# Introduction

Linux applications use the syslog utility to export all their errors and status messages to files located in the /var/log directory.

# syslog

syslog is a utility for tracking and logging all manner of system messages from the merely informational to the extremely critical. Each system message sent to the syslog server has two descriptive labels associated with it that makes the message easier to handle.

1. The function (facility) of the application that generated it. For example, applications such as mail and cron generate messages with easily identifiable facilities named mail and cron.
2. The degree of severity of the message. There are eight in all and they are listed in Table 5-1:

Configure syslog's /etc/rsyslog.conf configuration file to place messages of differing severities and facilities in different files.

## The /etc/rsyslog.conf File

The files to which syslog writes each type of message received is set in the /etc/rsyslog.conf configuration file.

This file consists of two columns. The first lists the facilities and severities of messages to expect and the second lists the files to which they should be logged. By default, RedHat/Fedora's /etc/rsyslog.conf file is configured to put most of the messages in the file/var/log/messages. Here is a sample:

\*.info;mail.none;authpriv.none;cron.none /var/log/messages

In this case, all messages of severity "info" and above are logged, but none from the mail, cron or authentication facilities/subsystems.

\*.debug /var/log/messages

This example may be more suitable for troubleshooting.

\*.=debug;\

auth,authpriv.none;\

news.none;mail.none -/var/log/debug

In this example, all debug severity messages; except auth, authpriv, news and mail; are logged to the /var/log/debug file in caching mode. Notice how you can spread the configuration syntax across several lines using the slash (\) symbol at the end of each line.

\*.=info;\*.=notice;\*.=warn;\

auth,authpriv.none;\

cron,daemon.none;\

mail,news.none -/var/log/messages

Here we see the /var/log/messages file configured in caching mode to receive only info, notice and warning messages except for the auth, authpriv, news and mail facilities.

\*.emerg \*

You can even have certain types of messages sent to the screen of all logged in users. In this example messages of severity emergency and above triggers this type of notification. The file definition is simply replaced by an asterisk to make this occur.

## Activating Changes to the syslog Configuration File

Changes to /etc/rsyslog.conf will not take effect until you restart syslog.

1. Different Linux distributions use different daemon management systems. Each system has its own set of commands to do similar operations. The most commonly used daemon management systems are SysV and Systemd.
2. The daemon name needs to be known. In this case the name of the daemon is **rsyslog**.

Armed with this information you can know how to:

* Start your daemons automatically on booting
* Stop, start and restart them later on during troubleshooting or when a configuration file change needs to be applied."

## How to View New Log Entries as They Happen

If you want to get new log entries to scroll on the screen as they occur, then you can use this command:

[root@bigboy tmp]# tail -f /var/log/messages

Similar commands can be applied to all log files. This is probably one of the best troubleshooting tools available in Linux. Another good command to use apart from tail is grep. grep will help you search for all occurrences of a string in a log file; you can pipe it through the more command so that you only get one screen at a time. Here is an example:

[root@bigboy tmp]# grep string /var/log/messages | more

## Logging syslog Messages to a Remote Linux Server

Logging your system messages to a remote server is a good security practice. With all servers logging to a central syslog server, it becomes easier to correlate events across your company. It also makes covering up mistakes or malicious activities harder because the purposeful deletion of log files on a server cannot simultaneously occur on your logging server, especially if you restrict the user access to the logging server.

### Configuring the Linux Syslog Server

By default syslog doesn't expect to receive messages from remote clients. Here's how to configure your Linux server to start listening for these messages.

You will have to restart syslog on the server for the changes to take effect. The server will now start to listen on UDP port 514, which you can verify using either one of the following netstat command variations.

[root@bigboy tmp]# netstat -a | grep syslog

udp 0 0 \*:syslog \*:\*

[root@bigboy tmp]# netstat -an | grep 514

udp 0 0 0.0.0.0:514 0.0.0.0:\*

[root@bigboy tmp]#

Go to the website <http://wiki.gentoo.org/wiki/Rsyslog#Severity>for more information.

*Rsyslog is an open source software used on UNIX and Unix-like computer systems for forwarding log messages in an IP network. It implements the basic syslog protocol, extends it with content-based filtering, rich filtering capabilities, flexible configuration options and adds important features such as using TCP for transport.*

## Installation

**Note**

It is a bad idea to run several system loggers on one physical host. You should remove or at least disable other local loggers.

| **USE flag** | **Default** | **Description** |
| --- | --- | --- |
| dbi | No | Enable [dev-db/libdbi](http://packages.gentoo.org/package/dev-db/libdbi) (database-independent abstraction layer) support |
| debug | No | Enable extra debug codepaths, like asserts and extra output. If you want to get meaningful backtraces see<http://www.gentoo.org/proj/en/qa/backtraces.xml> |
| doc | No | Adds extra documentation (API, Javadoc, etc). It is recommended to enable per package instead of globally |
| extras | No | Add support for the UDP spoofing module (omudpspoof) using [net-libs/libnet](http://packages.gentoo.org/package/net-libs/libnet) |
| kerberos | No | Adds kerberos support |
| mysql | No | Adds mySQL Database support |
| oracle | No | Enable Oracle Database support |
| postgres | No | Adds support for the postgresql database |
| relp | No | Add support for the Reliable Event Logging Protocol using [dev-libs/librelp](http://packages.gentoo.org/package/dev-libs/librelp) |
| snmp | No | Adds support for the Simple Network Management Protocol if available |
| static-libs | No | Build static libraries |
| zeromq | No | Add support for the ZeroMQ input and output plugins using [net-libs/zeromq](http://packages.gentoo.org/package/net-libs/zeromq) |
| ssl | No | Adds support for Secure Socket Layer connections |
| zlib | No | Adds support for zlib (de)compression |

**root #** emerge --ask rsyslog

Add rsyslog to default startup level

**root #** rc-update add rsyslog default

After emerge has finished, rsyslog should work out of the box with the shipped default configuration, at least for local logging. Start rsyslog

**root #** /etc/init.d/rsyslog start

Check file /var/log/messages for syslog entries:

**root #** tail -f /var/log/messages

2011-07-24T00:31:06.268926+02:00 server kernel: imklog 5.8.1, log source = /proc/kmsg started.

2011-07-24T00:31:06.269053+02:00 server rsyslogd: [origin software="rsyslogd" swVersion="5.8.1" x-pid="19116" x-info="http://www.rsyslog.com"] start

## Configuration

Basic rsyslog configuration:

**File/etc/rsyslog.conf**

# Support for Local System Logging

$ModLoad imuxsock.so

$IncludeConfig /etc/rsyslog.d/\*

\*.info;mail.none;authpriv.none;cron.none -/var/log/messages

authpriv.\* /var/log/secure

mail.\* -/var/log/maillog

cron.\* -/var/log/cron

\*.emerg \*

uucp,news.crit -/var/log/spooler

local7.\* /var/log/boot.log

Typically messages are logged to files, the file has to be specified with full pathname. Rsyslog uses a simple syntax to filter incoming messages. Syslog messages are classified by facility and severity defined below:

### Severity

| **Severity** | | |
| --- | --- | --- |
| **Numerical Code** | **Severity** | **Description** |
| 0 | emerg | system is unusable |
| 1 | alert | action must be taken immediately |
| 2 | crit | critical conditions |
| 3 | error | error conditions |
| 4 | warning | warning conditions |
| 5 | notice | normal but significant condition |
| 6 | info | informational messages |
| 7 | debug | debug-level messages |

### Facility

List of facilities used by rsyslog. Most facilities names are self-explanatory. Facilities local0 - local7 common usage is f.e. as network logs facilities for nodes and network equipment. Generally it depends on the situation how to classify logs and put them to facilities.

| **Facility** | | |
| --- | --- | --- |
| **Numerical Code** | **Facility** | **Description** |
| 0 | Kern | kernel messages |
| 1 | User | user-level messages |
| 2 | Mail | mail system |
| 3 | Daemon | system daemons |
| 4 | Auth | security/authorization messages |
| 5 | Syslog | messages generated internally by syslogd |
| 6 | Lpr | line printer subsystem |
| 7 | News | network news subsystem |
| 8 | Uucp | UUCP subsystem |
| 9 | Cron | clock daemon |
| 10 | Security | security/authorization messages |
| 11 | ftp | FTP daemon |
| 12 | Ntp | NTP subsystem |
| 13 | Logaudit | log audit |
| 14 | Logalert | log alert |
| 15 | Clock | clock daemon (note 2) |
| 16 | local0 | local use 0 (local0) |
| 17 | local1 | local use 1 (local1) |
| 18 | local2 | local use 2 (local2) |
| 19 | local3 | local use 3 (local3) |
| 20 | local4 | local use 4 (local4) |
| 21 | local5 | local use 5 (local5) |
| 22 | local6 | local use 6 (local6) |
| 23 | local7 | local use 7 (local7) |

### Filtering

List of filtering examples

Redirect all incoming messages from all facilities and with all severeties to /var/log/syslog

*\*.\* -/var/log/syslog*

Filter out messages with severity critical and save to file /var/log/critical

*\*.crit -/var/log/critical*

Do NOT redirect facilities mail, authentication and cron and mail to /var/log/messages, look for the keyword **none**

*mail.none;authpriv.none;cron.none -/var/log/messages*

### Local Logging

Enable local logging from all facilities, to see local events at all. *$ModLoad imuxsock.so*

### Remote Logging

To use remote logging to a syslog server you need to specify a client to log to a specific server or servers. And a server to receive messages sent by clients. Before configuring you have to choose the protocol. Syslog messages can be sent with UDP or with TCP. UDP is the default protocol and supported on most platforms. Not all platforms support TCP for syslog.

#### Client

To enable syslog UDP messages sending add following line to your /etc/rsyslog.conf file. In this example rsyslog sends all facilities and all priorities *\*.\** using protocol UDP *@* to remote server*192.168.20.254*

*\*.\* @192.168.20.254*

To enable TCP support for syslog messages, put following line to the rsyslog configuraton file, TCP is enabled by adding *@@*.

*\*.\* @@192.168.20.254*

You can use also a hostname instead of a IP address.

**Important**

Substitute **192.168.20.254** with the IP address of your own rsyslog server

Below an example syslog client configuration to send syslog messages to a remote server via TCP.

**File/etc/rsyslog.conf**

$ModLoad imuxsock.so

\*.\* @@192.168.20.254:10514

#### Server

To Provide UDP log reception and run the server on port 514. Running syslog with UDP is the default configuration.

*$ModLoad imudp* *$UDPServerRun 514*

UDP is not a reliable protocol, you could run the server with TCP logging support.

*$ModLoad imtcp* *$InputTCPServerRun 10514*

A simple configuration would look like this one:

**File/etc/rsyslog.conf**

$ModLoad imuxsock.so

$ModLoad imtcp.so

$InputTCPServerRun 10514

$ModLoad imudp.so

$UDPServerRun 514

### Database Logging

Rsyslog supports logging to MySQL databases

After choosing the database logs will be stored to a proper USE flag needs to be enabled and rsyslog has to be rebuild before continuing. This example uses a MySQL database.

**Note**

Next steps assume a working MySQL database server running on localhost, for installation details follow the [MySQL](http://wiki.gentoo.org/wiki/MySQL) article.

The package ships a the file with a SQL script named createDB.sql to create the database layout.

**File/usr/share/doc/rsyslog-5.8.1/scripts/mysql/createDB.sql**

CREATE DATABASE Syslog;

USE Syslog;

CREATE TABLE SystemEvents

(

ID int unsigned not null auto\_increment primary key,

CustomerID bigint,

ReceivedAt datetime NULL,

DeviceReportedTime datetime NULL,

Facility smallint NULL,

Priority smallint NULL,

FromHost varchar(60) NULL,

Message text,

NTSeverity int NULL,

Importance int NULL,

EventSource varchar(60),

EventUser varchar(60) NULL,

EventCategory int NULL,

EventID int NULL,

EventBinaryData text NULL,

MaxAvailable int NULL,

CurrUsage int NULL,

MinUsage int NULL,

MaxUsage int NULL,

InfoUnitID int NULL ,

SysLogTag varchar(60),

EventLogType varchar(60),

GenericFileName VarChar(60),

SystemID int NULL

);

CREATE TABLE SystemEventsProperties

(

ID int unsigned not null auto\_increment primary key,

SystemEventID int NULL ,

ParamName varchar(255) NULL ,

ParamValue text NULL

);

Import the /usr/share/doc/rsyslog-5.8.1/scripts/mysql/createDB.sql file to create the "Syslog" database.

**user $** mysql -u root -p < /usr/share/doc/rsyslog-5.8.1/scripts/mysql/createDB.sql

Create a datebase user for the Syslog database:

mysql> grant ALL ON Syslog.\* to rsyslog-user@localhost identified by - 'MySecretPassword'; flush privileges;

To provide SQL database logging support, enable the needed module in /etc/rsyslog.d/mysql.conf

*$ModLoad ommysql.so*

Tell rsyslog to forward all data to the database, add following to the end of the /etc/rsyslog.d/mysql.conf file:

*\*.\* :ommysql:localhost,Syslog,rsyslog-user,MySecretPassword*

Finally Restart the rsyslog server to adapt new settings

**root #** /etc/init.d/rsyslog restart

## Templates

If your network equipment logs into a central rsyslog server you will notice many vendors format its syslog messages differently. So it will be difficult after some time to for you filter the syslog server messages for a certain

* Date
* Facility
* Severity
* Host
* Syslogtag
* ProcessID
* MessageType
* Message

as they arrive at your own syslog server.

To unify syslog messages to a certain or preferred format, Rsyslog uses templates which parse arriving messages and "rewrites" them to the desired format. To maintain a simple and modular configuration, templates are stored within the /etc/rsyslog.d/ directory. To include files stored within the rsyslog.d directory add following line to/etc/rsyslog.conf file:

*$IncludeConfig /etc/rsyslog.d/\*.conf*

Templates should be stored to the /etc/rsyslog.d/ directory.

**root #** cd /etc/rsyslog.d/

Here a simple template for a cisco IOS host which logs to rsyslogd:

**File/etc/rsyslog.d/template\_cisco.conf**  
$template mysql\_cisco, "insert into SystemEvents (Message, Facility, FromHost, Priority, DeviceReportedTime, ReceivedAt, InfoUnitID, SysLogTag) values ('%msg:R,ERE,1,DFLT:%[A-Z0-9\_-]+: (.\*)--end%', %syslogfacility%, '%fromhost%', %syslog

priority%, '%timereported:::date-mysql%', '%timegenerated:::date-mysql%', %iut%, '%msg:R,ERE,0,DFLT:%[A-Z0-9\_-]+:--end%')",SQL

Here a simple template for a ScreenOS host which logs to rsyslogd:

**File/etc/rsyslog.d/template\_netscreen.conf**  
$template mysql\_netscreen, "insert into SystemEvents (Message, Facility, FromHost, Priority, DeviceReportedTime, ReceivedAt, InfoUnitID, SysLogTag) values ('%msg:R,ERE,1,DFLT:[a-zA-Z0-9-]+: (.\*)--end%', %syslogfacility%, '%fromhost%', %s

yslogpriority%, '%timereported:::date-mysql%', '%timegenerated:::date-mysql%', %iut%, '%msg:R,ERE,0,DFLT:[a-zA-Z0-9-]+:--end%')",SQL

Here a simple template for Linux host which logs to rsyslogd:

**File/etc/rsyslog.d/template\_linux.conf**  
$template mysql\_linux,"insert into SystemEvents (Message, Facility, FromHost, Priority, DeviceReportedTime, ReceivedAt, InfoUnitID, SysLogTag, ProcessID) values ('%msg%', %syslogfacility%, '%HOSTNAME%',%syslogpriority%, '%timereported:::

date-mysql%', '%timegenerated:::date-mysql%', %iut%, '%syslogtag:R,ERE,1,FIELD:(.+)(\[[0-9]{1,5}\]).\*--end%', '%syslogtag:R,ERE,1,BLANK:\[([0-9]{1,5})\]--end%')",SQL

Configure rsyslogd which predefined template to apply to which facility, add following template references to the end of the /etc/rsyslog.conf file:

* All messages arriving at facility **local4**, are Cisco IOS messages:

*local4.\* :ommysql:localhost,Syslog,rsyslog-user,MySecretPassword;mysql\_cisco*

* All messages arriving at facility **local5** , are ScreenOS messages:

*local5.\* :ommysql:localhost,Syslog,rsyslog-user,MySecretPassword;mysql\_netscreen*

* All messages arriving at syslog consider as Linux messages, and ignore **local4** and **local5** facilities which have their own templates.

*\*.\*;local4.none;local5.none :ommysql:localhost,Syslog,rsyslog-user,MySecretPassword;mysql\_linux*

Example how the /etc/rsyslog.conf file could look on your own syslog **server** with working templates:

**File/etc/rsyslog.conf**

$ModLoad imudp

$UDPServerRun 514

$ModLoad ommysql.so

$IncludeConfig /etc/rsyslog.d/\*.conf

\*.info;mail.none;authpriv.none;cron.none -/var/log/messages

authpriv.\* /var/log/secure

mail.\* -/var/log/maillog

cron.\* -/var/log/cron

\*.emerg \*

uucp,news.crit -/var/log/spooler

local7.\* /var/log/boot.log

local4.\* :ommysql:localhost,Syslog,rsyslog-user,MySecretPassword;mysql\_cisco

local5.\* :ommysql:localhost,Syslog,rsyslog-user,MySecretPassword;mysql\_netscreen

\*.\*;local4.none;local5.none :ommysql:localhost,Syslog,rsyslog-user,MySecretPassword;mysql\_linux

Reload rsyslog server to apply new changes. Further examples can be found [here](http://www.rsyslog.com/doc/rsyslog_conf_templates.html).

**root #** /etc/init.d/rsyslog reload

## Troubleshooting

Check if a syslog process is running

**root #** ps ux | grep rsyslog

root 9161 0.0 0.0 1323652 3424 ? Sl 00:51 0:00 /usr/sbin/rsyslogd -c5 -i /var/run/rsyslogd.pid -f /etc/rsyslog.conf

Verify network configuration

**root #** netstat -tulpen | grep rsyslog

udp 0 0 0.0.0.0:514 0.0.0.0:\* 0 33472286 9161/rsyslogd

udp6 0 0 :::514  :::\* 0 33472287 9161/rsyslogd

Verify with the command logger, if messages are arriving in at the syslog server

**user $** logger -t test my syslog-test-message

Following message should appear in the /var/log/messages file if rsyslog is working properly:

**root #** tail /var/log/messages

...

2011-11-23T00:47:05+01:00 Rsyslogserver test: my syslog-test-message

Installing rsyslog on Ubuntu is easy:

$ sudo aptitude install rsyslog

To get extra documentation:

$ sudo aptitude install rsyslog-doc

Examples:

1. A rule that sends messages for one facility and of any priority to file:

daemon.\* /var/log/daemon.log

2. A rule combination that sends messages for two facilities and of any priority to one file

and then everything else to another:

auth,authpriv.\* /var/log/auth.log

\*.\*;auth,authpriv.none /var/log/syslog

3. A selector that picks out messages for one facility and of a single priority:

mail.=info

4. A compound selector that picks out messages for all facilities but only of priority info,

notice and warning:

\*.info;\*.!err

5. A rule that sends messages for one facility and of all priorities except that of warning:

kern.\*;kern.!=warn /var/log/kernel/nowarnings

The terms "rule" and an "action" are often used synonymously. A rule *defines* an action.

Queues and queue parameters are covered later in this paper but for now we need to

Centralised logging with rsyslog 13 www.canonical.com

remember that:

• An *action queue* is created each time an action is specified.

• Action queue *parameters* are reset after an action queue has been created (allowing

the creation of a new action queue and its corresponding parameters).

**Output file syncing**

Due to performance degradation, rsyslog no longer retains sysklog's default of file syncing 1 if

not specified otherwise (by placing a dash in front of the output file name). Even if rsyslog finds

sync selector lines it will ignore them. In order to enable file syncing you must explicitly do so at

the top of your configuration file:

$MainMsgFileEnableSync on

**Timestamps**

Rsyslog comes with high-precision timestamp support (disabled by default). This feature is

controlled by the following parameter:

$MainMsgFileDefaultTemplate RSYSLOG\_TraditionalFileFormat

Comment it out in order to gain this feature (turn on high-precision timestamps).

Here is a comparison of traditional and high-precision timestamps, respectively:

Nov 17 12:01:44 client\_hostname ubuntu: test

2008-11-17T12:02:47.372490-05:00 client\_hostname ubuntu: test

**Templates**

Templates in rsyslog are user-definable formats that are applied to message queues. A

template is associated with every type of output and can also be used for dynamic file name

generation. In particular, for database logging,

a template is used that converts a message into a proper SQL statement.

Each client system should have rsyslog installed. The server software does not need to be rsyslog as long as it is compatible with the chosen transport protocol. All configuration, with the exception of the above defaults file, is placed in the */etc/rsyslog.conf* file or in files found under the */etc/rsyslog.d* directory.

You first need to decide what protocol you want to use: UDP, TCP.

On the server, assuming you are running rsyslog, you do this by enabling the appropriate input module, as well as specifying the port to be used:

UDP:

$ModLoad imudp

$UDPServerRun 514

TCP:

$ModLoad imtcp

$InputTCPServerRun 514

On the client, you specify either the UDP or TCP protocol with the '@' or '@@' characters respectively.

Local logging functionality is provided by the **imuxsock** plug-in (also enabled by default).

$ModLoad imuxsock

$ModLoad imklog

Naturally, any configuration changes require a restart:

$ sudo /etc/init.d/rsyslog restart

All modules and global directives need to be specified one per line and must start with a dollar sign ($).

Selectors can contain the special keywords ' \* ' and 'none', meaning all or nothing, respectively.

A selector can include multiple facilities separated with commas. Multiple selectors can also be combined with semicolons.

1. A rule that sends messages for one facility and of any priority to file:

daemon.\* /var/log/daemon.log

2. A rule combination that sends messages for two facilities and of any priority to one file

and then everything else to another:

auth,authpriv.\* /var/log/auth.log

\*.\*;auth,authpriv.none /var/log/syslog

3. A selector that picks out messages for one facility and of a single priority:

mail.=info

4. A compound selector that picks out messages for all facilities but only of priority info,

notice and warning:

\*.info;\*.!err

5. A rule that sends messages for one facility and of all priorities except that of warning:

kern.\*;kern.!=warn /var/log/kernel/nowarnings

The terms "rule" and an "action" are often used synonymously. A rule *defines* an action.

Queues and queue parameters are covered

Examples:

• Here we are directing the client to forward all its logs via UDP to the logging server

whose IP address is 192.168.0.1:

\*.\* @192.168.0.1:514

• The same but over TCP:

\*.\* @@192.168.0.1:514

First, install the module on the logging server:

$ sudo aptitude install rsyslog-mysql

Next, the server should load the output module **ommysql** and be configured to connect to the database. Its configuration should be similar to the following:

$ModLoad ommysql

\*.\* :ommysql:localhost,Syslog,rsyslog,abc123

This configuration is set up automatically when you install the above package. It is implemented as an include file found: */etc/rsyslog.d/mysql.conf .*

In addition, the MySQL database creation process is also automated. The newly created database is called 'Syslog', containing two tables: 'SystemEvents' and 'SystemEventsProperties'. The installation process will prompt you for a password (user 'rsyslog' is used by the software to access the database). A password will be generated if you do not provide one and the credentials end up in the configuration above (i.e. I provided the

password).