Collecting log messages from UDP sources

June 19, 2018

Table of Contents

Central logging is based on TCP connections in most cases (or even encrypted TCP connections) as it provides several reliability features that are simply not available using UDP. Still, there are certain situations, when you have to use UDP. One use case is when the company standard syslog configuration for servers contains only a single, common UDP destination, and it cannot be altered because of company IT policies. The other, more common use case is, that network devices, for example routers, switches or firewalls send their logs using UDP. Most of the time the TCP implementation of syslog is completely missing from these devices. In some cases it is there, but badly broken and therefore avoided by the users.

All questions, comments or inquiries should be directed to <info@balabit.com> or by post to the following address: One Identity LLC 1117 Budapest, Alíz Str. 2 Phone: +36 1 398 6700 Fax: +36 1 208 0875 Web: *https://www.balabit.com/*

1. Problems

When compared to TCP, UDP is more lightweight, and it consumes less computing resources. However, this is probably the only positive feature of UDP when discussing log message transfer. On the sender side UDP sends packets with a “fire and forget” method, meaning that it does not ensure that the packet was received, there is no error handling, acknowledgement, retransmission or timeout and therefore it is prone to losing messages.

There are several points during the transmission where messages can be lost when using UDP:

- When there are multiple hops, a burst of log messages can be lost.
- Kernel network card driver level: virtualization might cause problems, also certain drivers and cards are unable to keep up with high traffic and therefore log messages can be lost.
- Kernel buffer size: when syslog-ng cannot read fast enough from the kernel, buffers are filled and log messages can be lost.
- In syslog-ng when the destination side is not fast enough and buffers are filled. For example when the disk I/O is held by some other application, therefore the messages cannot get written by syslog-ng. Even on a dedicated server, just browsing the stored logs can be a reason for that.

The effects of these problems can be reduced, but keep in mind, that UDP is not a reliable protocol.

2. Resolving UDP message loss problems

Even if UDP is not a reliable protocol, and message loss can not be completely avoided, there are many ways to improve the situation. If you only have a few incoming UDP messages, you might not need any or all of these safe guards. However, as soon as you expect higher UDP traffic or burst of UDP packages, it is definitely worth implementing them.

Collect messages close to the source

UDP packages easily get lost if they have to travel through switches and routers. This kind of loss is difficult to detect, unless you can count the number of messages both on the sending and receiving side. To avoid message loss, install a syslog-ng relay collecting UDP messages as close as possible to the log source, ideally on the same switch or at least on the same subnetwork. Forward the logs to your central log server using TCP or RLTP protocols.

Use the right network card and network driver

Some network cards handle high load better than others. In high traffic environments it is worth using server class network cards that can off-load many of the tasks. The same hardware can sometimes be used with different drivers, so select the driver that suits your needs best.

To generate syslog traffic for stress testing your environment, use `loggen` (a tool bundled with syslog-ng). To check for dropped packages at hardware level, use `ifconfig` or `ethtool`.

Use physical machines instead of virtual ones

Full hardware virtualization, for example VMware can degrade performance, especially during hardware access like high speed networking. In some situations, using the right network driver inside the virtual machine can resolve this problem. On a loaded host machine even the right network driver will not solve these issues, therefore it is better to move log collection to a physical machine.

Use large buffers in kernel

If syslog-ng cannot read the messages fast enough from the UDP socket, the kernel receive buffers will start to fill and after the configured limit has been reached, the kernel will start discarding messages. In this case, it is necessary to adjust the buffer size accordingly. To raise the size of the kernel receive buffers, use the `sysctl` command to tune the *`net.core.rmem_max`* parameter. Next, raise the size of the *`so-rcvbuf`* option of the syslog-ng source definition as well, so that syslog-ng is capable of utilizing the larger kernel receive buffers. In a high traffic environment as high as 256MB might be necessary:

```
sysctl -w net.core.rmem_max=268435456
```

Enter the value in bytes. In the example above, 256*1024*1024=`268435456` bytes. As a rule of thumb, this buffer size should be enough to accommodate incoming peak message rate for at least one second.

Warning
Any issue causing the loss of buffer content can result in higher message loss because of the higher buffer size. To minimize risk, it is important to determine the required buffer size instead of using a buffer that is higher than what is absolutely necessary.

To monitor packet loss, use the following command:

```
netstat -su | grep "receive errors"
```

Tuning syslog-ng for UDP

There are several configuration possibilities in syslog-ng that are related to using UDP protocol. To be able to handle message bursts, increase the value of the *log_fifo_size()* option. To match the value configured for `net.core.rmem_max` in the previous step, increase the value of the *so-rcvbuf()* accordingly.

Flow control in syslog-ng slows down message reception if the destination slowed down for some reason. This works fine when messages are coming from TCP sources, as the sender side will notice slower reception and send messages at a lower rate. UDP, as a stateless protocol cannot handle this situation, the sender side will never notice that the receiving end slowed down and this will result in immediate message loss. Therefore in case of UDP sources, do not enable flow control for the destination.

When you use a syslog-ng Store Box as the central logserver, flow-control is always applied to the log paths where messages are written into logspaces in order to avoid message loss. Therefore, log processing may slow down if logs are massively searched by dozens of users (this is similar to the example described in the “Problems” section above). Contrary to the highly configurable syslog-ng PE, you do not have the ability to fine-tune the low level OS level buffers underneath the syslog-ng Store Box log management application. Therefore, you may turn the UDP-based log collection into TCP-based already on a lower message rate.

The UDP source driver in syslog-ng is not multi-threaded. A single UDP source runs only in a single thread on a single CPU core. If you have a high message rate and multiple CPU cores, define multiple UDP sources in your syslog-ng configuration. This way load is distributed among multiple CPU cores, ensuring that CPU does not become a bottleneck. Still in this case, if the messages received from several UDP sources are written into the very same file, the destination side can become a bottleneck.

The consequence is that UDP-based collection always comes with a risk of message loss. The possibility of losing messages is different in each environment, therefore the situation might be examined in all cases in order to select the right safe guards that are sufficient to eliminate the risk of losing messages.