this database; Sawmill itself will be using additional memory for other purposes, so the total Sawmill memory usage will be higher than this number.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`output.progress_page_interval`
Command line shortcut:	`ppi`
Command line syntax:	`-ppi` *integer*
Default value:	1
Minimum value	1

All Options

Documentation for "Convert export charset"

Short description

Long description

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`output.convert_export_charset`
Command line shortcut:	`cec`
Command line syntax:	`-cec` *boolean*
Default value:	false

All Options

Documentation for "Convert export from charset"

Short description

The charset to convert from, when converting a final exported CSV file

---

Long description

If this option is not empty, it will be used together with **Convert export to charset** to convert the result of a CSV export to a different charset. This option specifies the charset the export is in to begin with; **Convert export to charset** specifies the charset that the export text will be in when it is displayed

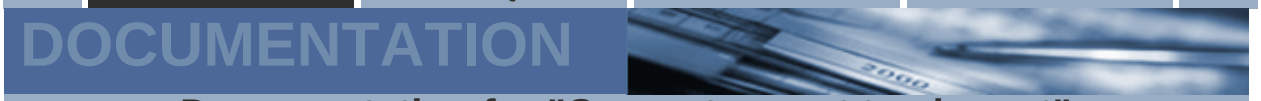
---

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`output.convert_export_from_charset`
Command line shortcut:	`cefc`
Command line syntax:	`-cefc` *value*
Default value:	

All Options



Documentation for "Convert export to charset"

Short description

The charset to convert to, when converting a final exported CSV file

---

Long description

If this option is not empty, it will be used together with **Convert export from charset** to convert the result of a CSV export to a different charset. **Convert export from charset** specifies the charset the export is in to begin with; this option specifies the charset that the export text will be in when it is displayed.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`output.convert_export_to_charset`
Command line shortcut:	`cetc`
Command line syntax:	`-cetc` *value*
Default value:	

All Options

Documentation for "Allow viewers to rebuild/update database"

Short description

Allow all statistics viewers to rebuild/update the database

Long description

When this option is checked (true), anyone viewing the statistics for the profile and rebuild or update the database, using the rebuild/update links in the reports. When this option is unchecked (false), only administrators will be able to use those links--the links will not be visible for non-administrative viewers.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`security.allow_viewers_to_rebuild`
Command line shortcut:	`avtr`
Command line syntax:	`-avtr` *boolean*
Default value:	true

All Options

Documentation for "Cache reports"

Short description

True if reports should be cached for faster repeat display

Long description

This controls whether reports are cached on disk. When this option is true, reports are saved on the disk, so if the exact same report is requested again later, it can be quickly generated without requiring database access or report generation. When this option is false, reports are regenerated every time they are viewed. Caching uses additional disk space, so it may be useful to turn this off if disk space is at a premium.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.cache_reports`
Command line shortcut:	`cr`
Command line syntax:	`-cr` *boolean*
Default value:	true

All Options

Documentation for "Log entry name"

Short description

The word to use to describe a log entry

Long description

This option specifies the word used to refer to a single log entry. For web log, for instance, this may be "hit", or for email logs it may be "message". This option is set in the log format plug-in, and generally does not need to be changed unless you are creating a new plug-in. This will appear in various places in statistics pages.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.entry_name`
Command line shortcut:	`en`
Command line syntax:	`-en` *value*
Default value:	event

All Options

Documentation for "First weekday"

Short description

The first weekday of the week (0=Sunday, 1=Monday, ...)

Long description

This controls the weekday that is considered the first day of the week. The first weekday will be the first column in calendar months and it will be the first row in weekday tables. Use 0 for Sunday, 1 for Monday, 2 for Tuesday, 3 for Wednesday, 4 for Thursday, 5 for Friday, and 6 for Saturday.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.first_weekday`
Command line shortcut:	`fw`
Command line syntax:	`-fw` *integer*
Default value:	0

All Options

Documentation for "Hidden views URL"

Short description

The URL to link view buttons to when the views are not visible

---

Long description

This controls the page that view buttons link to when the associated view is hidden. If this option is empty, the view button itself will also be hidden. Otherwise, this view button will be dimmed, and clicking the button will take you to the URL specified by this option.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.hidden_views_url`
Command line shortcut:	`hvu`
Command line syntax:	`-hvu` *value*
Default value:	

All Options

Documentation for "Marked weekday"

Short description

The weekday which appears marked in calendar months displays (0=Sunday, 1=Monday, ...)

Long description

This controls the weekday which appears in a different color in calendar months displays. The marked weekday will be displayed in a different color than the other weekdays, i.e. weekday = 0 will display the "S" for Sunday in red color. Use 0 for Sunday, 1 for Monday, 2 for Tuesday, 3 for Wednesday, 4 for Thursday, 5 for Friday, and 6 for Saturday.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.marked_weekday`
Command line shortcut:	`mw`
Command line syntax:	`-mw` *integer*
Default value:	0

All Options

Documentation for "Maximum session duration (seconds)"

Short description

The maximum duration of a session; longer sessions are discarded from the session information

---

Long description

This controls the maximum length of a session in the session information. This affects the display of session-based statistics reports like the "sessions overview", and the entry/exit page views. Sessions longer than the value specified will be ignored, and will not appear in the session information. This option is useful because some large ISPs (e.g. AOL) and other large companies use web caches that effectively make all hits from their customers to appear to be coming from one or just a few computers. When many people are using these caches at the same time, this can result in the intermixing of several true sessions in a single apparent session, resulting in incorrect session information. By discarding long sessions, which are probably the result of these caches, this problem is reduced. Also, long visits are often the result of spider visits, which are usually not useful in session reporting. The problem with caches can be eliminated entirely by configuring your web server to track "true" sessions using cookies, and then configuring Sawmill to use the cookie value (rather than the hostname field) as the visitor id. Setting this option to 0 removes any limit on session duration, so all sessions will be included.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.maximum_session_duration`
Command line shortcut:	`msd`
Command line syntax:	`-msd` *integer*
Default value:	7200
Minimum value	0

All Options

Documentation for "Footer text"

Short description

HTML code to place at the bottom of statistics pages

Long description

This specifies the HTML text to appear at the bottom of statistics pages. If both this and **Footer file** are specified, both will appear, and this will appear second. See also **Footer file**, **Header text**, **Header file**, and **Page frame command**.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.page_footer`
Command line shortcut:	`pf`
Command line syntax:	`-pf` *value*
Default value:	

All Options

Documentation for "Footer file"

Short description

An HTML file whose contents go at the bottom of statistics pages

---

Long description

This specifies a file containing HTML text to appear at the bottom of statistics pages. If both this and **Footer text** are specified, both will appear, and this will appear first. See also **Footer text**, **Header text**, **Header file**, and **Page frame command**.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.page_footer_file`
Command line shortcut:	`pff`
Command line syntax:	`-pff` *value*
Default value:	

All Options

Documentation for "Header file"

Short description

An HTML file whose contents go at the top of the statistics pages

---

Long description

This specifies a file containing HTML text to appear at the top of statistics pages. If both this and **Header text** are specified, both will appear, and this will appear second. See also **Header text**, **Footer text**, **Footer file**, and **Page frame command**.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.page_header_file`
Command line shortcut:	`phf`
Command line syntax:	`-phf` *value*
Default value:	

All Options

Documentation for "Page frame command"

Short description

A command which is executed to generate HTML to frame Sawmill's statistics

---

Long description

This option specifies a command-line program to run (UNIX and Windows only) to generate an HTML "frame" into which Sawmill's statistics page output should be inserted. This is useful for integrating Sawmill's output with the look and feel of a web site. The program should generate this HTML to its standard output stream. The frame should be a complete HTML document, starting with <HTML> and ending with </HTML>. Somewhere in the document, the text [[[[STATISTICS]]]] should appear. Sawmill will generate statistics pages by replacing that text with the statistics information, and leaving the rest of the page unchanged. See also **Footer text**, **Header text**, **Footer file**, and **Header file**.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.page_frame_command`
Command line shortcut:	`pfc`
Command line syntax:	`-pfc` *value*
Default value:	

All Options

Documentation for "Show page links"

Short description

Shows table items which starts with "http://" as a link.

Long description

This option specifies if table items which starts with "http://" should be shown as a link. If this option is enabled all table items which starts with "http://" will be shown as a link and will open the page as specified by the table item URL.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.show_http_link`
Command line shortcut:	`shl`
Command line syntax:	`-shl` *boolean*
Default value:	true

All Options

Documentation for "Root URL of log data server"

Short description

The root URL of the server being analyzed

Long description

This specifies the root URL (e.g. http://www.myserver.com/) of the web server which generated the log data. If a server root is specified, Sawmill will generate links, where possible, back to the server; these links will appear as red arrows next to page items in the tables and pie charts in reports. If the server root is not specified, these linked icons will not appear.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.server_root`
Command line shortcut:	`sr`
Command line syntax:	`-sr` *value*
Default value:	

All Options

Documentation for "Session timeout (seconds)"

Short description

The interval after which events from the same user are considered to be part of a new session

---

Long description

This controls the amount of time a session can be idle before it is considered complete. This affects the display of session-based statistics reports like the "sessions overview", and the entry/exit page views. Sessions are considered ended when a user has not contributed an event in the number of seconds specified here. For instance, if this interval is 3600 (one hour), then if a user does not contribute an event for an hour, the previous events are considered to be a single session, and any subsequent events are considered to be a new session.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.session_timeout`
Command line shortcut:	`st`
Command line syntax:	`-st` *integer*
Default value:	1800
Minimum value	0

All Options

Documentation for "User agent for email"

Short description

Specifies the target user agent when sending emails.

Long description

This option specifies the target user agent (web browser) when sending emails. Setting the user agent allows Sawmill to optimally handle line breaking for the target web browser. The user agent can be set to "msie" for Microsoft Internet Explorer, "safari" for Safari, "netscape" for Netscape and Mozilla and "unknown" if the user agent (web browser) is not known. Setting the user agent to "unknown" will break lines by spaces and by inserting a
tag; setting it to a known user agent will break lines by spaces, characters and tags as supported in the specified web browser.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.user_agent_for_emails`
Command line shortcut:	`uafe`
Command line syntax:	`-uafe` *value*
Default value:	unknown

All Options

Documentation for "User agent for report files"

Short description

Specifies the target user agent when generating report files.

Long description

This option specifies the target user agent (web browser) when generating report files. Setting the user agent allows Sawmill to optimally handle line breaking for the target web browser. The user agent can be set to "msie" for Microsoft Internet Explorer, "safari" for Safari, "netscape" for Netscape and Mozilla and "unknown" if the user agent (web browser) is not known. Setting the user agent to "unknown" will break lines by spaces and by inserting a
tag; setting it to a known user agent will break lines by spaces, characters and tags as supported in the specified web browser.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.miscellaneous.user_agent_for_files`
Command line shortcut:	`uaff`
Command line syntax:	`-uaff` *value*
Default value:	unknown

All Options

Documentation for "Expand paths greater than this"

Short description

The number of sessions through a path that causes the path to be expanded with "expand all" or in offline (static) statistics

---

Long description

This is the number of sessions through a particular path that are required for that path to be expanded in the "paths" view when "expand all" is clicked in statistics, or in offline ("Generate HTML Files") statistics. The paths view will appear with all path segments (arrows) larger than this value expanded; all paths smaller than this value will be collapsed. If you set this value too small, your paths page may be extremely large.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.sizes.expand_paths_greater_than`
Command line shortcut:	`epgt`
Command line syntax:	`-epgt` *integer*
Default value:	500
Minimum value	0

All Options

Documentation for "Maximum continuous text length"

Short description

Specifies the maximum number of characters per table item per line.

Long description

This option specifies the maximum number of characters per table item per line. Sawmill inserts a break entity if the number of characters of a table item is greater than the maximum continuous text length.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.sizes.table_cell.maximum_continuous_text_length`
Command line shortcut:	`mctl`
Command line syntax:	`-mctl` *integer*
Default value:	40

All Options

Documentation for "Maximum continuous text length offset"

Short description

Specifies the minimum number of characters to break the last table item line.

Long description

This option specifies the minimum number of characters to break the last table item line. If break entities are inserted into a long table item line then Sawmill checks the number of characters of the last line for that table item, if the number of characters are less than the specified offset then the line will not break, so the offset avoids that very few characters break into a new line. The recommended offset is 4 - 12 characters.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.sizes.table_cell.maximum_continuous_text_length_offset`
Command line shortcut:	`mctlo`
Command line syntax:	`-mctlo` *integer*
Default value:	8

All Options

Documentation for "Maximum text length"

Short description

Specifies the maximum number of characters per table item.

Long description

This option specifies the maximum number of characters per table item. Characters exceeding the maximum text length will be truncated.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.sizes.table_cell.maximum_text_length`
Command line shortcut:	`mtl`
Command line syntax:	`-mtl` *integer*
Default value:	120

All Options

Documentation for "Maximum continuous text length"

Short description

Specifies the maximum number of characters per line in a session path and path through a page report..

Long description

This option specifies the maximum number of characters per line in a session path and path through a page report. Sawmill inserts a break entity if the number of characters in a line is greater than the maximum continuous text length.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.sizes.session_path.maximum_continuous_text_length`
Command line shortcut:	`spmctl`
Command line syntax:	`-spmctl` *integer*
Default value:	70

All Options

Documentation for "Maximum continuous text length offset"

Short description

Specifies the minimum number of characters to break the last line in a session path and path through a page report.

Long description

This option specifies the minimum number of characters to break the last line in a session path and path through a page report. If break entities are inserted into a long line then Sawmill checks the number of characters of the last line, if the number of characters are less than the specified offset then the line will not break, so the offset avoids that very few characters break into a new line. The recommended offset is 4 - 12 characters.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.sizes.session_path.maximum_continuous_text_length_offset`
Command line shortcut:	`spmctlo`
Command line syntax:	`-spmctlo` *integer*
Default value:	20

All Options

Documentation for "Maximum text length"

Short description

Specifies the maximum number of characters of page names in the session path and path through a page report.

Long description

This option specifies the maximum number of characters of page names in the session path and path through a page report. Characters exceeding the maximum session path text length will be truncated.

Command line usage

Note: This is a profile option, and can only be used on the command line if a profile is specified with -p

Command line name:	`statistics.sizes.session_path.maximum_text_length`
Command line shortcut:	`spmtl`
Command line syntax:	`-spmtl` *integer*
Default value:	140

All Options

Sawmill Samples

The Sawmill Samples are live sample reports. These Samples are a very good way to get an idea of the power of Sawmill.

Sample 1: Web Log Analysis Sample (click to see sample)

Sample 2: Mail Log Analysis Sample (SpamFilter ISP) (click to see sample)

Sample 3: Media Server Log Analysis Sample (Microsoft Media Server) (click to see sample)

We need more samples! Sawmill is more than a web log analysis tool, and now supports over 500 formats, but we don't have good examples of other log data types. If you have firewall, proxy, cache, mail server, FTP server, network device, or other log data that we can use to publish other examples here, please contact us at support@flowerfire.com. We will pay store credit (i.e. you'll get a discount on Sawmill) for your log data!

If you'd rather try Sawmill on your own system, with your own log data, you can also download the trial version.

Frequently Asked Questions

Sawmill's online manual includes the answers to many frequently asked questions. Click below to view the FAQ:

The Sawmill FAQ

Sales and Technical Support

voice: +1-831-425-1758 (9AM-5PM PST)
fax: +1-831-604-1425
sales: sales@sawmill.net
support: support@sawmill.net

postal address:

Flowerfire, Inc.
125 Water Street, Suite A1
Santa Cruz, CA 95060
USA

Some users may be asked to download Mac HelpMate, or Win HelpMate, to allow for remote control.

Sales and Technical Support

voice: +1-831-425-1758 (9AM-5PM PST)
fax: +1-831-604-1425
sales: sales@sawmill.net
support: support@sawmill.net

postal address:

Flowerfire, Inc.
125 Water Street, Suite A1
Santa Cruz, CA 95060
USA

Some users may be asked to download Mac HelpMate, or Win HelpMate, to allow for remote control.

Sawmill Copyright

Info Links - Lite | Professional | Enterprise | Matrix | Profile Calc

Privacy Policy

Our Customers' Privacy

Flowerfire, Inc, the supplier of the Sawmill product line, is committed to maintaining the privacy of its customers and the online visitors. Flowerfire does not share personal information of its customers for any purpose other than as described within this Policy.

Personal Information Collection and Use

You can browse our website without the collection of any personal information. We use Sawmill to analyze web traffic, but it is in an anonymous, aggregate form. Our web log data collects IP addresses, user-agent and referrer information. Personal data is collected once you download or purchase Sawmill from us. Your personal information can include your name, email address, your mailing address and phone number. We do not share this with any third party organizations, other than our partners or resellers who have an active and direct interest in ensuring customers are kept up to date with new product announcements and releases. We use it to improve our website. Once you purchase our product, you will automatically receive emails from us. These emails are to inform you of new releases. If you wish to be removed, please email us at support@flowerfire.com.

The Use of Cookies

We set a cookie to your hard drive to identify returning users. A cookie cannot give us access to your information, beyond what you provide to us.

The Privacy of Children

Our site does not offer information or services intended to attract children. We do not intentionally gather personal information of children under 13.

Security of Your Information

Upon the purchase of Sawmill, your information is sent with SSL encryption across the Internet. We use the highest standards to protect your information against piracy or misuse of any kind.

Updating Your Personal Information

If you need to update any of your information, please email us at the support email address as indicated above.

Your Consent

By using our website, you consent to the collection and use of your personal information as outlined in this privacy policy. We may change our privacy policy and will update this site as changes are made. If you have

questions or concerns, please email, privacy@flowerfire.com or write to us: Flowerfire, Inc. 125 Water Street, Santa Cruz, CA 95060.