syslog-ng Premium Edition 6 LTS

Recovering log messages from corrupted disk queue files

Legend

WARNING: A WARNING icon highlights a potential risk of bodily injury or property damage, for which industry-standard safety precautions are advised. This icon is often associated with electrical hazards related to hardware.

CAUTION: A CAUTION icon indicates potential damage to hardware or loss of data if instructions are not followed.

syslog-ng PE Recovering log messages from corrupted disk queue files
Updated - 06 October 2023, 11:55
Version - 6 LTS

Contents

Corrupt disk queues **3**

Recovering corrupt disk queue files **4**

About us **10**

Contacting us 10

Technical support resources 10

Corrupt disk queues

Introduction

The syslog-ng Premium Edition application is a flexible and highly scalable system logging application that is ideal for creating centralized and trusted logging solutions.

Typically, syslog-ng Premium Edition is used to manage log messages and implement centralized logging, where the aim is to collect the log messages of several devices on a single, central log server. The different devices (called syslog-ng Premium Edition clients) all run syslog-ng Premium Edition and collect the log messages from the various applications, files, and other *sources*. The clients send all important log messages to the remote syslog-ng Premium Edition server, which sorts and stores them.

There is a possibility that the queue files become corrupt, therefore syslog-ng Premium Edition cannot read them and proceed. While syslog-ng Premium Edition and its tools cannot read from the corrupt queue file, there is a tool and a method to recover the corrupt log files.

CAUTION:

Due to the nature of the disk buffer implementation, this tool cannot guarantee that you can recover all logs from a queue file. In addition, when resending the logs, log loss or log duplication might occur.

WARNING:

The recovery tool only works on 64-bit Linux operating systems.

NOTE:

At the moment, this method only works with syslog-ng Premium Edition 6 LTS, as 6 LTS is the only version that contains the persist-tool necessary for resending the logs.

Recovering corrupt disk queue files

To recover corrupt disk queue files, complete the following steps.

1. **Check the corrupt queue files on the affected machine or in the core shell of SSB**.

   The default directory is /opt/syslog-ng/var but the files may be in a different folder (based on how you start syslog-ng or what you specified in the syslog-ng configuration).

   Checking corrupt queue files

   ```
   cd /opt/syslog-ng/var
   ls
   ```

   Sample output for the previous command

   ```
   (core/master/coreanalysis)root@ssb1:/opt/syslog-ng/var# ls
   run  syslog-ng-00000.qf  syslog-ng-00000.qf.corrupted  syslog-
   ng.persist  syslog-ng.static.persist
   ```

   If there are files with the .corrupted suffix in the directory you checked, they are corrupted queue files.

2. **Check the destination – queue file mapping**.

   Find the destination and disk queue mapping of the affected queue(s) in the persist file:

   Check destination-queue file mapping

   ```
   /opt/syslog-ng/bin/persist-tool dump /opt/syslog-ng/var/syslog-
   ng.persist
   ```

   Alternatively, you can use this one-liner snippet to display the affected mappings from the folder where the queue file resides:

Display affected mappings:

```
for i in $(ls -1 *.corrupted);do /opt/syslog-ng/bin/persist-tool dump
/opt/syslog-ng/var/syslog-ng.persist|grep ${i%.*};done
```

NOTE:

Keep in mind that the snippet above only works when there are no consecutive .corrupted filename suffixes.

Using the snippet above will result in an output like this:

Output

```
(core/master/coreanalysis)root@ssb1:/opt/syslog-ng/var# for i in $(ls
-1 *.corrupted);do /opt/syslog-ng/bin/persist-tool dump
                                                /opt/syslog-ng/var/syslog-ng.persist|gr
                                          afsocket_dd_qfile(stream,10.21.29.114:601) = {
"\/opt\/syslog-ng\/var\/syslog-ng-00000.qf" }
```

3. **Take a note of the mappings**.

   We will need the name of the destination and the queue file mapping to find the log sending format and also when re-sending the logs, so make sure you write them into a text file or on paper.

4. **Find the destination details**.

   Find the destination that sends messages to the address found in the mapping above on the web UI of SSB or in the configuration file of syslog-ng and write down the message template (if any exists).

5. **Move the queue files**.

   Copy the corrupted queue files to another machine or VM where you can recover their contents.

   NOTE:

   We strongly recommend you do this because the queue recovery process causes high IO load.

6. **The two following methods are possible to proceed**:

a. **To recover the logs entirely from start**

Use the dqread tool to recover the logs and preferably use the *screen* terminal application or anything else that does not stop when the user quits or the SSH connection is interrupted.

Navigate to the folder that contains the dqread tool and run it like this:

Run the dqread tool

```
LD_LIBRARY_PATH=$PWD ./syslog-ng_dqread -c 0 -g -t '[TEMPLATE_
COMES_HERE]' ../syslog-ng-00000.qf.corrupted 1>../syslog-ng-
00000.qf.recovered
```

This will print the status to the terminal and the recovered logs to a file.

If you are trying to recover a reliable disk queue that has the .rqf extension, add the -T r arguments.

If you made no changes to the default parameters, you can use the template below. Otherwise, use the same message template as the one you found on the SSB web UI or in the syslog-ng configuration.

If you used the legacy BSD-syslog template, the legacy template is the default in the tool, so you do not have to use a different template.

If you used IETF-syslog protocol, you can use this template:

IETF-syslog protocol template

```
<$PRI>1 $ISODATE $HOST $PROGRAM $PID - $SDATA ${MSG}\n
```

b. **To resume recovery if you get stuck or if SSB exited unexpectedly**.

If the recovery exited before it could entirely recover the corrupt queue file, you can restart and recover the logs into an other file.

If you used the -g switch, the current position is printed when the tool stops and you can use that byte as the starting position for the next run.

Resume recovery

```
LD_LIBRARY_PATH=$PWD ./syslog-ng_dqread -c 0 -g -p [START_BYTE_
POSITION] -t
                                                    '[TEMPLATE_COMES_HERE]' ..
1>../syslog-ng-00000.qf.recovered.2
```

7. **Wait for the file recovery process to finish**.

   NOTE:

   Depending on the size of the corrupted queue file, the recovery process can take hours or even days.

8. **Create a new syslog-ng configuration for resending the files**.

   Use syslog-ng to resend the files to the intended original destination. The config snippet below is necessary to rewrite the logs into a format that syslog-ng can send to the original destination. Put this snippet into a separate configuration file so you can start it independently from any syslog-ng instance already running.

   CAUTION:

   The destination has to use the template ("${MSG}\n") directive because the no-parse directive is used by syslog-ng in the source.

Config template to rewrite logs to syslog-ng compatible format

```
@version: 6.0
                                                    @include "scl.conf"

                                                    options {
                                                    stats_freq(0);
                                                    time_reopen(10);
                                                    chain_hostnames(off);
                                                    use_dns(no);
                                                    use_fqdn(no);
                                                    keep_hostname(yes);
                                                    keep_timestamp(yes);
                                                    };

                                                    source s_recovered_file { file("[/PATH/
```

```
flags(no-parse) ); };

                                                    # sample destination, the template dire
                                                    # use the same destination as found it
configuration
                                                    # and the message rate is throttled to
                                                    destination d_sampledest { syslog("10.2
throttle(100) template("${MSG}\n") ); };

                                                    # removes trailing garbage characters
                                                    rewrite r_removetrailing {
                                                    subst('\\x0a',
                                                    "",
                                                    value("MESSAGE")
                                                    flags("global" "substring")
                                                    );

                                                    subst("'$",
                                                    "",
                                                    value("MESSAGE")
                                                    type("pcre")
                                                    flags("global")
                                                    );
                                                    };

                                                    # formats any recovered Windows newline
proper one
                                                    rewrite r_newline {
                                                    subst('\\x0d\\x0a',
                                                    "\n",
                                                    value("MESSAGE")
                                                    flags("global" "substring")
                                                    );
                                                    };

                                                    # filter to use only the lines which co
                                                    filter f_startlines {
                                                    match("^msg;" value("MESSAGE") type("pc
                                                    };

                                                    # get rid of any unnecessary recovered
                                                    rewrite r_recoveredlog {
                                                    subst("^msg; pos='[0-9].*', len='[0-9].
```

```
                                        '',
                                        value("MESSAGE")
                                        type("pcre")
                                        flags("global")
                                        );
                                        };

                                        log {
                                        source(s_recovered_file);
                                        filter(f_startlines);
                                        rewrite(r_recoveredlog);
                                        rewrite(r_newline);
                                        rewrite(r_removetrailing);
                                        destination(d_sampledest);
                                };
```

9. **Run syslog-ng to resend the logs**.

   syslog-ng should be run with the following switches:

   ### syslog-ng --no-caps -Fev -f /path/to/config/file --persist-file=/path/to/new/persist/file --control=/path/to/new/control/socket

   This ensures that a separate instance is started and the persist file will hold the resending state, so it can be resumed.

Example

```
### /opt/syslog-ng/sbin/syslog-ng --no-caps -Fev -f
/var/tmp/resend.conf --persist-file=/var/tmp/resend.persist --
control=/var/tmp/syslog.ctl
```

About us

One Identity solutions eliminate the complexities and time-consuming processes often required to govern identities, manage privileged accounts and control access. Our solutions enhance business agility while addressing your IAM challenges with on-premises, cloud and hybrid environments.

Contacting us

For sales and other inquiries, such as licensing, support, and renewals, visit https://www.oneidentity.com/company/contact-us.aspx.

Technical support resources

Technical support is available to One Identity customers with a valid maintenance contract and customers who have trial versions. You can access the Support Portal at https://support.oneidentity.com/.

The Support Portal provides self-help tools you can use to solve problems quickly and independently, 24 hours a day, 365 days a year. The Support Portal enables you to:

- Submit and manage a Service Request
- View Knowledge Base articles
- Sign up for product notifications
- Download software and technical documentation
- View how-to videos at www.YouTube.com/OneIdentity
- Engage in community discussions
- Chat with support engineers online
- View services to assist you with your product