Logging with syslog-ng, Part One

By Forrest Hoffman

Used properly, system logs are like the pulse of a system. A log can often explain sources of configuration problems or foretell of impending hardware failures. Moreover, logs are invaluable for informing administrators of unscrupulous or mischievous user or cracker behavior. Of course these benefits are realized only through consistent monitoring of the logs, and that's why programs like *logwatch* (http://directory.fsf.org/logwatch.html) and *swatch* (http://swatch.sourceforge.net/) are so useful. Both provide regular email summaries and alerts based on system log activity.

Why are system logs pertient cluster and grid computing? Because cluster administrators often find themselves suddenly responsible for monitoring logs from hundreds of nodes, and grid administrators may find they need to monitor logs from a dozen, widely dispersed clusters, each containing hundreds of nodes sitting behind different firewalls enforcing different security policies. Even if clusters aren't involved, monitoring massive numbers of logs is equally challenging for enterprises with lots of servers and desktops.

Most experienced Linux administrators know about *syslogd*, and some know how to configure their systems to transmit log messages to a central server. However, few are doing so in real time for a hierarchy of servers across the Internet using encrypted transfers. If you need to monitor and manage such a configuration, try *syslog-ng* (*syslog, next generation*), a drop-in replacement for *syslogd*. *syslog-ng* provides more sophisticated log management capabilities and enables log transfers over the Internet.

Here, let's review the basic capabilities and configuration of *syslogd* and *syslog-ng* for the sake of the uninitiated. Next month's column will delve into secure transfer of the logs using encryption.

The test system used for this article was *Fedora Core 4* (FC4). Fedora has a rather complicated log handling configuration, making it ideal for demonstrating many of the features of *syslogd*. (However, trying out these features on any other Linux distribution should be relatively straightforward.)

Keeping Up with syslogd

System logs on Linux (and *Unix*) systems are typically handled by *syslogd*, a daemon whose origins lay in *BSD Unix*. *syslogd* receives and processes log messages, and records the messages in one or more log files. (In more recent Linux distributions, *syslogd* is supplemented with *klogd* for handling kernel messages separately. For example, FC4 has separate *syslogd* and *klogd* daemons both contained in a package called *sysklogd-1.4.1* which is installed by default for any configuration.)

The configuration file for *syslogd* is contained in */etc/syslog.conf*. *Listing One* shows the contents of this file for a normal FC4 installation. The first field is the *selector field* and the second field is the *action field*.

The selector field is composed of three parts: a comma-separated list of *facilities*, or sources of messages, a period (`.`), and a comma-separated list of *priorities*, or severities. You can use an asterisk (`*`) in the facility or priority portion to indicate all facilities and all priorities, respectively. A selector field may also contain multiple *facility.priority* specifications separated by semicolons (`;`) for a single action, as in `*.info; mail.none;authpriv.none;cron.none/var/log/messages`.

Table One lists all of the possible facilities (sources of messages); *Table Two* lists all possible priorities (severities of messages). While it's possible to use numeric codes for facility and priority in the */etc/syslog.conf* file, the practice is highly discouraged.

LISTING ONE: */etc/syslog.conf* for *Fedora Core 4*

```
01 # Log all kernel messages to the console.
02 # Logging much else clutters up the screen.
03 # kern.* /dev/console
04
05 # Log anything (except mail) of level info or
   higher.
06 # Don't log private authentication messages!
07 *.info;mail.none;authpriv.none;cron.none
   /var/log/messages
08
09 # The authpriv file has restricted access.
10 authpriv.* /var/log/secure
11
12 # Log all the mail messages in one place.
13 mail.* -/var/log/maillog
14
15 # Log cron stuff
16 cron.* /var/log/cron
17
18 # Everybody gets emergency messages
19 *.emerg *
20
21 # Save news errors of level crit and higher
    in a special file.
22 uucp,news.crit /var/log/spooler
23
24 # Save boot messages also to boot.log
25 local7.* /var/log/boot.log
```

TABLE ONE: *syslog facilities,* or message sources

FACILITY	FACILITY CODE
auth	4
authpriv	10
cron	9
daemon	3
kern	0
lpr	6
mail	2
mark	N/A
news	7
syslog	5
user	1
uucp	8
local0	16
…	…
local7	23

TABLE TWO: *Priorities* for messages, in increasing order of severity

PRIORITY	PRIORITY CODE
emerg	0
alert	1
crit	2
err	3
warning	4
notice	5
info	6
debug	7
none	N/A

A priority specification in the configuration file means that the rule applies to the specified priority *and any higher priority* (or lower priority code number). To limit a rule to a specific priority, use an equal sign (`=`) in front of the priority name.

For example, a selector of `mail.=info` applies only to `mail` facility messages of `info` priority, while a selector of `mail.info` applies to priorities of `info` *through* `emerg`. It's also possible to reverse the meaning of selector field components using an exclamation point (`!`).

Actions can be file names (full paths beginning with a slash), named pipes (beginning with a vertical bar, `|`), remote machines (beginning with the at sign, `@`), consoles (such as `/dev/console`), user names (comma separated), or all users (an asterisk). A minus sign (`-`) in front of a filename action means that the file should not be synced after each append. (This can improve performance on the system for files to which appends frequently occur, but some information may be lost if the system crashes before the file is synced.)

Given these rules, look again at *Listing One*. Line 7 states that all messages of priority `info` through `emerg` except for those originating from the `mail`, `authpriv`, and `cron` facilities should be logged to the file */var/log/messages*. (If you're familiar with Fedora, you'll likely recognize that almost all *syslog* entries are, in fact, written to */var/log/messages*.)

The remainder of the configuration file in *Listing One* says that messages from the `authpriv` facility should be appended to */var/log/secure*; messages from the `mail` facility should be appended to */var/log/maillog* (without syncing each time); messages from the `cron` facility should be appended to */var/log/cron*; messages from any facility with priority `emerg` should be sent to all users (`*`); messages from `uucp` and `news` of priority `crit`, `alert`, or `emerg` should be appended to */var/log/spooler;* and all boot messages (those from facility `local7`) should be appended to */usr/log/boot.log*.

A Centralized Log Collection Server

Remember that one of the possible *syslogd* actions was a remote hostname (prepended with an `@`). This feature allows a client machine to send its log messages to a centralized log server. This capability to collect log messages from network devices — including client desktops, routers, switches, and nodes in a cluster — makes *syslog* invaluable. In a Linux cluster, the master or head node can collect all of the *syslog* entries from all compute nodes.

When the action is a remote hostname, the relevant messages are sent to the remote log host over port 514 using the User Datagram Protocol (UDP). For this to work, *syslogd* on the remote log host or master node must be started with the `-r` flag to listen for this traffic. In addition, the server must not block this port/protocol in its firewall configuration (For this discussion, assume that the *syslog* client nodes are on the same network as the log-collecting server.)

In Fedora, runtime flags for both *syslogd* and *klogd* are contained in */etc/sysconfig/syslog*. By default, *syslogd* runs with only `-m 0`, which disables "mark" messages which would otherwise intersperse messages in the log file. To enable network reception of log entries from client nodes, add the `-r` flag to the options for *syslogd* only on the master node as follows:

```
SYSLOGD_OPTIONS="-m 0 -r"
```

It may also be necessary to add the `-x` flag to prevent *syslogd* from doing DNS lookups for messages coming over the network. This avoids the possibility of a deadlock if the DNS server is running on this same master node or server.

Now, restart *syslogd* with the command:

```
[root@master ~]# service syslog restart
```

Once this is done, the last lines in */var/log/messages* should look like this:

```
Sep  5 19:10:38 master kernel:
Kernel logging (proc) stopped.
Sep  5 19:10:38 master kernel:
Kernel log daemon terminating.
Sep  5 19:10:39 master exiting on signal 15
Sep  5 19:10:39 master syslogd 1.4.1:
restart (remote reception).
Sep  5 19:10:39 master kernel: klogd
1.4.1, log source = /proc/kmsg started.
```

As you can see, *syslogd* was restarted with remote reception enabled.

In a cluster, log entries should be sent only over the private, internal network link so they cannot be read from outside the cluster. The firewall configuration on the master node may be configured to accept all traffic over its private network interface, but if not, then *iptables* may block incoming *syslogd* messages from remote clients. To alter this configuration, run *system-config-securitylevel* to open a hole for port 514 UDP traffic, as shown in *Figure One*.

After adding this port/protocol with *system-config-securitylevel*, a new line is added to */etc/sysconfig/iptables*, which looks like this:

```
-A RH-Firewall-1-INPUT -m state --state
NEW -m udp -p udp --dport 514 -j ACCEPT
```

This firewall rule is probably too liberal since only traffic within the cluster (either on an internal network interface or from a specific block of IP addresses) should be accepted. For example, this rule could be changed to…

```
-A RH-Firewall-1-INPUT -m state --state
NEW -m udp -p udp -i eth0 --source
10.0.2.0/8 --dport 514 -j ACCEPT
```

… so that only messages originating from IP addresses *10.0.2.x* on the *eth0* network interface are accepted. If this change is made manually by simply editing the */etc/syslog/iptables* file, then the *iptables* service should be restarted with:

```
[root@master sysconfig]# service iptables restart
```

Configuring the Clients

Now, on the client or compute nodes, replace the */etc/syslog.conf* with the simple one shown in *Listing Two*. It forwards all messages, including kernel messages, to the master, keeping none of them on the local disk. After installing this new file, *syslogd* on the client nodes should be restarted with:

```
[root@node2 ~]# service syslog restart
```

FIGURE ONE: Screen dumps from *system-config-securitylevel,* used to create a hole for incoming *syslogd* messages on the master node/server

Messages from client nodes should immediately start appearing in the log files on the master. For example, at the end of */var/log/messages* on master, you should see…

```
Sep  5 19:02:27 master kernel:
ip_tables: (C) 2000-2002 Netfilter core team
Sep  5 19:02:27 master kernel:
ip_conntrack version 2.1 (3839 buckets,
30712 max) - 272 bytes per conntrack
Sep  5 19:10:38 master kernel:
Kernel logging (proc) stopped.
Sep  5 19:10:38 master kernel:
Kernel log daemon terminating.
Sep  5 19:10:39 master exiting on signal 15
Sep  5 19:10:39 master syslogd 1.4.1:
restart (remote reception).
Sep  5 19:10:39 master kernel: klogd
1.4.1, log source = /proc/kmsg started.
Sep  5 19:15:40 node2 syslogd 1.4.1: restart.
Sep  5 19:15:40 node2 kernel: klogd 1.4.1,
log source = /proc/kmsg started.
```

… which shows that the *iptables* and *syslog* services were restarted on master and that the *syslog* service was restarted

LISTING TWO: Client/compute node */etc/syslog.conf* file for forwarding messages to the master node

```
# Log *everything* to master instead of keeping
it local
*.*                                @master
```

FIGURE TWO: Steps for building and installing *libol* and *syslog-ng*

```
[root@master src]# cd /usr/local/src
[root@master src]# tar xzf libol-0.3.16.tar.gz
[root@master src]# cd libol-0.3.16
[root@master libol-0.3.16]# ./configure
[root@master libol-0.3.16]# make
[root@master libol-0.3.16]# make install
[root@master libol-0.3.16]# cd ..
[root@master src]# tar xzf syslog-ng-1.6.8.tar.gz
[root@master src]# cd syslog-ng-1.6.8
[root@master syslog-ng-1.6.8]# ./configure
[root@master syslog-ng-1.6.8]# make
[root@master syslog-ng-1.6.8]# make install
```

on *node2*. The name of the host originating the message is shown after the date and time for each entry.

Two other related programs, *logwatch* and *logrotate*, are usually run daily by *crond*. While *logrotate* should continue to be used on diskful client nodes to rotate off old files, the *logwatch* package could likely be removed, especially on cluster nodes. However, *logwatch* monitors other system status information and provides daily reports to the *root* user via email. Now that all system logs end up on the master node, the *logwatch* email reports and statistics generated there will include log entries from the client nodes.

Within clusters with private networks, this configuration works very well. However, if client nodes are widely distributed across the Internet (as may be the case for enterprise servers and desktops) or if one would like to collect system logs from a group of clusters that form a larger part of a grid computing network, sending messages in the clear over unreliable UDP doesn't suffice. What's needed is support for sending messages via the Transmission Control Protocol (TCP) in a secure manner through firewalls over the Internet. That's where *syslog-ng* comes in.

A syslog for the Next Generation

The *syslog-ng* package is a drop-in replacement for *syslogd* (and *klogd*) and provides very sophisticated message man-

LISTING THREE: The default *syslog-ng.conf* file for *Fedora*

```
# syslog-ng configuration file.
#
# This should behave pretty much like the original syslog on
RedHat. But
# it could be configured a lot smarter.
#
# See syslog-ng(8) and syslog-ng.conf(5) for more information.

options {
     sync (0);
     time_reopen (10);
     log_fifo_size (1000);
     long_hostnames (off);
     use_dns (no);
     use_fqdn (no);
     create_dirs (no);
     keep_hostname (yes);
};

source s_sys {
     file ("/proc/kmsg" log_prefix("kernel: "));
     unix-stream ("/dev/log");
     internal();
     # udp(ip(0.0.0.0) port(514));
};

destination d_cons { file("/dev/console"); };
destination d_mesg { file("/var/log/messages"); };
destination d_auth { file("/var/log/secure"); };
destination d_mail { file("/var/log/maillog" sync(10)); };
destination d_spol { file("/var/log/spooler"); };
destination d_boot { file("/var/log/boot.log"); };
destination d_cron { file("/var/log/cron"); };
destination d_mlal { usertty("*"); };

#filter f_filter1    { facility(kern); };
filter f_filter2    { level(info..emerg) and
                        not facility(mail,authpriv,cron); };
filter f_filter3    { facility(authpriv); };
filter f_filter4    { facility(mail); };
filter f_filter5    { level(emerg); };
filter f_filter6    { facility(uucp) or
                         (facility(news) and level(crit..emerg)); };
filter f_filter7    { facility(local7); };
filter f_filter8    { facility(cron); };

#log { source(s_sys); filter(f_filter1); destination(d_cons); };
log { source(s_sys); filter(f_filter2); destination(d_mesg); };
log { source(s_sys); filter(f_filter3); destination(d_auth); };
log { source(s_sys); filter(f_filter4); destination(d_mail); };
log { source(s_sys); filter(f_filter5); destination(d_mlal); };
log { source(s_sys); filter(f_filter6); destination(d_spol); };
log { source(s_sys); filter(f_filter7); destination(d_boot); };
log { source(s_sys); filter(f_filter8); destination(d_cron); };
```

agement, filtering, and wide area networking capabilities not available in *syslog*. Written by Balazs Scheidler, *syslog-ng* is distributed under the *GNU Public License* and is a part of BalaBit's Zorp IT (http://www.balabit.com/products/zorp/) security package. It's also included in most *Debian Linux* distributions.

syslog-ng can be downloaded from Balabit's website at http://www.balabit.com/downloads/syslog-ng/. The stable release is version *1.6.8*, and this version also requires *libol*, also available at the same site. A future *2.0* design is prototyped in the *1.9.x* series, and it appears that *1.9.5* is available as of this writing.

In a cluster environment, it is necessary to install *syslog-ng* only on the master node. Client/compute nodes can continue running plain old *syslogd*, passing their messages via UDP in the clear over the private cluster network to the master node. However, on the master node, it's probably desirable to keep local copies of all these log entries and send a copy to a centralized log host at the grid level. This configuration is easily achievable with *syslog-ng*.

The *gzip*-ped tarballs for *libol* and *syslog-ng* should be downloaded and placed in a convenient location, like */usr/local/src*. For this example, *libol-0.3.16.tar.gz* and *syslog-ng-1.6.8.tar.gz* were used. *Figure Two* shows the steps used to build and install the *syslog-ng* software.

At this point, a configuration file is needed for *syslog-ng*. Fortunately, many configuration examples are provided in *syslog-ng-1.6.8/contrib*, including support files for Fedora that match the default configuration used for *syslogd*. While it may ultimately be desirable to have a different configuration, using this layout is adequate for this example installation.

The default build of *syslog-ng* performed above places files under */usr/local* and expects the configuration file to be located at */usr/local/etc/syslog-ng/syslog-ng.conf*. However, some of the Fedora support files refer to different paths, so they should be carefully modified to reflect paths used in your own installation. Support files used in the configurations described here are available on the *Linux Magazine* web site at http://www.linux-mag.com/downloads/2005-11/extreme/ syslog-ng_fedora_defaults.zip.

Figure Three shows the steps needed to install the *syslog-ng* configuration and support files for Fedora. In addition to the main configura-

FIGURE THREE: Installing *syslog-ng* configuration and support files for Fedora

```
[root@master syslog-ng-1.6.8]# cd contrib/fedora-packaging/
[root@master fedora-packaging]# mkdir -p /usr/local/etc/syslog-ng
[root@master fedora-packaging]# cp syslog-ng.conf /usr/local/etc/syslog-ng/
[root@master fedora-packaging]# cp syslog-ng.logrotate \
  /etc/logrotate.d/syslog-ng
[root@master fedora-packaging]# cp -p syslog-ng.sysconfig \
  /etc/sysconfig/syslog-ng
[root@master fedora-packaging]# cp syslog-ng.init /etc/init.d/syslog-ng
[root@master fedora-packaging]# chmod 755 /etc/init.d/syslog-ng
```

tion file, which is copied to */usr/local/etc/syslog-ng/syslog-ng.conf*, files are provided for *logrotate* and for initialization/startup. After copying the files to the correct locations, edit */etc/init.d/syslog-ng* to be sure the file paths match your installation.

A *syslog-ng.conf* file consists of a series of option, source, destination, filter, and log sections. The options section is used to configure global options. One or more source sections are used to control where messages may originate. One or more destination sections declare where message should be sent. Filter sections declare filters that should be applied to messages. Log sections include one or more sources, one or more filters, and one or more destinations.

The default contributed configuration file for Fedora is shown in *Listing Three*. It contains an options section, a single source section, and various filter, destination, and log sections. As written, all sources are passed through the various filters and sent to corresponding destinations to match the style of the original *syslogd* configuration for Fedora. This certainly is not the only way to implement and manage log messages, but it is sufficient for demonstrating some of the capabilities of *syslog-ng* and for experimenting with cluster/grid configurations.

In the configuration in *Listing Three*, messages are concatenated from three sources: */proc/kmsg* (where kernel log messages originate), */dev/log* (where normal log messages originate), and from *syslog-ng* itself (referred to as `internal();`). The fourth source entry should be uncommented (by removing the leading `#`) to receive messages via UDP from other *syslog* daemons. For the master node configuration, this `udp()` line should be uncommented so that client/ compute node log entries are received by *syslog-ng*.

Each log entry in this configuration file sends all log entries from `source (s_sys)` through a filter and to a destination that corresponds to message types that would make it through the referenced filter. For example, filter `f_filter2` passes messages with priority levels of `info` through `emerg` where the facility is not one of `mail`, `authpriv`, or `cron`. Messages that pass this filter are sent to `destination (d_mesg)`, which appends to the */var/log/messages* file just as was done in *syslogd*.

Filters in *syslog-ng* are more useful than this configuration file implies because they not only limit messages by facility and level but also by content. So, for instance, a filter such as…

```
filter f_master1_deny { host("master1") and match("deny"); };
```

… could also be used to extract messages that should be sent directly to administrators, alerting them to potential cracking activity on the *master1* node. There are many other source, filter, and destination functions as well as configuration options; the *syslog-ng* documentation covers them pretty well.

See Extreme, *pg. 63*

Extreme, *from pg. 55*

To begin using *syslog-ng* on the master node, stop and disable the regular *syslog* service. Then enable and start the *syslog-ng* service. After all of the supporting files are installed as described above, this is accomplished as follows:

```
[root@master]# service syslog stop
Shutting down kernel logger: [  OK  ]
Shutting down system logger: [  OK  ]
[root@master] # chkconfig syslog off
[root@master] # chkconfig —add syslog-ng
[root@master] # chkconfig syslog-ng on
[root@master] # service syslog-ng start
Starting syslog-ng: [  OK  ]
```

Now *syslog-ng* should be running on the master node and receiving log entries from client/compute nodes running regular old *syslogd*. The next step is to have *syslog-ng* forward a copy of these logs to a central grid host also running *syslog-ng* over the Internet. This could be accomplished by adding a new destination to the configuration file as follows:

```
destination gridhost { tcp("123.45.67.89")
port(514)); };
```

This sends the logs to the machine at `123.45.67.89` via TCP to port 514, but there are a number of problems with this. First, TCP port 514 was originally used for *rsh*, but you should never run *rsh*, so the port is probably unused. Second, that port and many others are probably firewalled by institutional policy, so firewall exceptions at the remote site would be needed to make this work. Third, the log messages would traverse the Internet in the clear, so anyone can read them and learn all kinds of things about your machines and system configurations.

These problems can be overcome by forwarding messages over some encrypted connection between the two machines on two different site networks. Encrypted channels can be established either by using *ssh* to create a reverse tunnel from the central grid host or by using *stunnel* on both sides to create a tunnel between the two systems using SSL (secure sockets layer) certificates. Next month's column will explore both of these methods and their unique advantages and disadvantages.

In the interim, check out some of the configuration and filter options available in *syslog-ng*. They can make managing log information easy and help filter important messages that can alert you when hardware is failing or when nodes are under attack.

Forrest Hoffman is a computer modeling and simulation researcher at Oak Ridge National Laboratory. He can be reached at forrest @climate.ornl.gov.