



Log management



Log file locations

- Most logs will reside in `/var/log` directory
- Each application or daemon will typically have a log file titled as its name. for example `/var/log/yum.log`, which contains events and actions from the yum command
- Examples of important log files include:
 - `/var/log/boot.log` which contains rc startup scripts events
 - `/var/log/maillog` which includes mail-related messages
 - `/var/log/cron` where cron message are logged
- Any log messages that do not have its own log file are placed in the generic `/var/log/messages`
- The `/var/log/messages` also includes startup error messages and failed login attempts.



Understanding syslog

- Before syslog, each developer had to implement a specific logging policy for the application. System administrators could not easily determine log files location and had no control over which events to get logged and which to be ignored, depending on the severity level
- Syslog was designed to be the central logging point of the system. It aggregates all logs in a well-known location, can control logging according to the severity level and can also redirect logs to other locations.
- The syslog system is made up of the following components:
 - The daemon: syslogd. It can be configured from `/etc/sysconfig.conf`
 - The C functions: `openlog`, `closelog`, and `syslog` that are used to communicate with the daemon
 - The shell command: `logger`. It is used to communicate with the daemon from the shell.
- On Linux systems, the daemon that handles logging is called `rsyslog`

How do daemons talk to syslog?

- ▶ Syslogd daemon is configured by default to start at boot time
- ▶ Applications that are willing to communicate with the `syslogd` daemon to write their logs call the `syslog` routine (a C function) and write to the domain socket `/dev/log`. (A domain socket is a special file used for process intercommunication).
- ▶ The `syslogd` daemon reads data from this socket and acts according to its configuration file (writes to a specific log file or sends the message to a predefined destination).
- ▶ Any changes to the `/etc/syslog.conf` are not reflected until the daemon process receives a HUP signal. You can easily find the process id by reading it from `/var/run/syslog.pid` (instead running `ps -ef`). For example

```
kill -HUP `cat /var/run/syslog.pid`
```

Syslogd configuration

- It is located in `/etc/syslog.conf`
- Lines starting with `#` are ignored
- A typical line is formatted as follows:
`selector action`
- Selectors and actions must be separated by tabs (spaces may not work)
- A selector is made up of a facility (program) and a severity level separated by a dot. For example:
`mail.info /var/log/maillog`
- The facility is restricted to kernel programs (like mail and cron). Third party applications cannot have their own facilities, instead, they can be referred to as user.
- Multiple facilities can be grouped, separated by commas
- Both the facility and the severity can be replaced with `*` to “match all”
- A severity of “none” means that the facility is ignored (useful when deselecting a facility from a catch all condition like `*.info, facility.none`)



Severity levels

- ▶ They range from the most verbose (least important and up) to the least verbose (most important)
- ▶ They are eight levels:
 - ▶ emerg: top urgent (like panic)
 - ▶ alert: urgent but less than emerg
 - ▶ crit: critical situations
 - ▶ err: when errors happen
 - ▶ warning: just a warning, nothing serious happened (yet)
 - ▶ notice: less than warning
 - ▶ info: informational messages, no errors
 - ▶ debug: everything is logged
- ▶ Syslogd manages severity levels in a hierarchical order from down upwards; for example the crit level will also include the alert and emerg, while the emerg will only log emergency messages



Actions



- The following actions are available to syslog when it encounters an event that matches the configured severity level:
 - `/path/to/file`: a filename where the message will get logged
 - `@hostname/ip`: the resolvable name or the IP address of a machine to where the messages will be directed (use `@@` to use TCP instead of the default UDP)
 - `| fifo`: the fifo could be any socket file (named pipe). Other applications can use this socket to read messages emitted by syslog.
 - `username(s)`: the messages are written on the screen of the user. Usernames are separated by commas.
 - `*`: the messages are sent to the screens of all the logged in users

Lab: Configuring a central logging host

- In the `/etc/rsyslog.conf` of the host, uncomment the following lines:
`$ModLoad imudp`
`$UDPServerRun 514`
- In the client machine, edit the `/etc/rsyslog.conf` and add the following line
`*.* @192.168.0.101`
- Restart rsyslog daemon on both machines to reflect the changes
`service rsyslog restart`
- Test central logging by sending a log message using the logger command:
`logger -p info "Log message from $HOSTNAME"`
- Examine the last contents of `/var/log/messages` to ensure that the message have been logged
`tail /var/log/messages`

Lab: using LogAnalyzer for central log management



- Prerequisites: install the following packages using yum or apt-get
 - MySQL server (mysql-server)
 - Apache web server (httpd on RedHat and apache2 on Debian)
 - PHP (php on RedHat and php5 on Debian)
 - Rsyslog-mysql package
- Start the mysql, httpd, and rsyslog. Use chkconfig to make them start at boot
- Create the required rsyslog MySQL database

```
mysql -u root -p < /usr/share/doc/rsyslog-mysql-5.8.10/createDB.sql
```
- Create user rsyslog and grant it privileges on the database

```
GRANT ALL ON Syslog.* TO 'rsyslog'@'localhost' IDENTIFIED BY rsyslogPassword; FLUSH PRIVILEGES;
```
- Add the required module to /etc/rsyslog.conf

```
$ModLoad ommysql
```
- Configure rsyslog to direct messages to the database

```
*.* :ommysql:127.0.0.1,Syslog,rsyslog,rsyslogPassword
```

- 
- 
- Download the latest LogAnalyzer package from <http://loganalyzer.adiscon.com/downloads/>
 - Move the contents of the src directory to the webserver public directory (/var/www/html on RedHat)
 - Change the ownership of the files to be owned by the webserver
 - Navigate to the webserver root directory from the browser and follow installation steps



Log rotation

- Linux includes the `/usr/sbin/logrotate` tool that is used to rotate log files (rename old files after a certain file size, and create a new file where logging continues)
- The configuration options are placed in `/etc/logrotate.conf`
- A daily cron job runs the `logrotate` command to be applied on all log files. Their configuration is placed in `/etc/logrotate.d/`
- The following are important options in the configuration file:
 - `size`: the size threshold after which the log file is renamed
 - `create permissions user group`: the new file permissions and owner user and group
 - `rotate`: the number of log files to be kept. Older files are deleted. If set to 0 files are deleted instead of being rotated
 - `copytruncate`: will copy the original file (instead of renaming it) to the incremented filename and truncate the original file. This avoids letting the process continue writing to the renamed file.
 - `compress`: the rotated file will be compressed using `gzip` to save size
 - `dateext`: will add a date and time stamp to the rotated file extension
 - `daily/weekly/monthly`: `logrotate` will run only if it is the first hour of the day, first day of the week or first day of the month respectively