Container and Linux Logging

Author: Stefan Alfeis, stefan.alfeis@quant-x-sec.com
Translation: Jeremias Bogomolec, jeremias.bogomolec@quant-x-sec.com

System- and application-logs are an essential part of running both servers and apps. On most Linux systems the log files can be found in the path "*/var/log/*". From this point on there will be differences, depending on which distribution is used.

Events and Activities

All **global activities**, for example system startup messages, are stored in the path "*/var/log/syslog*" by Debian-based systems like Ubuntu. Red Hat-based systems (e.g. RHEL or CentOS) use the path "*/var/log/messages*".

Events that **concern security**, e.g. logins, root-user actions and the like are stored under the path "*/var/log/auth.log*" on Ubuntu and Debian systems. On RHEL and CentOS the path is called "*/var/log/secure*".

For **Kernel Events** (*/var/log/kern.log*) and **reoccurring tasks**, so-called cron-Jobs, (*/var/log/cron*) there is no difference between the various distributions.

Syslog

So the logging on Linux is primarily conducted via the Standard Syslog. Syslog describes, how logs are created and transported on Linux. The Syslog-Service provides a destination for applications in the path "*/dev/log*", in which these applications can then write their logs. The Syslog-Service can either store its files locally or redirect them to a connected server. The format of Syslog-files is predefined by the Syslog-Protocol (RFC 5424). The protocol also defines how files are transferred through a network and which ports are to be used (Port 514 for simple texts and Port 6514 for encoded messages).

Systemd

Today, many Linux distributions use systemd, a **process and service manager**. It is immensely versatile and was therefore incorporated in Linux Distributions. Not only does it organise daemons(background processes and programs) and other processes, but also various resources like Mount Points, hardware and sockets. Furthermore, systemd does not load processes and services sequentially, but parallel to one another, greatly accelerating the offset time.

Systemd's core are the so-called unit files. A unit file describes its source and tells systemd how to activate that source. Unit files have the following standard: *<resource_name>.<unit_type>* (e.g. cron.service or syslog.socket). The path for unit files, which consist of plain text, is named "*/lib/systemd/system*".

The journal is another part of systemd. It is a central place for all logged messages of different components of a Linux system and is a binary file. This binary file cannot be opened via text editors. The journal is controlled by the dedicated daemon. The related path is called "*/var/log/journal*". To now extract the information from journald (the daemon for journal) there is journalctl. It can read the binary file and convert it to text.

Log-Management in connection with container systems

All of the previously described processes take place within a console. Since it can be very inconvenient to read and examine log files in such a console, we use Log-Management-Systems. These systems are capable of fetching and processing log files from their save location. There are many Log-Management-Systems on the market, so I won't go into any more detail on the different options here.

Container systems are based on Linux systems and can therefore also be monitored and controlled via Log-Management-Systems. Depending on the container distribution the agents of each Log-Management-System are situated differently. In **Docker** the log-files can be stored in nearly every way; host-based, in a dedicated Logging-Container or the agent is directly accommodated within the container.

From the ground up **Kubernetes** brings its own logging-architecture with it and automatically collects all container-logs on the host. There are, however, also a large number of Log-Management-Systems, that create a Log-Pod directly within a cluster of Kubernetes and obtain the log-data from there.

The logging functions similarly with **Podman**. Here, too, Log-Management-Systems can accommodate agents directly within a pod. If the pods are run rootless though, there can be problems, that need to be examined individually. REMARK @Stefan: Add description of potential problems.

Logging Structure According to ISO27001

- User recognition
- System activation
- Date and time of important events
- Device identity (physical and virtual!)
- Successful and unsuccessful login attempts
- Successful and unsuccessful access to important data
- Access to files (with file type)
- Successful and unsuccessful access to important processes
- Change of system configuration
- Use of privileges (admin/root)
- Use of services and applications

- Use of network addresses and protocols
- Alarms raised by access control systems
- Activation and deactivation of protection sytsems (e. g. malware protection)
- Digital transactions from user applications

Conclusion

Logging is an important part of running servers and applications and that circumstance will not be changed by the establishment and use of container systems. The container systems each bring various ways to read and forward log-data, for example to a SIEM (Security Information Event Monitoring) system. Depending on usage and demand, the administrator can adjust and refine the logging configuration.

Source: https://www.loggly.com/ultimate-guide/linux-logging-basics/