Hands-on Lab Description



2020 Copyright Notice: The lab materials are only used for education purpose. Copy and redistribution is prohibited or need to get authors’ consent.

Please contact Professor Dijiang Huang: [Dijiang.Huang@asu.edu](mailto:Dijiang.Huang@asu.edu)

*CS-SYS-00009 –*

*Syslog (rsyslogd) Remote Logging on Linux*

## CONTENTS

[1 Selectors . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4](#_TOC_250009)

[1.1 Facility . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4](#_TOC_250008)

[1.2 Severity Levels . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4](#_TOC_250007)

[1.3 Log files in Linux File System . . . . . . . . . . . . . . . . . . . . . . . . . . . 4](#_TOC_250006)

[2 Task 1 Preparation of rsyslog . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7](#_TOC_250005)

[3 Task 2 Configure rsyslog server . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8](#_TOC_250004)

[4 Task 3 Configure rsyslog client . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9](#_TOC_250003)

[4.1 Task 4 Customaize rsyslogd configurations . . . . . . . . . . . . . 9](#_TOC_250002)

1. [Task 5 Log iptables messages to a remote server . . . . . . . . . . . . . . . . 9](#_TOC_250001)
2. [Related Information and Resource 10](#_TOC_250000)

1

### Category:

CS-SYS: Computer System

### Objectives:

1

Learn System Logging Service basis

Learn how to set up rsyslog services in a network setup

2

### Estimated Lab Duration:

1

Expert: 20 minutes

Novice: 100 minutes

2

### Difficulty Diagram:

Implementation



Design 5

4

3

2

1

1

Time

**Difficulty Table.**

Time 1

Measurements Values (0-5)

Design 0

Implementation 0

Configuration 2

Knowledge 2

Score (Average) 1

Configuration

Knowledge

### Required OS:

Linux: Ubuntu 18.04 LTS

### Lab Running Environment:

ThoTh Lab: https://thothlab.org



Client

Server



Network

1

Client: Linux (Ubuntu 18.04 LTS)

Server: Linux (Ubuntu 18.04 LTS)

Network Setup: connected through a local network

2

3

**Lab Preparations:**

Initial setup: basic Ubuntu 18.04 LTS is required for this lab

Basic Linux knowledge and operations. Reference Lab: CS-SYS-00001 (Linux Tutorial) and CS-SYS-00008 (Syslog Remote Logging on Linux).

**Lab Overview**

In ubuntu distribution, *rsyslogd* is the default system log implementation compared to its precursor syslogd. In this lab, you will learn and practice on how to setup rsyslog forwarding from a client to a centralized server. The syslog bases are presented in the lab lab-cs-sys-00008 (Syslog (syslogd) Local Logging on Linux). Since rsyslog is a successor of syslog, the basic concepts of log facilities, actions, and priority are the same. On Ubuntu system, instead of using *syslog.conf*, rsyslog uses *rsyslog* as it configuration file. Detailed rsyslog configuration files are maintained under */etc/rsyslog.d/* folder. Default rsyslog setup configurations are maintained in the file *50-default.conf*, which is similar to *syslog.conf* using *syslogd*. In the following sections Selectors, Facility,

and Severity Levels, we use the same descriptions from the lab lab-cs-sys-00008 (Syslog (syslogd) on Linux).

# Selectors

The selector field itself again consists of two parts, a *facility* and a *priority*, separated by a period (“.”). Both parts are case insensitive. For example,

Kern.none, mail.info, etc.

Here,

Kern = Facility

None = severity or priority

## Facility

The facility is one of the following keywords: *auth*, *authpriv*, *cron*, *daemon*, *kern*, *lpr*, *mail*, *mark*, *news*, *security* (same as *auth*), *syslog*, *user*, *uucp* and *local0* through *local7*. The keyword *security* should not be used anymore and mark is only for internal use and therefore should not be used in applications.

You may want to specify and redirect these messages here. The facility specifies the subsystem that pro- duced the message, i.e., all mail programs *log* with the *mail* facility (*LOG MAIL*) if they log using *syslog*. The facility numbers are given in Table CS-SYS-00009.1.

## Severity Levels

The priority is one of the following keywords, in ascending order: *debug, info, notice, warning, warn* (same as *warning*), *err, error* (same as *err*), *crit, alert, emerg, panic* (same as *emerg*). The keywords *error, warn* and *panic* are deprecated and should not be used anymore. The priority defines the severity of the message, which is presented in Table CS-SYS-00009.2.

You can specify multiple facilities with the same priority pattern in one statement using the comma (“,”) operator. You may specify as much facilities as you want. Multiple selectors may be specified for a single action using the semicolon (“;”) separator. Remember that each selector in the selector field is capable to overwrite the preceding ones. Using this behavior you can exclude some priorities from the pattern.

## Log files in Linux File System

Logs are stored at */var/log/* on the Linux filesystem. At this location, you should see multiple log files, each one having a name describing what they actually store. For example, you can execute the following command:

$ ls -l /var/log

-rw-r----- 1 syslog

adm

120999 Jul 24 18:04 auth.log

### Table CS-SYS-00009.1

Facility Number.

|  |  |  |
| --- | --- | --- |
| Keyword | 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 | clock daemon |  |
| 10 | authpriv | security/authorization messages |
| 11 | ftp | FTP daemon |
| 12 | - | NTP susbsystem |
| 13 | - | log audit |
| 14 | - | log alert |
| 15 | cron | clock daemon |
| 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) |

|  |  |  |
| --- | --- | --- |
| -rw-r--r-- 1 root | root | 127503 Jul 20 06:35 dpkg.log |
| -rw-r----- 1 syslog | adm | 0 Jul 15 06:25 kern.log |
| drwxrwxr-x 2 logstash | root | 4096 Jul 8 18:33 logstash |
| drwxr-xr-x 2 root | root | 4096 Sep 10 2018 lxd |
| drwxr-xr-x 2 mongodb | mongodb | 4096 Jul 8 06:25 mongodb |

In this example, you have dedicated log files for authentication purposes or for kernel related logs. The folder */var/log* does not contain only files, but it also contain dedicated folders that vendors create when the application is installed. Suppose that three machines are sending logs to the log server, each machine is going to have its own own *auth.log*, *kern.log* or *dpkg.log* files. As a consequence, you may want logs to be stored in dedicated folders, one for each instance. A typical folder architecture on the server side is presented in Figure.

### Table CS-SYS-00009.2

Facility Number.

|  |  |
| --- | --- |
| Integer | Facility |
| 0 | Emergency: System is unusable |
| 1 | Alert: Action must be taken immediately |
| 2 | Critical: critical conditions |
| 3 | Error: Error conditions |
| 4 | Warning: Warning conditions |
| 5 | Notice: Normal but significant conditions |
| 6 | Informational: Informational messages |
| 7 | Debug: Debug level messages |

var log

Client-1 Client-2

Client-3

………

.

.

.

Client-n

auth.log kern.log authpriv.log syslog.log daemon.log mail.log cron.log user.log

Syslog Facilities

### Figure CS-SYS-00009.1

Divided by client hosts

Centralized logging: folder architecture.

Each machine (i.e., a client) wants to write logs via *rsyslog*. To this end, each machine is configured as client-server syslog instances. They can create logs and store logs locally in their file system. In the centralized logging architecture, client machines will be configured to use *rsyslog* as a client, and they will forward every single log to a remote *rsyslog* server, as shown in Figure CS-SYS-00009.2, which is the central server.

Certificate Authority

issues certificates to

Client #1

sends logs to

rsyslog

sends logs to

Memory Action Queue

Disk

Client #2

sends logs to

rsyslog

Client #3

TCP forwarding module

sends logs to

rsyslog

TLS/SSL Centralized

Forward Server encrypted

messages to

Client #n

sends logs to

rsyslog

Optional back-pressure protocol

Local processes

### Figure CS-SYS-00009.2

………

Centralized logging Architecture.

Compared to *syslog*, *rsyslog* tries to address the following improved features:

* Secure transmitted log data: implement TLS to protect data transmission;
* Increase the reliability: implement a memory caching module to address when the network is down. Using the centralized logging architecture, we can achieve the following benefits:
* Inspect all your logs from a single place instead of accessing individual machine.
* Access to logs without access privilege to individual machines.

Set up a complete log monitoring infrastructure in one place to improve the nentwork management capability.

*•*

The downside of using centralized logging server is summarized as follows:

The syslog server can be overloaded: with this architecture, you are pushing logs to a remote server. As a consequence, if one machine starts sending excessive a mount of log data, it may overload the log server.

*•*

The centralized system suffers single-point failure. If the log server goes down, then the ability to look at all the logs sent by the clients is lost. Moreover, if the server goes down, clients will start storing messages locally until the server is available again, thus it may fill up disk space on the client side.

*•*

# Task 1 Preparation of rsyslog

The implementation of *syslog* on Unbuntu is *rsyslogd*. To determine if *rsyslogd* is installed on Ubuntu, you can issue the following command:

$ rsyslogd -v

$ sudo systemctl status rsyslog % check rsyslog running status

A common *rsyslogd* version reply will be looked like the following

rsyslogd 8.32.0, compiled with:

PLATFORM: x86\_64-pc-linux-gnu PLATFORM (lsb\_release -d): FEATURE\_REGEXP: Yes

GSSAPI Kerberos 5 support: Yes FEATURE\_DEBUG (debug build, slow code): No 32bit Atomic operations supported: Yes 64bit Atomic operations supported: Yes memory allocator: system default Runtime Instrumentation (slow code): No uuid support: Yes

systemd support: Yes

Number of Bits in RainerScript integers: 64

See [http://www.rsyslog.com](http://www.rsyslog.com/) for more information.

To check rsyslog running status:

$ sudo systemctl status rsyslog

A common *rsyslogd* status checking reply will be looked like the following

See [http://www.rsyslog.com](http://www.rsyslog.com/) for more information.

rsyslog.service - System Logging Service

Loaded: loaded (/lib/systemd/system/rsyslog.service; enabled; vendor preset: enabled)

Active: active (running) since Tue 2020-09-29 15:54:05 UTC; 1h 0min ago Docs: man:rsyslogd(8)

<http://www.rsyslog.com/doc/> Main PID: 21503 (rsyslogd)

Tasks: 4 (limit: 2317)

CGroup: /system.slice/rsyslog.service

--21503 /usr/sbin/rsyslogd -n

Sep 29 15:54:05 ubuntu systemd[1]: Starting System Logging Service... Sep 29 15:54:05 ubuntu systemd[1]: Started System Logging Service.

If not, you can install *rsyslogd*. At a terminal prompt, you can enter the following command:

$ sudo apt update && apt install rsyslog

$ sudo systemctl enable rsyslog

$ sudo systemctl start rsyslog

# Task 2 Configure rsyslog server

First of all, you have to configure your rsyslog server for it to accept incoming logs on port 514. We are going to use TCP for log transmission. You can use UDP if the network reliability is not a big concern. On the server, go to */etc/rsyslog.d* folder. This is the directory that stores templates as well as files that contain the *rsyslog* rules.

In this directory, there is a *50-default.conf* file. You are going to create your own configuration file, prefixing

it with a number lower than the configuration file one. In this way, your configuration file takes priority over the default one.

$ sudo touch /etc/rsyslog.d/01-server.conf % create a new configuration file with a

small value

Edit the file and provide the following inputs:

# Listen for TCP

$ModLoad imtcp

# Listen on port 514

$InputTCPServerRun 514

$template RemoteServer, "/var/log/%HOSTNAME%/%SYSLOGFACILITY-TEXT%.log"

\*.\* ?RemoteServer

In this example,

* $template Remote Server means declaring a rsyslog gemplate named “RemoteServer”.

%HOSTNAME% and %SYSLOGFACILITY-TEXT% are *rsyslog* properties and they together identify the path where data will be stored on the remote server.

*•*

* . ?RemoteServer means to send for every facility, and every severity to the remote server template.

With this syntax, log files will be grouped by hostname, a.k.a., the computer name sending the log, and then by *syslog* facility (*kern, user, auth* etc.). Now, you can restart the *rsyslog* server, and make sure that it is now listening on the port 514 for TCP:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| $ sudo systemctl restart rsyslog  $ netstat -tna | grep :514 | | | | |
| tcp | 0 | 0 0.0.0.0:514 | 0.0.0.0:\* | LISTEN |
| tcp6 | 0 | 0 :::514 | :::\* | LISTEN |

With this configuration file, server logs are no longer directly stored in */var/log*, but in */var/log/hostname*, hostname being the current name of the host (for example, /var/log/ubuntu). To check the hostname:

$ uname -a

Linux ubuntu 4.15.0-42-generic #45-Ubuntu SMP Thu Nov 15 19:32:57 UTC 2018 x86\_64

x86\_64 x86\_64 GNU/Linux

# Task 3 Configure rsyslog client

Now, let’s configure *rsyslog* as a client, and on the client machine go to */etc/rsyslog.d*. Similarly to the procedure done on the server-side create a *01-client.conf* file.

$ sudo touch /etc/rsyslog.d/01-client.conf % create a new configuration file with a

small value

Edit the file and provide the following inputs:

\*.\* @@distant-server-ip:514

Restart the *rsyslog* server, and make sure that there are no errors on the client-side.

$ sudo systemctl restart rsyslog

$ journalctl -f -u rsyslog

Now, on the server side, go to */var/log* and you will find a folder *var/log/ubuntu* is created for the client with the hostname ”ubuntu”. You are successfully received the log file from the client.

## Task 4 Customaize rsyslogd configurations

Now, you can practice a customized logging service for one of your client. In this task, we assume that you want to send a subset of logs captured on your client to the log server. On your client, append the following line inside */etc/rsyslogd/01-client.conf* :

\*.=crit @@distant-server-ip:514

Assume that you are currently logging all the kernel related messages in separate log file inside */var/log/- firewall.log*. Make the following addition in */etc/rsyslogd/01-client.conf* :

# Add a new line Kern.\*

@@distant-server-ip:514

# Add a new entry at the end of the below line

# Log anything (except mail) of level info or higher. # don’t log private authentication messages!

# don’t log kernel related events and messages

\*.info;mail.none;authpriv.none;cron.none;kern.none @@distant-server-ip:514

Assume that you are currently logging all the firewall warning level messages inside */var/log/firewall- warning.log*. Make the following addition in */etc/rsyslogd/01-client.conf* :

Kern.warning @@distant-server-ip:514

Restart the *rsyslog* server, and make sure that there are no errors on the client-side.

$ sudo systemctl restart rsyslog

$ journalctl -f -u rsyslog

# Task 5 Log iptables messages to a remote server

In this task, we targets to achieve the following goals:

* How to modify the iptables rules to let it log at the appropriate level?
* How to configure syslog to log the iptables messages to a different log file?
* To stop iptables messages to get logged into a remote server ?

In the previous task, you had made the following update in the */etc/rsyslog.d/01-client.conf* :

# comment iptables log kern.warning

@@distant-server-ip:514

Now, we need to set up iptables to allow it generating log messages by doing follows.

1. Make sure the iptables rule is logging at the appropriate level. This can be done by using the log-level switch. Default log-level is warning. Please refer to Table CS-SYS-00009.2 regarding to log levels. Below example will log ssh attempts:

$ iptables -I INPUT -p tcp --dport 22 -j LOG --log-level 4

Below example use to log ping and add the prefix “#### Firewall ####”.

$

iptables -I INPUT -p icmp --icmp-type ping -j LOG --log-prefix "#### Firewall ####"

Now, you can run ssh and ping to test your firewall logs received at the server side.

Optionally, if you want to print your iptables logs on your monitoring console, you can follow the steps below:

**Step 1** Add below entry in /etc/sysctl.conf

kernel.printk = 4 1 1 7 % refer to the reference link on printk

**Step 2** Run below command to make changes effectively at runtime.

$ sudo /sbin/sysctl -p /etc/sysctl.conf

**Step 3** Check the changes at below file

$ cat /proc/sys/kernel/printk

# Related Information and Resource

Syslog Linux Man page: https://linux.die.net/man/8/syslogd Syslog Wiki: https://en.wikipedia.org/wiki/Syslog

Message logging with printk: https://[www.kernel.org/doc/html/latest/core-api/printk-basics.html](http://www.kernel.org/doc/html/latest/core-api/printk-basics.html)

Guide to Centralized Logging:

https://devconnected.com/the-definitive-guide-to-centralized-logging-with-syslog-on-linux/