Centralized Logging With syslog-ng

Ryan Matteson
matty91@gmail.com
http://prefetch.net

Presentation Overview

- Tonight I am going to discuss centralized logging and how syslog-ng can be used to create a centralized logging infrastructure
- I am planning to split my presentation into two parts:
  - Part 1 will provide an overview of syslog-ng
  - Part 2 will show how to configure syslog-ng to act as a centralized logging server

What Is Centralized Logging?

- Centralized logging allows you to store your Linux, UNIX and Windows logs in a centralized repository
- Provides several benefits:
  - Single location to check for system errors (ever had a disk die that disrupted local logging?)
  - Security, especially when you need to put together timelines after a system compromise
  - Often required for security compliance

What Is syslog-ng?

- Syslog-ng is a flexible and robust open source syslog implementation
- Provides numerous features:
  - Logging via udp or tcp
  - Mutual authentication through digital certificates
  - Encryption of log traffic via TLS
  - Filters can be used to sort traffic based on host, facility, log level, message contents, etc.
  - Messages can be parsed and rewritten (this is especially useful for removing sensitive data from log messages)
  - Logs can be sent to a SQL database

How Does syslog-ng Work?

- Syslog-ng is configured through a single text file, which contains one or more sections that describe where to read log messages from, how to process them, and where to send them after processing
- Sections are broken down into:
  - Global options
  - Filter statements
  - Parser and rewrite statements
  - Traffic sources
  - Traffic destinations
  - Log statement

Syslog-ng Global Options

- Global options allow you to control the global behavior of syslog-ng
- Global options include:
  - Entries to resolve hosts through DNS
  - How many log entries to write(2) out at a time
  - Permissions to assign to files
  - Whether or not to preserve names when entries are forwarded through another syslog process

Global Options Example

- Global options are specified in an options block:

```
@version: 3.0

options {
        flush_lines(100);
        use_dns(no);
        owner(root);
        group(logs);
        perm(0640);
        dir_perm(0750);
        dir_owner(root);
        dir_group(logs);
        create_dirs(yes);
        stats_freq(3600);
};
```

Traffic Sources

- Syslog-ng uses traffic sources to define where syslog-ng should read log messages from
- Several types of sources exist:
  - ***internal*** – messages generated by syslog-ng
  - ***file*** – contents of a file
  - ***fifo*** – read from a named pipe
  - ***program*** – execute program to get data
  - ***tcp / udp*** - listen on a tcp or udp socket
  - ***unix-dgram / unix-stream*** – listen for messages on a UNIX domain socket

Example Sources

- Sources are created by adding a source statement along with one or more configuration directives to a source block:

```
source local {
        file ("/proc/kmsg" log_prefix("kernel: "));
        unix-stream ("/dev/log");
        internal();
};

source network {
        udp(ip(0.0.0.0) port(514));
};
```

Log Destinations

- Syslog-ng uses destinations to specify where log messages should be written or forwarded to
- Several types of destinations exist:
  - ***file*** – write message to a file
  - ***fifo*** – write the message to a named pipe
  - ***program*** – Launches a program
  - ***sql*** – write the message to a SQL database
  - ***tcp / udp*** – forward the message to a remote server:port
  - ***unix-dgram / unix-stream*** – send the message to a UNIX domain socket
  - ***usertty*** – Send the message to a user's tty
- Several macros are available to allow flexible naming:
  - ***$HOST*** contains the hostname
  - ***$SOURCEIP*** contains the SRC IP of the client who sent the message
  - ***$MONTH, $DAY, $YEAR*** contain the date the message was created
  - The syslog-ng manual contains the full list

Example Destination

- Destinations can be created by defining a destination {} with a log destination, and adding optional destination options:

```
destination d_unix_oom_msgs {
file("/log/unix/kernoom.$HOST.$YEAR.$MONTH.$DAY"
        owner(matty) group(matty) perm(0600)
        dir_owner(matty) dir_group(matty)
        dir_perm(0700));
 };
```

Filters

- Filters allow you to route incoming messages to destinations based on or more types of criteria
- Criteria can be matched using one or more filter functions:
  - ***facility*** – matches by the facility name
  - ***level*** – matches by the log level
  - ***match*** – matches against a string in message and headers
  - ***message*** – matches a string against the message
  - ***host*** – match against the IP or hostname
  - ***netmask*** – match against an IP/netmask
  - Additional functions are listed in the syslog-ng manual
- Complex filters can be created using POSIX and PCRE regular expressions (*, ^, [], etc.), as well as through the use of one or more logical operators (or, and, not)

Example Filter

- The following filter looks for messages sent from 192.168.1.1 and 192.168.1.2 that are part of the kern facility and contain the string “Out of Memory”:

```
filter f_kern_oom {
        ((host("192.168.1.1") or
          host("192.168.1.2")) and
          facility(kern) and
          level(debug…emerg) and
          message("Out of Memory"));
};
```

Log Statements

- Log statements allow you to combine filters, sources and destinations to control where messages are sent:

```
log { source(network);
        filter(f_kern_oom);
        destination(d_unix_oom_msgs);
        flags(final);
};
```

Monitoring syslog-ng Usage

- Syslog-ng gathers statistics for each log destination, and will write them out periodically (the interval is controlled by the stats(time interval) directive) to the system logs:

  *Oct 3 14:40:07 local@foo syslog-ng[1234]: *
  *Log statistics; processed='center(queued)=24169972', *
  *processed='center(received)=24170053', *
  *processed='destination(linux)=1235', *
  *processed='source(local)=253', *
  processed='source(network)=24169800'

Debugging syslog-ng Issues

- If a filter isn't working the way you expect it to, you can run syslog-ng with the "-d" (debug) and "-e" (log to stdout) options to observe rule processing:

  $ ***syslog-ng –e –d > /var/tmp/syslog.out 2>&1***

  $ ***less /var/tmp/syslog.out***

Conclusion

- Syslog-ng offers a flexible and easy way to configure centralized logging solution
- When combined with tools such as logwatch and swatch, you will be able to understand exactly what is going on with your servers, and will have one place to look when things go wrong

References

- Syslog-ng website:
  http://www.balabit.com/network-security/syslog-ng/
- Syslog-ng manual:
  http://www.balabit.com/dl/guides/syslog-ng-v3.0-guide-admin-en.pdf

Questions?