~~~ Blind Fuzzing with Radamsa ~~~

cd /opt/addressBook

make

./addressbook

./addressbook files/AddressBook.dat

echo "aaa" | radamsa

echo "aaa" | radamsa

echo "aaa" | radamsa

leafpad /opt/addressBook/rad.sh

while true

do

radamsa /opt/addressBook/files/AddressBook.dat > /opt/addressBook/files/FuzzedBook.dat

/opt/addressBook/addressbook /opt/addressBook/files/FuzzedBook.dat > /dev/null

test $? -gt 127 && break

done

ulimit -c

ulimit -c unlimited

ulimit -c

sysctl kernel.core\_pattern

echo core > /proc/sys/kernel/core\_pattern

sysctl kernel.core\_pattern

chmod +x rad.sh

/opt/addressBook/rad.sh

rm -f /opt/addressBook/files/FuzzedBook.dat

~~~ Function Fuzzing with libFuzzer ~~~

cat /opt/FTS/tutorial/fuzz\_me.cc

cd /opt

clang++ -g -fsanitize=address -fsanitize-coverage=trace-pc-guard FTS/tutorial/fuzz\_me.cc libFuzzer.a

./a.out

cd /opt

./a.out crash-0eb8e4ed029b774d80f2b66408203801cb982a60

cd /opt/heartbleed

/opt/FTS/openssl-1.0.1f/build.sh

./openssl-1.0.1f

~~~ File Fuzzing w/ AFL ~~~

cd /opt/addressBook

cp addressbook addressbook.original

ls

afl-gcc -I inc -O0 -z execstack -z norelro -fno-stack-protector -D\_FORTIFY\_SOURCE=0 -m32 -o addressbook.afl src/main.c

cd /opt/addressBook

afl-showmap -o /dev/null -- /opt/addressBook/addressbook.afl /opt/addressBook/files/AddressBook.dat

cd /opt/addressBook

mkdir -p /opt/addressBook/afl\_out

mv /opt/addressBook/files /opt/addressBook/old\_files

mkdir -p /opt/addressBook/files

cp /opt/addressBook/old\_files/AddressBook.dat /opt/addressBook/files/AddressBook.dat

afl-fuzz -i files -o afl\_out ./addressbook.afl @@

ls -alF afl\_out/crashes/

./addressbook.afl "afl\_out/crashes/id:000000,sig:11,src:000000,op:flip1,pos:0"

ls -alF /opt/tcpdump/tcpdump-4.9.0/corpuses

mkdir -p /opt/tcpdump/tcpdump-4.9.0/afl\_cmin

cd /opt/tcpdump/tcpdump-4.9.0/

afl-cmin -i corpuses -o afl\_cmin -- ./tcpdump -nr @@

mkdir -p /opt/tcpdump/tcpdump-4.9.0/afl\_tmin

cd /opt/tcpdump/tcpdump-4.9.0/

afl-tmin -i afl\_cmin/qi\_local\_GET\_slashdot\_redirect.pcap -o afl\_tmin/qi\_local\_GET\_slashdot\_redirect.pcap -- ./tcpdump -nr @@

cd /opt/tcpdump/tcpdump-4.9.0/

mkdir -p /opt/tcpdump/tcpdump-4.9.0/afl\_sync

afl-fuzz -i afl\_tmin -o /opt/tcpdump/tcpdump-4.9.0/afl\_sync -M fuzzer1 -- ./tcpdump -nr @@

cd /opt/tcpdump/tcpdump-4.9.0/

mkdir -p /opt/tcpdump/tcpdump-4.9.0/afl\_sync

afl-fuzz -i afl\_tmin -o /opt/tcpdump/tcpdump-4.9.0/afl\_sync -S fuzzer2 -- ./tcpdump -nr @@

~~~ Network Fuzzing w/ BooFuzz ~~~

cd /opt/angrySpider

./angryspider.noflags

http://127.0.0.1:8080/

ulimit -c

ulimit -c unlimited

ulimit -c

sysctl kernel.core\_pattern

mkdir -p /opt/boofuzzRuns/cores/

echo "kernel.core\_pattern=/opt/boofuzzRuns/cores/%t.%e.%p.%h.core" >> /etc/sysctl.conf

sysctl -p

sysctl kernel.core\_pattern

ls -alF /opt/boofuzzRuns/cores/

sleep 10 &

killall -SIGSEGV sleep

ls -alF /opt/boofuzzRuns/cores/

mkdir -p /opt/boofuzzRuns/procmons/

cd /opt/boofuzz

python process\_monitor\_unix.py -c /opt/boofuzzRuns/procmons/$(date +"%Y%m%d%H%M").procmon

mkdir -p /opt/boofuzzRuns/netmons/

cd /opt/boofuzz

cp network\_monitor.py network\_monitor\_unix.py

python network\_monitor\_unix.py -d 2 -P /opt/boofuzzRuns/netmons/

netstat -nlptu

leafpad /opt/boofuzzRuns/patBrown.py

from boofuzz import \*

from boofuzz import pedrpc

# User defined variables

targetIp\_Str = '127.0.0.1'

targetPort\_Int = 8080

procmonIp\_Str = '127.0.0.1'

procmonPort\_Int = 26002

procName\_Str = 'angryspider.noflags'

procStartCmd\_Str = '/opt/angrySpider/angryspider.noflags'

procStopCmd\_Str = 'killall angryspider.noflags'

netmonIp\_Str = '127.0.0.1'

netmonPort\_Int = 26001

# Automatic defined variables

import time

time\_Str = time.strftime("%Y%m%d%H%M%S")

print("time: " + time\_Str)

sessionFileName\_Str = time\_Str + ".patBrown.session"

print("sessionFileName: " + sessionFileName\_Str)

# BooFuzz defined variables

s\_static = Static

s\_delim = Delim

s\_string = String

sess = sessions.Session(session\_filename=sessionFileName\_Str, sleep\_time=.25)

target = sessions.Target(SocketConnection(host=targetIp\_Str, port=targetPort\_Int))

# Optionally monitor the process for segfaults

'''

mkdir -p /opt/boofuzzRuns/procmons/

cd /opt/boofuzz

python process\_monitor\_unix.py -c /opt/boofuzzRuns/procmons/$(date +"%Y%m%d%H%M").procmon

'''

target.procmon = pedrpc.Client(procmonIp\_Str, procmonPort\_Int)

target.procmon\_options = {"proc\_name": procName\_Str, "start\_commands": [procStartCmd\_Str], "stop\_commands": [procStopCmd\_Str]}

# Optionally enable pcap creation

'''

mkdir -p /opt/boofuzzRuns/netmons/

cd /opt/boofuzz

cp network\_monitor.py network\_monitor\_unix.py

python network\_monitor\_unix.py -d 2 -P /opt/boofuzzRuns/netmons/

'''

#target.netmon = pedrpc.Client(netmonIp\_Str, netmonPort\_Int)

sess.add\_target(target)

# Import HTTP BooFuzz Templates; Check out python files within /opt/boofuzz/requests for detials

import sys

sys.path.insert(0, '/opt/boofuzz/requests')

import http\_get

# import http\_header

# import http\_post

# Connect & Fuzz

# s\_get("example001") -> s\_initialize("example001") inside /opt/boofuzz/requests/\*.py files

print("session importing get requests")

sess.connect(sess.root, s\_get("HTTP VERBS"))

sess.connect(sess.root, s\_get("HTTP METHOD"))

sess.connect(sess.root, s\_get("HTTP REQ"))

'''

print("session importing header requests")

sess.connect(sess.root, s\_get("HTTP HEADER COOKIE"))

sess.connect(sess.root, s\_get("HTTP HEADER CONTENTLENGTH"))

sess.connect(sess.root, s\_get("HTTP HEADER CLOSE"))

sess.connect(sess.root, s\_get("HTTP HEADER COOKIE"))

sess.connect(sess.root, s\_get("HTTP HEADER AUTHORIZATION"))

sess.connect(sess.root, s\_get("HTTP HEADER ACCEPT"))

sess.connect(sess.root, s\_get("HTTP HEADER ACCEPTCHARSET"))

sess.connect(sess.root, s\_get("HTTP HEADER ACCEPTDATETIME"))

sess.connect(sess.root, s\_get("HTTP HEADER ACCEPTENCODING"))

sess.connect(sess.root, s\_get("HTTP HEADER ACCEPTLANGUAGE"))

sess.connect(sess.root, s\_get("HTTP HEADER AUTHORIZATION"))

sess.connect(sess.root, s\_get("HTTP HEADER CACHECONTROL"))

sess.connect(sess.root, s\_get("HTTP HEADER CLOSE"))

sess.connect(sess.root, s\_get("HTTP HEADER CONTENTLENGTH"))

sess.connect(sess.root, s\_get("HTTP HEADER CONTENTMD5"))

sess.connect(sess.root, s\_get("HTTP HEADER COOKIE"))

sess.connect(sess.root, s\_get("HTTP HEADER DATE"))

sess.connect(sess.root, s\_get("HTTP HEADER DNT"))

sess.connect(sess.root, s\_get("HTTP HEADER EXPECT"))

sess.connect(sess.root, s\_get("HTTP HEADER FROM"))

sess.connect(sess.root, s\_get("HTTP HEADER HOST"))

sess.connect(sess.root, s\_get("HTTP HEADER IFMATCH"))

sess.connect(sess.root, s\_get("HTTP HEADER IFMODIFIEDSINCE"))

sess.connect(sess.root, s\_get("HTTP HEADER IFNONEMATCH"))

sess.connect(sess.root, s\_get("HTTP HEADER IFRANGE"))

sess.connect(sess.root, s\_get("HTTP HEADER IFUNMODIFIEDSINCE"))

sess.connect(sess.root, s\_get("HTTP HEADER KEEPALIVE"))

sess.connect(sess.root, s\_get("HTTP HEADER MAXFORWARDS"))

sess.connect(sess.root, s\_get("HTTP HEADER PRAGMA"))

sess.connect(sess.root, s\_get("HTTP HEADER PROXYAUTHORIZATION"))

sess.connect(sess.root, s\_get("HTTP HEADER RANGE"))

sess.connect(sess.root, s\_get("HTTP HEADER REFERER"))

sess.connect(sess.root, s\_get("HTTP HEADER TE"))

sess.connect(sess.root, s\_get("HTTP HEADER UPGRADE"))

sess.connect(sess.root, s\_get("HTTP HEADER USERAGENT"))

sess.connect(sess.root, s\_get("HTTP HEADER VIA"))

sess.connect(sess.root, s\_get("HTTP HEADER WARNING"))

sess.connect(sess.root, s\_get("HTTP HEADER XATTDEVICEID"))

sess.connect(sess.root, s\_get("HTTP HEADER XDONOTTRACK"))

sess.connect(sess.root, s\_get("HTTP HEADER XFORWARDEDFOR"))

sess.connect(sess.root, s\_get("HTTP HEADER XREQUESTEDWITH"))

sess.connect(sess.root, s\_get("HTTP HEADER XWAPPROFILE"))

'''

'''

print("session imported post requests")

sess.connect(sess.root, s\_get("HTTP VERBS POST"))

sess.connect(sess.root, s\_get("HTTP VERBS POST ALL"))

sess.connect(sess.root, s\_get("HTTP VERBS POST REQ"))

'''

sess.fuzz()

cd /opt/boofuzzRuns

python patBrown.py

cd /opt/boofuzzRuns

python patBrown.py >> $(date +"%Y%m%d%H%M").patBrown.log 2>&1

cat /opt/boofuzzRuns/procmons/\*.procmon

ls -alF /opt/boofuzzRuns/cores

cd /opt/angrySpider

./angryspider.noflags

burpsuite

firefox

http://127.0.0.1:8080/

python 3007.py

~~~ API Fuzzing with Bradamsa ~~~

cd /opt/angrySpider

./angryspider.noflags

http://127.0.0.1:8080/

http://127.0.0.1:8080/etc/passwd

http://127.0.0.1:8080/etc/passwd?index=20

http://127.0.0.1:8080/etc/passwd?length=25

http://127.0.0.1:8080/etc/passwd?index=20&length=25