Linux Security - Forwarding the Journal logs

Taz Wake | Halkyn Consulting Ltd | Originally published 2024-07-17

Recently I wrote an article about how to analyse the Systemd Journal during incident response. There was a follow-up discussion about how to make sure this data was moved to a log centralisation platform/SIEM. Now, forwarding the data is definitely a good idea as it allows you to analyse it remotely and reduces the risk that an attacker can tamper with it.

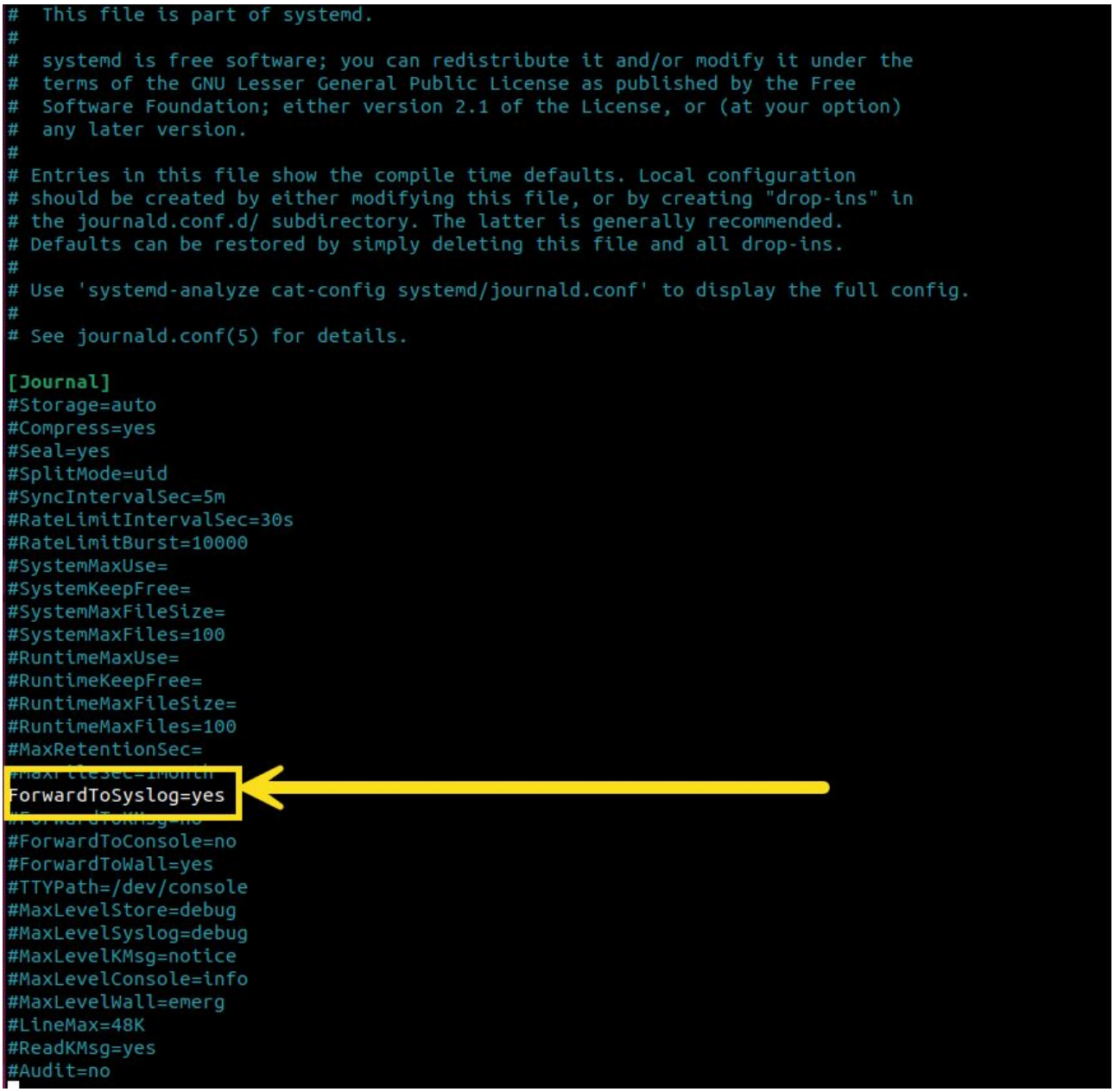
This article is a quick summary of how you can configure a Linux system to forward the logs, using either Rsyslog or Syslog-ng. I am going to use Ubuntu as the example system but it is pretty similar when using RHEL/SLES etc.

Note: This focuses on the journal, but you can also use it to forward any data you've sent to Syslog.

Forwarding the journal with rsyslog

Step 1. Make sure the journal forwards logs to rsyslog

Edit the journald.conf file to uncomment the entry which reads "ForwardToSyslog=yes" and then restart the journal. For example:



Uncomment the entry which reads "ForwardToSyslog" then save and exit the file.

You might want to also use this time to look at other ways to optimise your journal, but if you are confident this is the only change, you can even roll this into a single-line command:

You now have a journal that is sending data to syslog! What we need to do next is make syslog (or rsyslog in this case) send that data to where we need it.

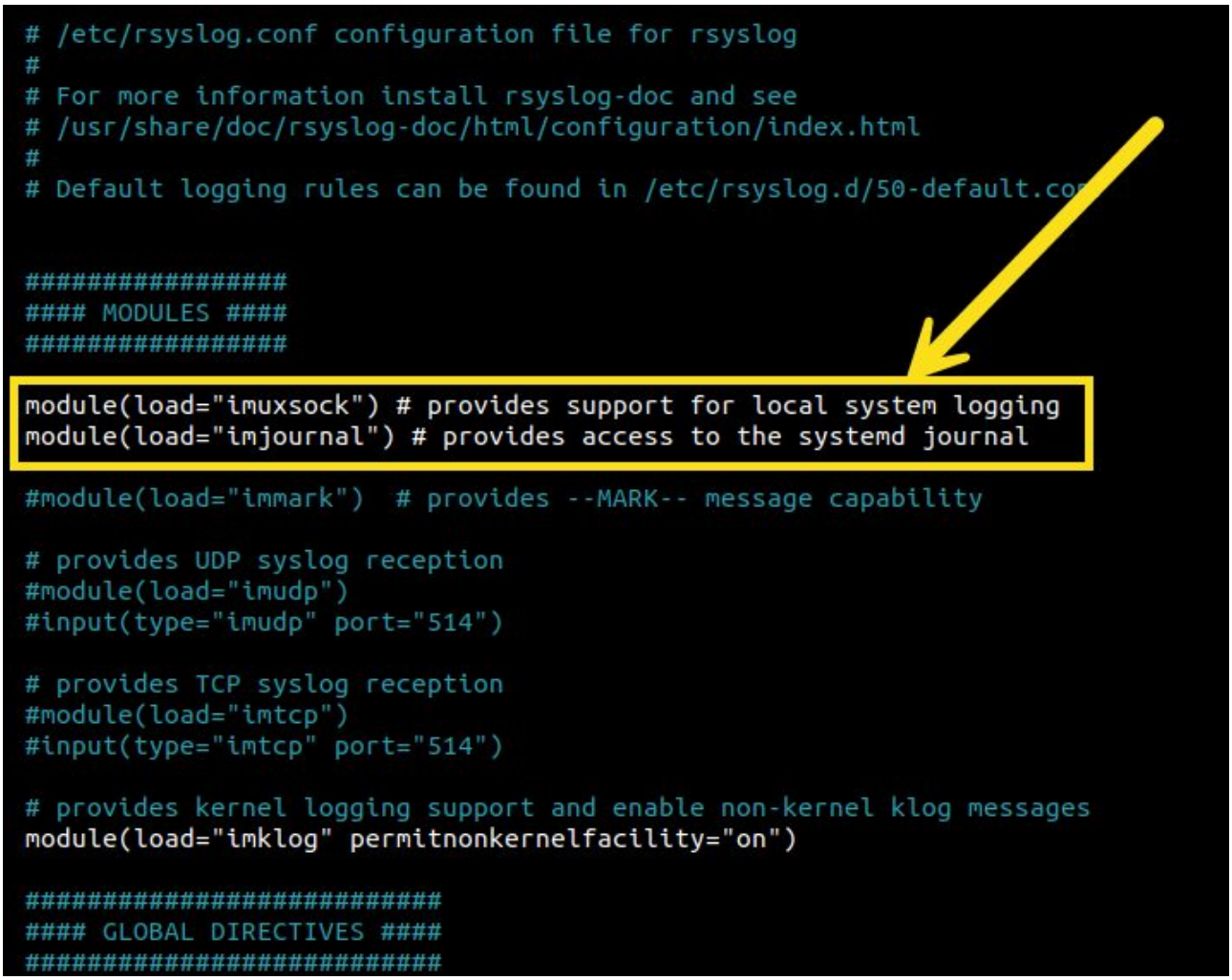
Step 2. Configure rsyslog to forward the event data

This involves modifying Ryslog.

Open the rsyslog configuration file, ensure that the imuxsoc and imjournal modules are loaded (uncommented) and then add a line at the end which forwards data to the collector.

First, edit the config file:

Ensure that the following lines are present - and not commented out. If they are missing, then please add them.



Checking that the correct modules are loaded.

Next, add this string to the end of the ryslog.conf file (replace 10.10.10.10 with the IP address of your centralised log storage device)

```
# Send traffic to remote collector
*.* @@10.10.10.10
```

Sending events to a remote collector

Finally, restart rsyslog

Step 3. Check things are working.

On the source system, you can verify this as follows

Check that rsyslog is working:

```
```

It should look something like this:

```
ops@asgard:~$ sudo systemctl status rsyslog
● rsyslog.service - System Logging Service
     Loaded: loaded (/lib/systemd/system/rsyslog.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2024-07-17 16:20:37 GMT; 2min 43s ago
TriggeredBy: ● syslog.socket
       Docs: man:rsyslogd(8)
             man:rsyslog.conf(5)
             https://www.rsyslog.com/doc/
   Main PID: 3138 (rsyslogd)
      Tasks: 5 (limit: 9378)
     Memory: 55.3M
        CPU: 9.121s
     CGroup: /system.slice/rsyslog.service
             └─3138 /usr/sbin/rsyslogd -n -iNONE

Jul 17 16:20:37 asgard systemd[1]: Starting System Logging Service...
Jul 17 16:20:37 asgard systemd[1]: Started System Logging Service.
Jul 17 16:20:37 asgard rsyslogd[3138]: imuxsock: Acquired UNIX socket '/run/systemd/journal/syslog' (fd 3) from systemd.  [v8.2112.0]
Jul 17 16:20:37 asgard rsyslogd[3138]: rsyslogd's groupid changed to 111
Jul 17 16:20:37 asgard rsyslogd[3138]: rsyslogd's userid changed to 104
Jul 17 16:20:37 asgard rsyslogd[3138]: [origin software="rsyslogd" swVersion="8.2112.0" x-pid="3138" x-info="https://www.rsyslog.com"] start
Jul 17 16:20:41 asgard rsyslogd[3138]: imjournal from <asgard:NetworkManager>: begin to drop messages due to rate-limiting
Jul 17 16:20:45 asgard rsyslogd[3138]: imjournal: journal files changed, reloading...  [v8.2112.0 try https://www.rsyslog.com/e/0 ]
```

Rsyslog is working

You can also check that traffic is flowing - although this will need you to generate event data. A quick and simple approach is to use tcpdump. (Make sure the IP address relates to your system)

```
```

```
ops@asgard:~$ sudo tcpdump -i any host 192.168.50.131 and port 514
tcpdump: data link type LINUX_SLL2
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
16:21:58.358230 ens33 Out IP asgard.47406 > 192.168.50.131.shell: Flags [P.], seq 805240781:805240905, ack 3963691495, win 502, options [nop,nop,TS val 3927169746 ecr 705
16:21:58.359070 ens33 In  IP 192.168.50.131.shell > asgard.47406: Flags [.], ack 124, win 1432, options [nop,nop,TS val 705966158 ecr 3927169746], length 0
16:21:58.359105 ens33 Out IP asgard.47406 > 192.168.50.131.shell: Flags [P.], seq 124:231, ack 1, win 502, options [nop,nop,TS val 3927169747 ecr 705966158], length 107
16:21:58.359687 ens33 In  IP 192.168.50.131.shell > asgard.47406: Flags [.], ack 231, win 1432, options [nop,nop,TS val 705966158 ecr 3927169747], length 0
16:21:58.378106 ens33 Out IP asgard.47406 > 192.168.50.131.shell: Flags [P.], seq 231:382, ack 1, win 502, options [nop,nop,TS val 3927169766 ecr 705966158], length 151
16:21:58.379618 ens33 In  IP 192.168.50.131.shell > asgard.47406: Flags [.], ack 382, win 1432, options [nop,nop,TS val 705966178 ecr 3927169766], length 0
16:21:58.379655 ens33 Out IP asgard.47406 > 192.168.50.131.shell: Flags [P.], seq 382:460, ack 1, win 502, options [nop,nop,TS val 3927169768 ecr 705966178], length 78
16:21:58.380269 ens33 In  IP 192.168.50.131.shell > asgard.47406: Flags [.], ack 460, win 1432, options [nop,nop,TS val 705966179 ecr 3927169768], length 0
16:21:58.380943 ens33 Out IP asgard.47406 > 192.168.50.131.shell: Flags [P.], seq 460:536, ack 1, win 502, options [nop,nop,TS val 3927169769 ecr 705966179], length 76
16:21:58.381617 ens33 In  IP 192.168.50.131.shell > asgard.47406: Flags [.], ack 536, win 1432, options [nop,nop,TS val 705966180 ecr 3927169769], length 0
16:21:58.405579 ens33 Out IP asgard.47406 > 192.168.50.131.shell: Flags [P.], seq 536:615, ack 1, win 502, options [nop,nop,TS val 3927169794 ecr 705966180], length 79
16:21:58.406197 ens33 In  IP 192.168.50.131.shell > asgard.47406: Flags [.], ack 615, win 1432, options [nop,nop,TS val 705966204 ecr 3927169794], length 0
16:21:58.441571 ens33 Out IP asgard.47406 > 192.168.50.131.shell: Flags [P.], seq 615:691, ack 1, win 502, options [nop,nop,TS val 3927169830 ecr 705966204], length 76
16:21:58.442273 ens33 In  IP 192.168.50.131.shell > asgard.47406: Flags [.], ack 691, win 1432, options [nop,nop,TS val 705966240 ecr 3927169830], length 0
16:21:58.442316 ens33 Out IP asgard.47406 > 192.168.50.131.shell: Flags [P.], seq 691:838, ack 1, win 502, options [nop,nop,TS val 3927169830 ecr 705966240], length 147
16:21:58.442851 ens33 In  IP 192.168.50.131.shell > asgard.47406: Flags [.], ack 838, win 1432, options [nop,nop,TS val 705966241 ecr 3927169830], length 0
16:21:58.444777 ens33 Out IP asgard.47406 > 192.168.50.131.shell: Flags [P.], seq 838:924, ack 1, win 502, options [nop,nop,TS val 3927169833 ecr 705966241], length 86
```

Tcpdump showing traffic on port 514

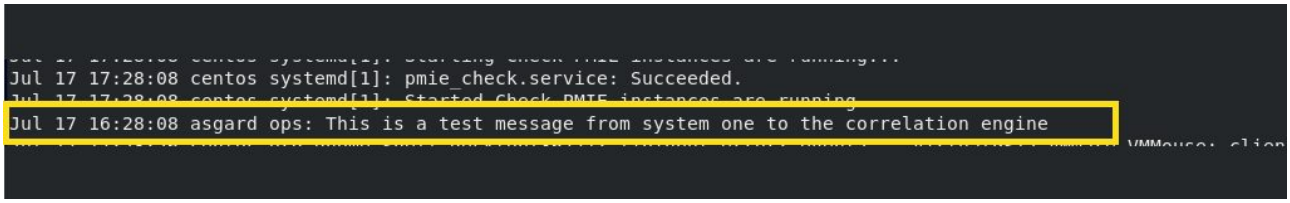
Finally, you can use the logger command to send a test message and check it is received on the correlation system.

```
```

```
ops@asgard:~$ logger "This is a test message from system one to the correlation engine"
ops@asgard:~$
```

Using logger on the source system

And then check it has been received on the destination system. How you do this depends on the destination system but examples include:



Output in /var/log/messages on the correlation system

You have now successfully configured the Linux journal to be logged to a remote log collector.

Forwarding the journal with Syslog-ng

Syslog-ng users can have effectively the same outcome, but there are some minor changes. The journal configuration remains the same but this time, modify the syslog-ng.conf file and add the remote destination data.

```
destination d_remote {
    tcp("10.10.10.10" port(514));
};

log {
    source(src);
    destination(d_remote);
};
```

Add remote collector details to syslog-ng configuration file

Then test the connection as mentioned previously.

What about other distros?

The examples here are from Ubuntu, so you might be curious about how to get it working on RHEL, SUSE etc.

The good news is that the syntax is pretty much identical. The only noticeable difference would be if you needed to install Rsyslog or Syslog-ng then you'd use dnf or yum (etc). The configuration options remain consistent across platforms.

Summary

The journal is very much the future of event data on Linux systems. By default, this is only written locally to a binary file, and this is less than ideal for incident response purposes. With a few simple configuration steps, you can send this data to a remote

collector, so even if you dont have a fully configured SIEM, you can have a central analytics point.

And yes, it is better if you have a SIEM in place.

About Halkyn Consulting

Halkyn Consulting Ltd is a UK-based security and risk management consultancy providing 24/7 incident response, digital forensics, threat hunting, virtual security leadership (vCISO) and security compliance support to organisations worldwide. We deliver clear, practical, vendor-independent advice based on real-world threats — no hype, no upsell.

More free security resources are available at www.halkynconsulting.co.uk/security-resources.

Web: www.halkynconsulting.co.uk | **Email:** info@halkynconsulting.co.uk | **Tel:** +44 1244 940858