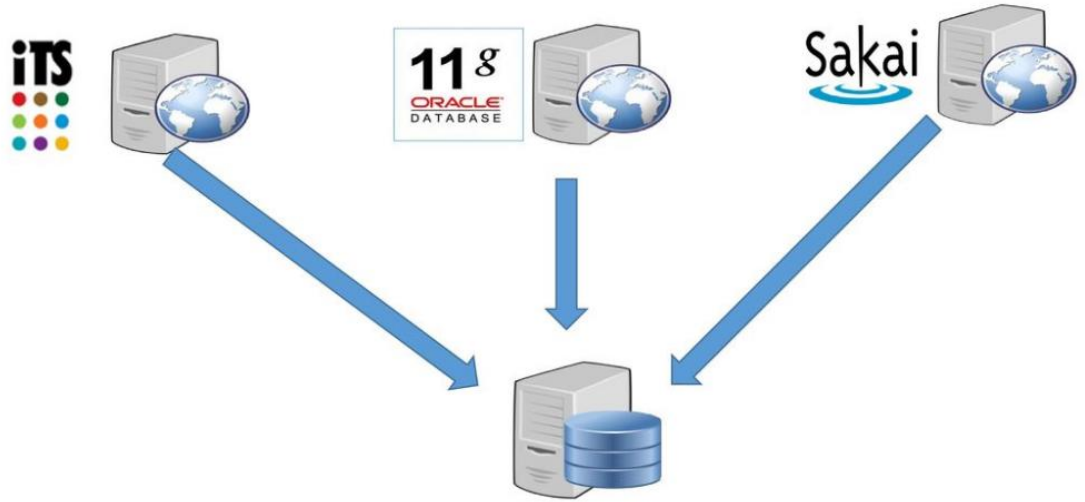
Centralizing logs with Rsyslog

Log Management

- Keep your logs in a secure place where they can be easily inspected
- Watch your log files
- They contain important information:
  - Lots of things happen and someone needs to review them
  - It's not practical to do this manually.

Centralizing Logs

- Centralize and consolidate log files
- Real time sending of all log messages from your servers to a single node – a log server.
- Save a copy of the logs locally, but, also, save them to a central log server.



What is Rsyslog ?

- Rsyslog is the **r**ocket-fast **sys**tem for **log** processing.
- Rsyslog is an open-source software utility 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 configuration options and adds features such as using TCP for transport.
- Rsyslog is a replacement for regular syslog.
- It adds a bunch of features:
  - Better security controls
  - More filtering options/syntax
  - More reliable transport mechanisms
  - Writing to databases

Rsyslog Protocol

- Rsyslog uses the standard BSD syslog protocol, specified in RFC 3164 (RFC 5424).
- Rsyslog supports many of these extensions. The format of relayed messages can be customized.
- The most important extensions of the original protocol supported by rsyslog are:
  - ISO 8601 timestamp with millisecond granularity and time zone information
  - Reliable transport using TCP
  - Logging directly into various database engines.
  - Support for RELP - Reliable Event Logging Protocol

Log rotation in Rsyslog

- Maximum file size condition can be used here.
- This can be configured in rsyslog.conf
- Lets assume you do not want to spend more than 100 MB hard disc space for you logs.
- With rsyslog you can configure Output Channels to specify maximum file size.
- If the maximum file size is reached it will perform an action: moving the file, removing the file, etc.

Cont'd

- Example

```
# start log rotation via outchannel
# outchannel definition
$outchannel log_rotation,/var/log/log_rotation.log, 52428800,/home/me/./log_rotation_script
#  activate the channel and log everything to it
*.* :omfile:$log_rotation
# end log rotation via outchannel
```

- This will instruct rsyslog to log everything to the destination file '/var/log/log_rotation.log' until the give file size of 50 MB is reached.
- If the max file size is reached it will perform an action.
- In this case it executes the script /home/me/log_rotation_script which contains a single command: move the original log to a kind of backup log file.

Log filtering in Rsyslog

- Rsyslog offers different types "filter conditions":

1. Facility and severity based selectors

The selector field itself again consists of two parts, a facility and a priority, separated by a period (".")

Facility: syslog messages are widely classified on the basis of the sources generating them.

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.

Severity: the source or facility generating the syslog message, also specifies the severity of the message.

The **severity** 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).

Cont'd

An asterisk ("*") stands for all facilities or all severities, depending on where it is used (before or after the period).

The keyword none stands for no priority of the given facility.

You can specify multiple facilities with the same priority pattern in one statement using the comma (",") operator.

Example:

- To select all mail syslog messages with priority crit and higher, use this form:

```
mail.crit
```

Cont'd

2. property-based filters

They allow to filter on any property, like hostname, and msg.

A property-based filter must start with a colon.

The colon must be followed by the property name, a comma, the name of the compare operation to carry out, another comma and then the value to compare against.

This value must be quoted.

Property names and compare operations are case-sensitive,

Example:

```
:msg, contains, "ID-4711"
```

This filter will match when the message contains the string "ID-4711"

Cont'd

3. expression-based filters

Expression based filters allow filtering on arbitrary complex expressions, which can include boolean, arithmetic and string operations.

Expression filters will evolve into a full configuration scripting language.

Expression based filters are indicated by the keyword "if" in column 1 of a new line. They have this format:

```
if expr then action-part-of-selector-line
```

"if" and "then" are fixed keywords that must be present. "expr" is a (potentially quite complex) expression.

So the expression documentation for details.

"action-part-of-selector-line" is an action,

just as you know it (e.g. "/var/log/logfile" to write to that file).

Sending specific logs to Rsyslog server

- Let's say you wanted to send **cron** logs to 10.1.1.15 and **FTP** logs to 10.1.1.20
- Then the Rsyslog client configuration starts like the following:

```
cron.* @10.1.1.15

ftp.* @10.1.1.20
```

Need Help ?

- Support channels
  - Stack Overflow
  - GitHub

Thank You