Insider Tips: Syslog

The System Logger

Unix systems collect messages in a central repository, making them easier to manage and troubleshoot. Syslog is a service that most admins come to depend on. MARC ANDRE SELIG

Most programs produce two types of data. Normal results are one type. For example, a game might draw images on the screen, and an FTP client would retrieve data from a server. While a program is running, it can also generate progress reports. The game might tell the user that it has finished initializing the graphics adapter. At the same time it might complain about a missing joystick. The FTP client might inform the user that a connection has been established or that a file is missing. There is a distinction between the productive data generated by a program and the same program's diagnostic messages.

Programs that run interactively output diagnostic messages directly to the screen. They often pop up dialog boxes like the ones used by modern office packages. Daemons that provide background services should not write to the screen, however. The messages they generate would only confuse the user working on the console, especially as they are unlikely to have anything to do with that user's current job. But the worst thing is that messages are simply lost, unless somebody happens to be working on the console.

Protocol files solve this issue. If a background program needs to output a diagnostic message, it has to use a file. This has always been the case for server machines, and even operating systems like Windows have rudimentary logfiles. Having every tool create a logfile of its own is inefficient. Large numbers of open files would consume system resources, and how would a program know where to store its files? Allowing programmers to decide where to save logfiles could lead to chaos.

Syslog

Most Unix systems use the effective and practical solution provided by syslog. Instead of being stored in files, messages are forwarded to a central daemon by library functions. The daemon sorts the entries and decides what to do with them, applying two criteria to make this decision: the priority and the source of the messages.

Some messages are so critical that syslog will immediately inform any users logged on to the system. For example, if my laptop's battery is low on power, I want to know about it immediately, even – or especially – if I am using an editor at the time. In contrast, statistics on the local DNS cache load status are of little importance.

Sorting messages by source is also useful. Many admins collect messages concerning incoming and outgoing email in a file. This allows you to generate traffic statistics, or check the status of missing messages. Also, some messages may be confidential. Troubleshooting an authentication module or a PPP daemon could reveal cleartext passwords. Logs of this type need more protection than a Web server's access statistics. It makes sense to store them in separate files.

There are exceptions to every rule, however, and this also holds true for central logging on Unix. Figure 1 shows an overview of the most important mechanisms and exceptions. Most programs send messages to the syslog daemon, *syslogd*, for the daemon to sort and distribute. Kernel messages are sent to the kernel logging daemon *klogd* instead. The daemon typically forwards them to syslog. Programs that do a lot of logging tend to use files of their own. Apache is a good example.

Inside Syslog

Most modern distributions store syslog files in */var/log*, although this setting is configurable. Listing 1 shows a few typical examples of syslog entries.

The message format always follows the same basic pattern. The date and timestamp come first, followed by the computer name – *undine* in our example –, and then the message itself. The message starts

with the name of the program from which it originated, typically followed by the process number in square brackets.

The first message in Listing 1 was generated by the kernel. The computer with the WLAN network card was too far away from the access point. As already shown in Figure 1, the kernel logging daemon forwarded this message to the syslog daemon. The issue that caused the message lasted for several seconds. The kernel generated a whole bunch of identical warnings during this period. Syslog recognizes this repetition and uses a simple trick to save space. Instead of repeating the message, syslog simply indicates that the message was repeated a few seconds later.

The third and fourth lines contain messages generated by programs. In this case, the cron daemon has launched a command, and the user *mas* has used the *su* utility to assume root privileges. The programs themselves are responsible for the message format. Cron uses capital letters and supplies the process ID. *su* is more subdued.

The bottom line shows a message from *syslogd* itself. The service uses markers like this one periodically if there have been no activities worth logging. This can be useful for forensic investigations as it tells a system's uptime prior to a crash. Fortunately, crashes are extremely rare on Linux, and the marker function is often disabled.

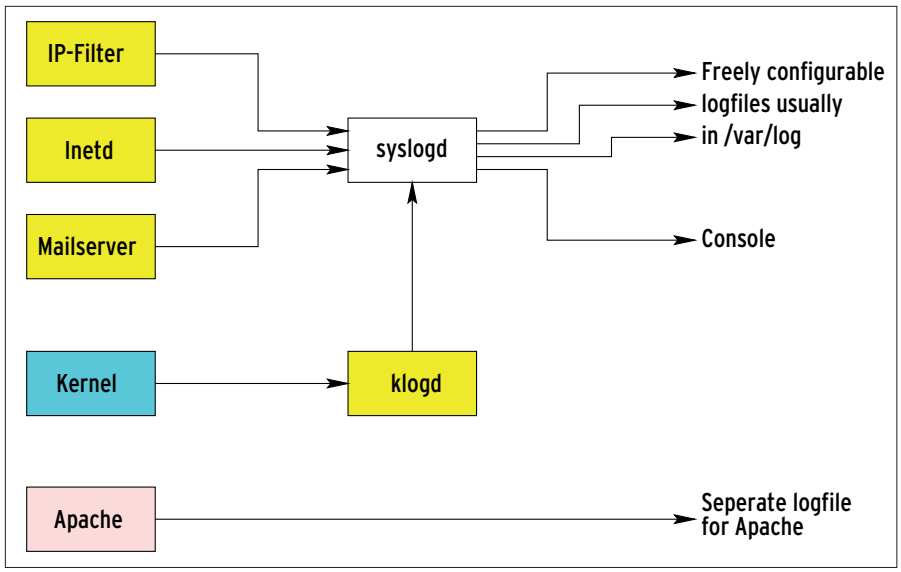

Figure 1: Most Linux programs log messages with the central syslog daemon. The kernel uses *klogd*. Syslog then stores the logs in files or forwards them to other machines

Configuration

One of syslog's most practical features is that it supports granular configuration. The admin decides where to store the logfiles. The central configuration file, */etc/syslog.conf* (see Listing 2) is used for this task.

The configuration file maps message sources (on the left) to logfile targets (on the right). The source comprises a so-called facility, that is the functional area of the system where the message originated, and the priority (separated by a period). The facilities include *mail*, *news*, *ftp*, *auth*, *kern* and others. The priorities in ascending order of importance are as follows: *debug*, *info*, *notice*, *warning*, *err*, *crit*, *alert*, and *emerg*.

The minimal *syslog.conf* in Listing 2 sends any messages that occur to the virtual console */dev/tty8*. You can press Ctrl + F8 at any time to view the messages. The activities of the mail subsystem (except debug messages) are stored in a separate file called */var/log/mail*. Any other messages go to */var/log/messages*.

The last line shows a special application where syslog forwards any critical messages to the syslog daemon on a computer called *loghost.zpid.com*. This ensures that the messages are not lost, even if the computer that generated the critical message blows up a short time later. Admins running clusters can forward syslog messages to a central system and use the syslog configuration file on the target system. This allows easy diagnostics management throughout the cluster.

The minus sign to the left of the filename in the third line is a neat tuning trick. *syslogd* will normally force the system to write each message out to disk immediately. If a lot of messages are generated, syslog will occupy the hard disk for most of the time. Files with a minus sign remove this need, and use the typical Linux cache mechanism instead. This means that messages will be stored in memory for up to thirty seconds, thus reducing the load on the system.

Admins need to tell *syslog.conf* about any changes to the system. To do so, they can either use the */etc/init.d/syslog reload* command, or if their distribution does not support that command, use the *HUP* (Hangup) signal.

```
# ps ax | grep syslog
  442 ?  S  0:00  /sbin/syslogd
# kill -HUP 442
```

Signals are one of many ways that Linux programs use to talk to each other. We will be looking into the details in another issue of this column. ■

Listing 1: Syslog messages

```
Dec  8 21:50:21 undine kernel: Tx error occurred (error 0x10)!! (maybe
distance too high?)
Dec  8 21:50:28 undine last message repeated 36 times
Dec  8 21:59:00 undine /USR/SBIN/CRON[1730]: (root) CMD ( rm -f /var/
spool/cron/lastrun/cron.hourly)
Dec  8 22:10:06 undine su: (to root) mas on /dev/pts/0
Dec  8 22:29:18 undine -- MARK --
```

Listing 2: Syslog configuration

```
*.*             /dev/tty8
mail.info       /var/log/mail
*.*;mail.none   -/var/log/messages
*.crit          @loghost.zpid.com
```