Installing Snort 3 Alpha 4 on openSUSE Leap 42.3 64 bits

Boris A. Gómez
Universidad Tecnológica de Panamá
November 2017

About This Guide
This guide has been tested on openSUSE Leap 42.3 64 bits, using DAQ 2.2.2 for Snort 3 and Snort 3 Alpha 4.

Snort 3 is still in the early testing phase (Alpha version) so it should not be installed on production systems.

Our Snort 3 was installed in a virtual machine:
   Virtual Machine Manager: VirtualBox 5.1.30
   HOST operating system:     Windows 10, 64 bits
   GUEST operating system:   openSUSE Leap 42.3

This guide is based on “Installing Snort++ in Ubuntu (Version 3.0 Alpha 4 build 240)” by Noah Dietrich.

VirtualBox - Network Configuration
Configure the network section of the guest machine to the bridge mode:

Guest Machine
Run the guest machine and set its network interface card to a static IP, for example 192.168.99.10.

then check settings:

```
sudo ifconfig
```

et1  Link encap:Ethernet  HWaddr 08:00:27:04:F6:F0
Verify that you can access internet by accessing any web page, for example: https://snort.org

**Required Packages**
Use YAST to install the following packages:

<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmake 3.5.2</td>
<td>for managing the build process</td>
</tr>
<tr>
<td>gcc-c++ 4.8, libgcc_s1 7.1.1, libgcc_s1-32bit 7.1.1</td>
<td>the system GNU C++ Compiler and libraries</td>
</tr>
<tr>
<td>libdnet1 1.12, libdnet-devel 1.12 (faltaban)</td>
<td>for network utility functions</td>
</tr>
<tr>
<td>hwloc 1.10.1, hwloc-devel 1.10.1</td>
<td>provides portable abstraction across platforms</td>
</tr>
<tr>
<td>lua51-luajit 2.0.4, lua51-luajit-devel 2.0.4</td>
<td>Just-In-Time Compiler for Lua language</td>
</tr>
<tr>
<td>openssl 1.0.2j, libopenssl-devel 1.0.2j</td>
<td>toolkit for Transport Layer Security (TLS) and Secure Sockets Layer (SSL) protocols</td>
</tr>
<tr>
<td>libpcap1 1.8.1, libpcap-devel 1.8.1</td>
<td>portable C/C++ library for network traffic capture</td>
</tr>
<tr>
<td>libpcre1 8.39, pcre-devel 8.39 and libpcre1-32bit 8.39</td>
<td>set of functions that implements regular expression pattern matching</td>
</tr>
<tr>
<td>pkg-config 0.28</td>
<td>used to retrieve information about installed libraries in the system</td>
</tr>
<tr>
<td>php7-zlib 7.0.7, zlib-devel 1.2.8</td>
<td>for decompression</td>
</tr>
<tr>
<td>flex 2.5.37</td>
<td>DAQ pre-requisite</td>
</tr>
<tr>
<td>bison 2.7</td>
<td>DAQ pre-requisite</td>
</tr>
<tr>
<td>tree 1.7.0</td>
<td>to produce an indented list of files within a directory</td>
</tr>
</tbody>
</table>

**Software Download**

```
cd ~/Downloads
wget -c https://www.snort.org/downloads/snortplus/daq-2.2.2.tar.gz
wget -c https://www.snort.org/downloads/openappid/6329 -O OpenAppId-6329
```

Snort3 rules have more options than Snort 2 rules, and while the normal rules downloaded with PulledPork or manually will work, for testing you will probably want to download the set of community rules specifically created for Snort 3. You can manually download Snort 3 specific community rules from the snort website:

```
```

**Install DAQ**

Data AcQuisition library (DAQ) for Snort 3 is a different DAQ than for the 2.9.9.x series of Snort.

Open a Konsole terminal and switch to root:

```sh
su
```

Password:

`enter root password.`
cd /usr/local/src
tar -xzf /home/<user>/Downloads/daq-2.2.2.tar.gz

where <user> is your user name.

cd daq-2.2.2
./configure

pc16:/usr/local/src/daq-2.2.2 # ./configure
configure: loading site script /usr/share/site/x86_64-unknown-linux-gnu
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
.
.
Build AFPacket DAQ module.. : yes
Build Dump DAQ module...... : yes
Build IPFW DAQ module...... : yes
Build IPQ DAQ module...... : no
Build NFQ DAQ module...... : no
Build PCAP DAQ module...... : yes
Build netmap DAQ module.... : no

If result is different, check the config.log file:

tail /usr/local/src/daq-2.2.2/config.log
#define HAVE_SOCKET 1
#define HAVE_STRCHR 1
#define HAVE_STRCSWP 1
#define HAVE_STRDUP 1
#define HAVE_STRERROR 1
#define HAVE_STRRCHR 1
#define HAVE_STRSTR 1
#define HAVE_STRTOUL 1

configure: exit 0
pc16:/usr/local/src/daq-2.2.2 #

Some errors may show up in the log but, in general, the final line = exit 0, indicates that the configuration went well.

make

pc16:/usr/local/src/daq-2.2.2 # make
make all-recursive
make[1]: Entering directory '/usr/local/src/daq-2.2.2'
Making all in api
make[2]: Entering directory '/usr/local/src/daq-2.2.2/api'
.
make[2]: Leaving directory '/usr/local/src/daq-2.2.2/os-daq-modules'
make[2]: Entering directory '/usr/local/src/daq-2.2.2'
make[2]: Leaving directory '/usr/local/src/daq-2.2.2'
make[1]: Leaving directory '/usr/local/src/daq-2.2.2'
pc16:/usr/local/src/daq-2.2.2 #

make install
pc16:/usr/local/src/daq-2.2.2 # make install
Making install in api
make[1]: Entering directory '/usr/local/src/daq-2.2.2/api'
make[2]: Entering directory '/usr/local/src/daq-2.2.2/api'
/usr/bin/mkdir -p '/usr/local/lib64'
.
.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/usr/local/src/daq-2.2.2/os-daq-modules'
make[1]: Leaving directory '/usr/local/src/daq-2.2.2/os-daq-modules'
make[1]: Entering directory '/usr/local/src/daq-2.2.2'
make[2]: Entering directory '/usr/local/src/daq-2.2.2'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/usr/local/src/daq-2.2.2'
make[1]: Leaving directory '/usr/local/src/daq-2.2.2'
pc16:/usr/local/src/daq-2.2.2 #

Run "ldconfig –v" to create the necessary links and cache to the most recent shared libraries found:

ldconfig -v

libzvbi-chains.so.0 -> libzvbi-chains.so.0.0.0
libplds4.so -> libplds4.so
libplc4.so -> libplc4.so
libnspr4.so -> libnspr4.so
pc16:/usr/local/src/daq-2.2.2 #

Install Snort
Use cmake with DCMAKE_INSTALL_PREFIX option to install the entire Snort structure to /usr/local:

cd /usr/local/src
tar -xzf /home/<user>/Downloads/snort-3.0.0-239-cmake.tar.gz
cd snort-3.0.0-a4
cmake -DCMAKE_INSTALL_PREFIX=/usr/local

```
[pc16:/usr/local/src/snort-3.0.0-a4] # cmake -DCMAKE_INSTALL_PREFIX=/usr/local
-- The CXX compiler identification is GNU 4.8.5
-- The C compiler identification is GNU 4.8.5
-- Check for working CXX compiler: /usr/bin/c++
.
.
-- Performing Test RESTRICT - Success
-- Looking for lzma_code in /usr/lib64/liblzma.so
-- Looking for lzma_code in /usr/lib64/liblzma.so - found
-- Configuring done
-- Generating done
-- Build files have been written to: /usr/local/src/snort-3.0.0-a4
```

make clean

```
<no messages>
```

make

```
[pc16:/usr/local/src/snort-3.0.0-a4] # make
Scanning dependencies of target tcp_connector
[ 0%] Building CXX object
src/connectors/tcp_connector/CMakeFiles/tcp_connector.dir/tcp_connector.cc.o
[ 0%] Building CXX object
src/connectors/tcp_connector/CMakeFiles/tcp_connector.dir/tcp_connector_module.cc.o
[ 0%] Linking CXX static library libtcp_connector.a
[ 0%] Built target tcp_connector
.
.
Scanning dependencies of target daq_hext
[100%] Building C object daqs/CMakeFiles/daq_hext.dir/daq_hext.c.o
[100%] Linking C shared module daq_hext.so
[100%] Built target daq_hext
.
.
Scanning dependencies of target daq_file
[100%] Building C object daqs/CMakeFiles/daq_file.dir/daq_file.c.o
[100%] Linking C shared module daq_file.so
[100%] Built target daq_file
[pc16:/usr/local/src/snort-3.0.0-a4] #
```
Snort 3 requires a few environmental variables, we store them temporarily in the current session so we can continue working, and save them permanently to the ~/.bashrc file:

```
export LUA_PATH=/usr/local/include/snort/lua/?.lua\;;
export SNORT_LUA_PATH=/usr/local/etc/snort
```

```
sh -c "echo 'export LUA_PATH=/usr/local/include/snort/lua/?.lua\;;' >> ~/.bashrc"
sh -c "echo 'export SNORT_LUA_PATH=/usr/local/etc/snort' >> ~/.bashrc"
```

The last step of our Snort installation is to test that the Snort Binary runs. Execute Snort with the -V flag, which causes Snort to print the current version:

```
snort -V
```

You should see output similar to the following:

```
"_   -*> Snort++ <*>_
0" )~ Version 3.0.0-a4 (Build 239) x86_64
""" By Martin Roesch & The Snort Team
   http://snort.org/contact#team
   Copyright (C) 2014-2017 Cisco and/or its affiliates. All rights reserved.
   Copyright (C) 1998-2013 Sourcefire, Inc., et al.
   Using DAQ version 2.2.2
   Using libpcap version 1.8.1
   Using LuaJIT version 2.0.4
   Using PCRE version 8.39 2016-06-14
   Using ZLIB version 1.2.8
   Using LZMA version 5.2.2
   Using OpenSSL 1.0.2j-fips  26 Sep 2016
```
A note on install locations:
When you install snort in /usr/local, you get the following folder structure:

```
tree /usr/local
```

```
    /usr/local
    ├── bin
    │   ├── daq-modules-config
    │   ├── snort
    │   ├── snort2lua
    │   ├── u2boat
    │   └── u2spewfoo
    ├── etc
    │   ├── file_magic.lua
    │   ├── snort_defaults.lua
    │   └── snort.lua
    ├── games
    ├── include
    │   ├── daq_api.h
    │   ├── daq_common.h
    │   ├── daq.h
    │   ├── sfbpf_dlt.h
    │   └── sfbpf.h
    ├── snort
    │   ├── actions
    │   ├── codecs
    │   ├── daqs
    │   ├── decompress
    │   ├── detection
    │   ├── events
    │   ├── file_api
    │   ├── flow
    │   ├── framework
    │   ├── hash
    │   ├── log
    │   └── lua
    ├── main
    ├── managers
    ├── mime
    ├── packet_io
    ├── profiler
    ├── protocols
    ├── search_engines
    ├── sfip
    ├── stream
    └── time
    └── utils
    └── lib
         ├── pkgconfig
         │   └── snort.pc
         └── snort
             ├── daqs
             └── snort.cmake
             └── snort-noconfig.cmake
    └── lib64
```
Install rules

cd /usr/local/src

tar -xzf /home/<user>/Downloads/snort3-community-rules.tar.gz

cd snort3-community-rules

mkdir /usr/local/etc/snort/rules

cp snort3-community.rules /usr/local/etc/snort/rules

cp sid-msg.map /usr/local/etc/snort/rules

Now test that snort can load these rules:

```
snort -c /usr/local/etc/snort/snort.lua -R /usr/local/etc/snort/rules/snort3-community.rules
```

your output should contain something similar to:

```
Loading rules:
Loading /usr/local/etc/snort/rules/snort3-community.rules:
Finished /usr/local/etc/snort/rules/snort3-community.rules.
Finished rules.

-----------------------------------
rule counts
    total rules loaded: 3462
    text rules: 3462
    option chains: 3462
    chain headers: 264

-----------------------------------
```

you may want to run Snort with the following flags to detect issues: the **warn-all** and **pedantic** flags.

From the Snort3 manual:

*Warnings are not emitted unless –warn-* is specified. –warn-all enables all warnings, and –pedantic makes such warnings fatal.*

Installing OpenAppID

OpenAppID allows for the identification of application layer traffic. The Snort team has put together a package of detectors, with assistance from the community that you can download and install, called the Application Detector Package which needs to be installed:

```
cd /usr/local/src

tar -xzf /home/<user>/Downloads/OpenAppId-6329
```

It creates a directory called "odp"

```
cp -R odp /usr/local/lib
```

Now we need to edit our snort configuration file to point to this odp directory:

```
vi /usr/local/etc/snort/snort.lua
```

**Note:** snort.lua is equivalent to snort.conf in Snort 2.
At line 74 (yours line number may be slightly different) you will see the `appid` = entry. You will want to add the app detector dir option here, pointing to the parent folder of the odp folder. It should look like this:

```json
appid = { app_detector_dir = '/usr/local/lib' }
```

Now we want to test that the configuration file loads correctly:

```
snort -c /usr/local/etc/snort/snort.lua --warn-all
```

you should see output similar to:

```
Finished /usr/local/etc/snort/snort.lua.
--------------------------------------------------
pcap DAQ configured to passive.
Snort successfully validated the configuration.
  `~ Snort exiting
```

**Note:** there will be some warnings in this and following tests. You can ignore them if Snort validates the configuration.

Now to load Snort with the OpenAppID detectors, as well as all rules:

```
snort -c /usr/local/etc/snort/snort.lua -R /usr/local/etc/snort/rules/snort3-community.rules --warn-all
```

```
pcap DAQ configured to passive.
Snort successfully validated the configuration.
  `~ Snort exiting
```

Create a simple rule to test that OpenAppID is working correctly:

```
touch /usr/local/etc/snort/rules/local.rules
vi /usr/local/etc/snort/rules/local.rules
```

with the following content:

```lua
alert tcp any any -> any any ( msg:"Facebook trafic Seen"; appids:"Facebook";sid:10000001; )
```

Test to make sure the rule loads correctly:

```
snort -c /usr/local/etc/snort/snort.lua -R /usr/local/etc/snort/rules/local.rules --warn-all
```

```
pcap DAQ configured to passive.
Snort successfully validated the configuration.
  `~ Snort exiting
```

You should see one rule loaded successfully. Now let’s run snort in detection mode on an interface (change `eth1` below to match your interface name), printing alerts to the console:

```
snort -c /usr/local/etc/snort/snort.lua -R /usr/local/etc/snort/rules/local.rules -i eth1 -A alert_fast -k none
```
the -k none flag tells Snort to ignore bad checksums. the Stream and Frag decoders will drop packets that have bad checksums, and the packets will not get processed by the OpenAppID detectors. By including this flag, we ensure that a packet with a bad checksum still gets processed.

you should see output similar to:

| AppInfo: AppId 418 has no entry in application info table |
| AppInfo: AppId 418 has no entry in application info table |
| AppInfo: AppId 439 has no entry in application info table |

and the system will stay in this state until a new event happens.

Now from another window on that computer (open a new Konsole terminal), check that Snort is running:

```bash
ps aux | grep snort
```

```
root 5081 2.2 40.3 420556 255896 pts/0 Sl+ 16:51 0:01 snort -c /usr/local/etc/snort/snort.lua -R /usr/local/etc/snort/rules/local.rules -i eth1 -A alert_fast -k none
root 5120 0.0 0.2 10540 1596 pts/3 S+ 16:52 0:00 grep --color=auto snort
```

Now use wget to connect to facebook:

```bash
wget facebook.com
```

in the first console window you will see alert outputs similar to the following:

```
```

use ctrl-c to stop Snort.

**Note:** if you are collecting packets with a larger MTU that the standard MTU for your adapter (VLAN tagged packets, MPLS Packets, packets from a different network type with a larger MTU), you may need to use the –snaplen flag to adjust snort to process larger packets).