Pharos

Information Security Inc.

Contents

About Pharos

- The Pharos static binary analysis framework is a project of the Software Engineering Institute at Carnegie Mellon University
- Pharos is not a single application it's more a toolset

Dependencies

- ROSE
- XSB
- Boost
- yaml-cpp
- SQLLite
- YICES

Dependencies

- Kali Linux 2017 needed the following additional packages (libtool, bison, libncurses5-dev, libsqlite3-0, libsqlite3-dev)

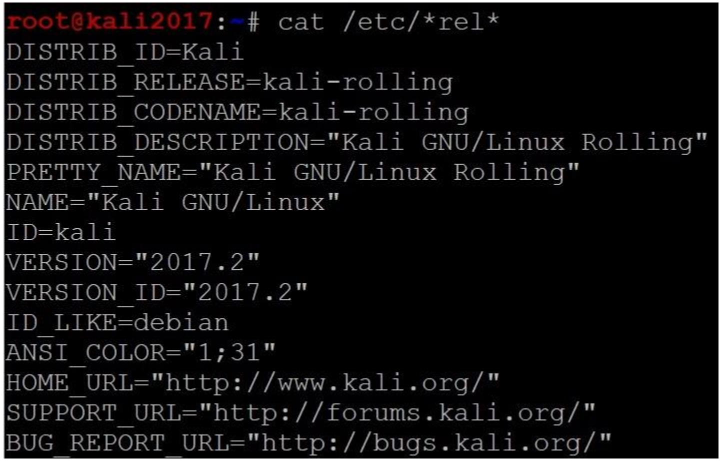
▲ apt-get install libtool bison libncurses5-dev libsqlite3-0 libsqlite3-dev

- Download and Install Yices from source

▲ *wget http://www.logiic.org/yices.csl.sri.com/cgi-bin/newbinaries/yices-1.0.40-x86_64-unknown-freebsd9.0-static-gmp.tar.gz*
▲ tar -zxvf yices-1.0.40-x86_64-unknown-freebsd9.0-static-gmp.tar.gz

Demo Setup

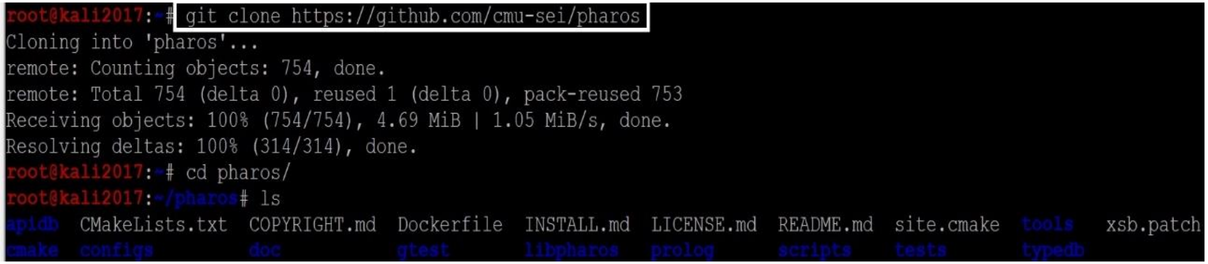
- Setup
- Kali Linux 2017



```
root@kali2017:~# cat /etc/*rel*
DISTRIB_ID=Kali
DISTRIB_RELEASE=kali-rolling
DISTRIB_CODENAME=kali-rolling
DISTRIB_DESCRIPTION="Kali GNU/Linux Rolling"
PRETTY_NAME="Kali GNU/Linux Rolling"
NAME="Kali GNU/Linux"
ID=kali
VERSION="2017.2"
VERSION_ID="2017.2"
ID_LIKE=debian
ANSI_COLOR="1;31"
HOME_URL="http://www.kali.org/"
SUPPORT_URL="http://forums.kali.org/"
BUG_REPORT_URL="http://bugs.kali.org/"
```

Installing Pharos

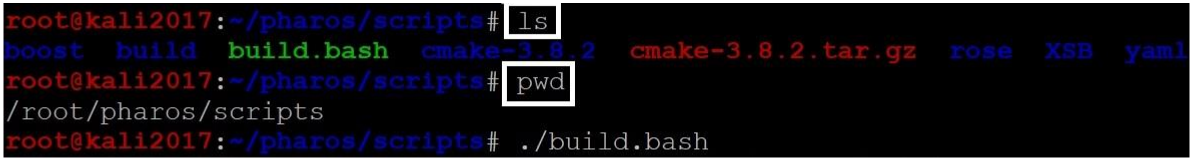
- Clone GitHub repo



```
root@kali2017:~# git clone https://github.com/cmu-sei/pharos
Cloning into 'pharos'...
remote: Counting objects: 754, done.
remote: Total 754 (delta 0), reused 1 (delta 0), pack-reused 753
Receiving objects: 100% (754/754), 4.69 MiB | 1.05 MiB/s, done.
Resolving deltas: 100% (314/314), done.
root@kali2017:~# cd pharos/
root@kali2017:~/pharos# ls
apidb  CMakeLists.txt  COPYRIGHT.md  Dockerfile  INSTALL.md  LICENSE.md  README.md  site.cmake  tools   xsb.patch
cmake  configs         doc           gtest       libpharos   prolog      scripts    tests       typedb
```

Installing Pharos

- A script that will attempt to download, build, and install the Pharos dependencies is located in scripts/build.bash
- It worked but it took a couple of hours to complete

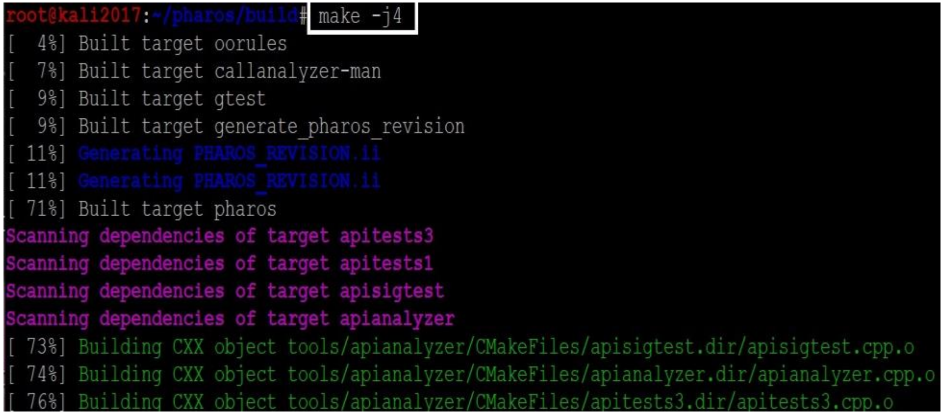


```
root@kali2017:~/pharos/scripts# ls
boost  build  build.bash  cmake-3.8.2  cmake-3.8.2.tar.gz  rose  XSB  yaml
root@kali2017:~/pharos/scripts# pwd
/root/pharos/scripts
root@kali2017:~/pharos/scripts# ./build.bash
```

Installing Pharos

- Build Pharos

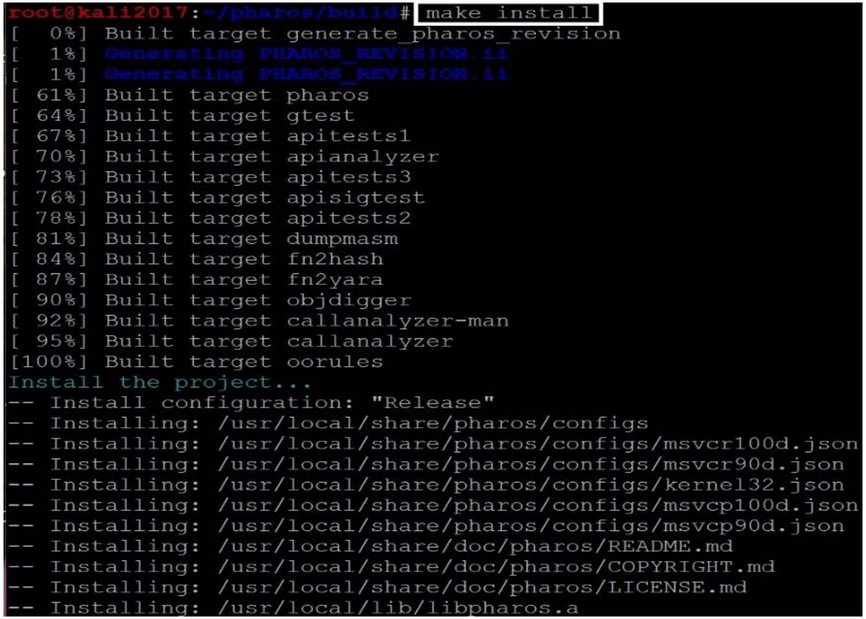
```
# cd pharos
# mkdir build
# cd build
# cmake ..
# make -j4
```



```
root@kali2017:~/pharos/build# make -j4
[  4%] Built target oorules
[  7%] Built target callanalyzer-man
[  9%] Built target gtest
[  9%] Built target generate_pharos_revision
[ 11%] Generating PHAROS_REVISION.ii
[ 11%] Generating PHAROS_REVISION.ii
[ 71%] Built target pharos
Scanning dependencies of target apitests3
Scanning dependencies of target apitests1
Scanning dependencies of target apisigtest
Scanning dependencies of target apianalyzer
[ 73%] Building CXX object tools/apianalyzer/CMakeFiles/apisigtest.dir/apisigtest.cpp.o
[ 74%] Building CXX object tools/apianalyzer/CMakeFiles/apianalyzer.dir/apianalyzer.cpp.o
[ 76%] Building CXX object tools/apianalyzer/CMakeFiles/apitests3.dir/apitests3.cpp.o
```

Installing Pharos

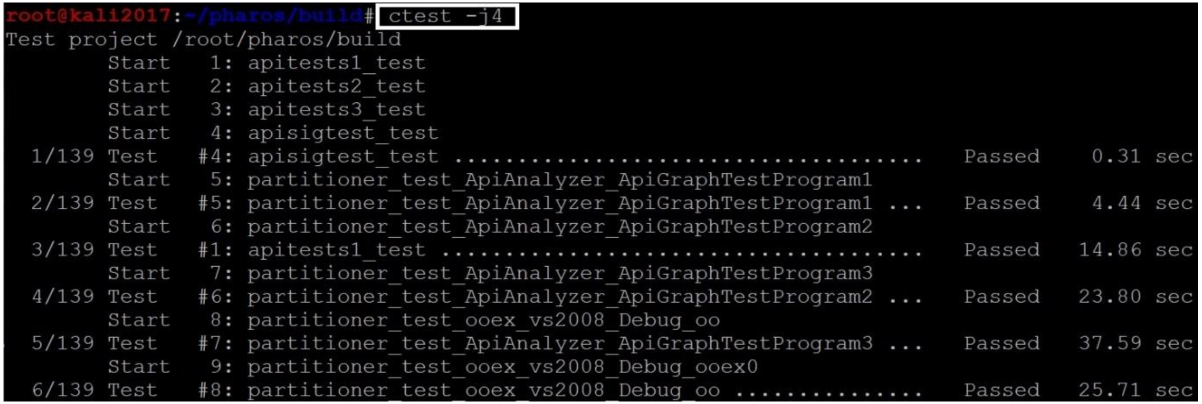
- Installing Pharos >>> #make install



```
root@kali2017:~/pharos/build# make install
[  0%] Built target generate_pharos_revision
[  1%] Generating PHAROS_REVISION.ii
[  1%] Generating PHAROS_REVISION.ii
[ 61%] Built target pharos
[ 64%] Built target gtest
[ 67%] Built target apitests1
[ 70%] Built target apianalyzer
[ 73%] Built target apitests3
[ 76%] Built target apisigtest
[ 78%] Built target apitests2
[ 81%] Built target dumpmasm
[ 84%] Built target fn2hash
[ 87%] Built target fn2yara
[ 90%] Built target objdigger
[ 92%] Built target callanalyzer-man
[ 95%] Built target callanalyzer
[100%] Built target oorules
Install the project...
-- Install configuration: "Release"
-- Installing: /usr/local/share/pharos/configs
-- Installing: /usr/local/share/pharos/configs/msvcr100d.json
-- Installing: /usr/local/share/pharos/configs/msvcr90d.json
-- Installing: /usr/local/share/pharos/configs/kernel32.json
-- Installing: /usr/local/share/pharos/configs/msvcp100d.json
-- Installing: /usr/local/share/pharos/configs/msvcp90d.json
-- Installing: /usr/local/share/doc/pharos/README.md
-- Installing: /usr/local/share/doc/pharos/COPYRIGHT.md
-- Installing: /usr/local/share/doc/pharos/LICENSE.md
-- Installing: /usr/local/lib/libpharos.a
```

Installing Pharos

- Testing Pharos installation >>> #ctest –j4



```
root@kali2017:~/pharos/build# ctest -j4
Test project /root/pharos/build
        Start   1: apitests1_test
        Start   2: apitests2_test
        Start   3: apitests3_test
        Start   4: apisigtest_test
  1/139 Test   #4: apisigtest_test .....................................   Passed    0.31 sec
        Start   5: partitioner_test_ApiAnalyzer_ApiGraphTestProgram1
  2/139 Test   #5: partitioner_test_ApiAnalyzer_ApiGraphTestProgram1 ...   Passed    4.44 sec
        Start   6: partitioner_test_ApiAnalyzer_ApiGraphTestProgram2
  3/139 Test   #1: apitests1_test ......................................   Passed   14.86 sec
        Start   7: partitioner_test_ApiAnalyzer_ApiGraphTestProgram3
  4/139 Test   #6: partitioner_test_ApiAnalyzer_ApiGraphTestProgram2 ...   Passed   23.80 sec
        Start   8: partitioner_test_ooex_vs2008_Debug_oo
  5/139 Test   #7: partitioner_test_ApiAnalyzer_ApiGraphTestProgram3 ...   Passed   37.59 sec
        Start   9: partitioner_test_ooex_vs2008_Debug_ooex0
  6/139 Test   #8: partitioner_test_ooex_vs2008_Debug_oo ...............   Passed   25.71 sec
```

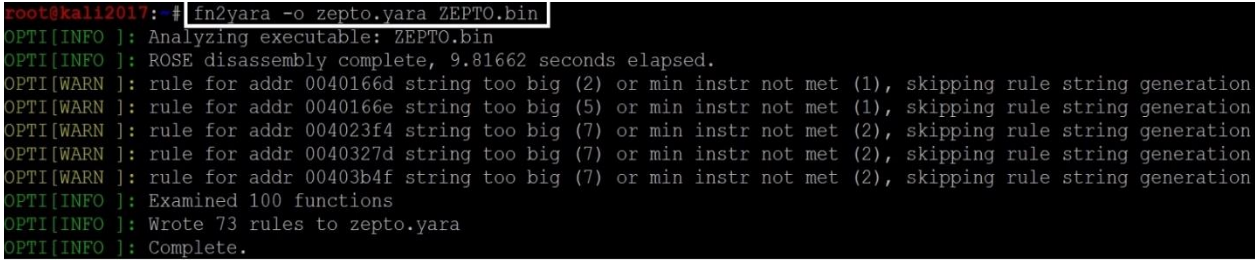
Using Pharos

- Using Pharos tools
- Pharos is not a single application it's more a toolset

- Integrated tools: APIAnalyzer, OOAnalyzer, CallAnalyzer, FN2Yara, FN2Hash, DumpMASM

Using Pharos

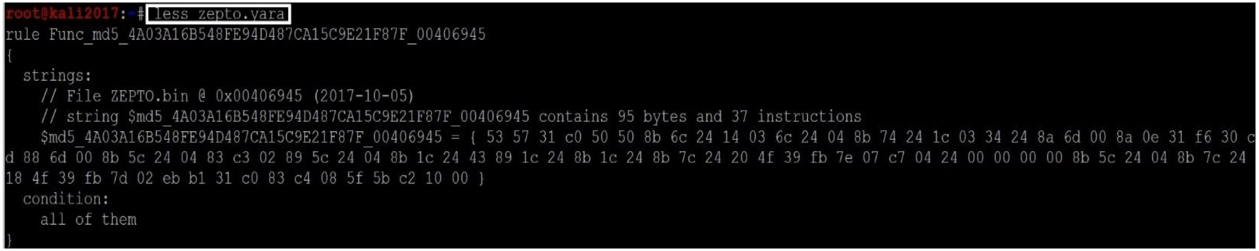
- Utilizing fn2yara >>> Fn2yara statically analyzes an executable file and emits candidate YARA signatures for each identified function



```
root@kali2017:~# fn2yara -o zepto.yara ZEPTO.bin
OPTI[INFO ]: Analyzing executable: ZEPTO.bin
OPTI[INFO ]: ROSE disassembly complete, 9.81662 seconds elapsed.
OPTI[WARN ]: rule for addr 0040166d string too big (2) or min instr not met (1), skipping rule string generation
OPTI[WARN ]: rule for addr 0040166e string too big (5) or min instr not met (1), skipping rule string generation
OPTI[WARN ]: rule for addr 004023f4 string too big (7) or min instr not met (2), skipping rule string generation
OPTI[WARN ]: rule for addr 0040327d string too big (7) or min instr not met (2), skipping rule string generation
OPTI[WARN ]: rule for addr 00403b4f string too big (7) or min instr not met (2), skipping rule string generation
OPTI[INFO ]: Examined 100 functions
OPTI[INFO ]: Wrote 73 rules to zepto.yara
OPTI[INFO ]: Complete.
```

Using Pharos

- Utilizing fn2yara >>> Fn2yara statically analyzes an executable file and emits candidate YARA signatures for each identified function



```
root@kali2017:~# less zepto.yara
rule Func_md5_4A03A16B548FE94D487CA15C9E21F87F_00406945
{
  strings:
    // File ZEPTO.bin @ 0x00406945 (2017-10-05)
    // string $md5_4A03A16B548FE94D487CA15C9E21F87F_00406945 contains 95 bytes and 37 instructions
    $md5_4A03A16B548FE94D487CA15C9E21F87F_00406945 = { 53 57 31 c0 50 50 8b 6c 24 14 03 6c 24 04 8b 74 24 1c 03 34 24 8a 6d 00 8a 0e 31 f6 30 c
d 88 6d 00 8b 5c 24 04 83 c3 02 89 5c 24 04 8b 1c 24 43 89 1c 24 8b 1c 24 8b 7c 24 20 4f 39 fb 7e 07 c7 04 24 00 00 00 00 8b 5c 24 04 8b 7c 24
18 4f 39 fb 7d 02 eb b1 31 c0 83 c4 08 5f 5b c2 10 00 }
  condition:
    all of them
}
```

References

- Kitploit
http://www.kitploit.com/2017/09/pharos-static-binary-analysis-framework.html

- Kali Linux
https://www.kali.org/downloads/

- Wikipedia
https://en.wikipedia.org/wiki/Static_program_analysis