Potential system logging issues which may arise with Snort 2.9.x and rsyslog.

Last Revised on February 6 2013

The document below uses the following color codes for items/steps the user should be aware of:

Blue - informational messages and comments
Orange – These are commands that the user types at the shell prompt
Red – **Read carefully before proceeding**.

On systems where **rsyslog** is the default system logging utility (as opposed to traditional **syslog** or **sysklogd**), **there may be an issue** with **snort 2.9.x** when it is **logging** to **/var/log/messages** (the file where most Linux and Unix based systems log system messages).

On the following platforms (32 or 64 bit), rsyslog is the default system logging utility:

CentOS 6.x
Debian 5.0 or greater
Fedora 13 or greater
OpenSuSE 11.x/12.x
Ubuntu 10.0 or greater

The BSD based systems listed below use traditional syslogd for system logging:

FreeBSD 8.x/9.0
NetBSD 5.x/6.0
OpenBSD 5.x

To see which system logging utility is in use, issue the following command at the shell prompt:

```
ps aux | grep -i "syslog" <enter>
```

The following output was returned from my CentOS 6.3 system:

```
root     13323  0.0  0.1 183544  1420 ?        Sl   18:49   0:00 /sbin/rsyslogd -i
/var/run/syslogd.pid -c 5
```

If the output returns 'rsyslogd', rsyslog is the logging utility in use, otherwise it is 'syslogd' or 'sysklogd'. If rsyslog is being used, be aware that the default settings for rsyslog, any process ID (PID) which generates more than 200 messages in a 5 second interval will be rate-limited (discarded) to prevent /var/log/messages from being overrun by a potentially out of control process.

In reviewing /var/log/messages when snort 2.9.x is started from /etc/init.d or /etc/rc.d on systems where 'rsyslog' is running, you may see messages like this in /var/log/messages:

```
Feb  5 13:07:52 plugh snort[12105]:       Non-RFC Compliant Characters: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07
Feb  5 13:07:52 plugh snort[12105]:       Whitespace Characters: 0x09 0x0b 0x0c 0x0d
Feb  5 13:07:52 plugh snort[12105]: rpc_decode arguments:
Feb  5 13:07:52 plugh snort[12105]:     Ports to decode RPC on: 111 32770 32771 32772 32773 32774 32775 32776 32777 32778 32779
Feb  5 13:07:52 plugh snort[12105]:     alert_fragments: INACTIVE
Feb  5 13:07:52 plugh snort[12105]:     alert_large_fragments: INACTIVE
Feb  5 13:07:52 plugh snort[12105]:     alert_incomplete: INACTIVE
Feb  5 13:07:52 plugh snort[12105]:     alert_multiple_requests: INACTIVE
Feb  5 13:07:52 plugh snort[12105]: FTPTelnet Config:
Feb  5 13:07:52 plugh snort[12105]:     GLOBAL CONFIG
Feb  5 13:07:52 plugh rsyslogd-2177: imuxsock begins to drop messages from pid 12105 due to rate-limiting     ← LOOK HERE
Feb  5 13:07:57 plugh snort[12106]: Daemon initialized, signaled parent pid: 12105
Feb  5 13:07:57 plugh snort[12106]: Reload thread starting...
Feb  5 13:07:57 plugh snort[12106]: Reload thread started, thread 0x7f4039a37700 (12106)
Feb  5 13:07:57 plugh kernel: device eth0 entered promiscuous mode
Feb  5 13:07:57 plugh snort[12106]: Decoding Ethernet
Feb  5 13:07:57 plugh snort[12106]: Checking PID path...
Feb  5 13:07:57 plugh snort[12106]: PID path stat checked out ok, PID path set to /var/run/
Feb  5 13:07:57 plugh snort[12106]: Writing PID "12106" to file "/var/run//snort_eth0.pid"
Feb  5 13:07:57 plugh snort[12106]: Set gid to 40000
Feb  5 13:07:57 plugh snort[12106]: Set uid to 40000
Feb  5 13:07:57 plugh snort[12106]:
Feb  5 13:07:57 plugh snort[12106]:         --== Initialization Complete ==--
Feb  5 13:07:57 plugh snort[12106]: Commencing packet processing (pid=12106)
```

Note: PID 12105 is the ID assigned to snort when I started it on my CentOS 6.3 system.

This is an indication that PID 12105 is sending more than 200 messages in a 5 second interval to /var/log/messages. Also, if you notice the timestamps on the **imuxsock** line and the **Daemon initialized** line, 5 seconds have passed (in which **all messages** that snort 2.9.x sent to /var/log/messages were **discarded** by rsyslogd.

During initialization or normal operation of SNORT 2.9.x, this limit may be exceeded and result in possibly valuable information being lost to the system or snort administrator on this computer.

It may also be exceeded on a system which has a large amount of traffic that snort is processing or where multiple instances of snort are running on the same system (e.g. - where snort is monitoring multiple network cards), or where the system has an exceptionally heavy workload (e.g. - a load average constantly in excess of 10.00).

In above cases, there are two possible workarounds, the first would be to increase the message and time intervals for rsyslog, or the second would be to simply turn off rate-limiting for rsyslog.

If you wish to use the first solution, do the following:

Locate the file(s) rsyslog.conf and/or rsyslog.early.conf (usually found in /etc)

IMPORTANT: remember to make a backup copy of any rsyslog.* files before editing

vi rsyslog.conf and/or rsyslog.early.conf and add the following lines:

```
$SystemLogRateLimitInterval 10
$SystemLogRateLimitBurst 500
```

after any ModLoad commands in rsyslog.conf or rsyslog.early.conf

This will tell rsyslog that it will start rate-limiting (discarding messages) when more than 500 messages from a single PID are received within a 10 second interval (these numbers are not absolutes, they can be tailored to any given system, btw). Additionally, after reviewing a general form of this article that was posted on the SANS diary, a reader also suggested the following information:

The proper way to reduce the risk of discarding messages would be to INCREASE the $SystemLogRateLimitBurst value and/or DECREASE the $SystemLogRateLimitInterval value. This would guarantee that the message rate that triggers the limit has increased.

The second solution is to simply turn off rate-limiting for rsyslog, and to do this, add the following line to rsyslog.early.conf and/or rsyslog.conf using your favorite editor (I'm a vi/vim/gvim hound):

```
$SystemLogRateLimitInterval 0
```

after any ModLoad commands in rsyslog.conf and/or rsyslog.early.conf

This will disable any rate-limiting in effect for the rsyslog process running on this system. Note that by doing this, an out of control process ID on your system can fill up /var/log/messages with a lot of useless messages (which is why rate-limiting is enabled by default in rsyslog).

After doing one of the two solutions above, stop and start the rsyslog daemon in /etc/rc.d or /etc/init.d by **./rsyslog stop|start** or by using **systemctl stop|start rsyslog**

After doing this, be sure to check that rsyslog is actually running by using the following command:

```
ps aux | grep -i "syslog" <enter>
```

If it's running, you should see some output like:

```
root      1740  0.0  0.1  56364  1304 ?        Sl   15:57   0:00 /sbin/rsyslogd -c 5 -f
/etc/rsyslog.conf
```

on your terminal or console.

FYI, I know the list of operating systems above is incomplete, but this document is based on distributions that I actively build and test snort on.

If you need more information about rsyslog, you can visit the following URL:

http://www.rsyslog.com/doc

If you have any questions, comments, suggestions, or additions please email me at:

wp02855@gmail.com (wp02855 at gmail dot com)

Bill Parker