Using Sulley for fuzzing local applications

By: Pepelux pepeluxx@gmail.com – http://www.pepelux.org / http://www.enye-sec.org

---

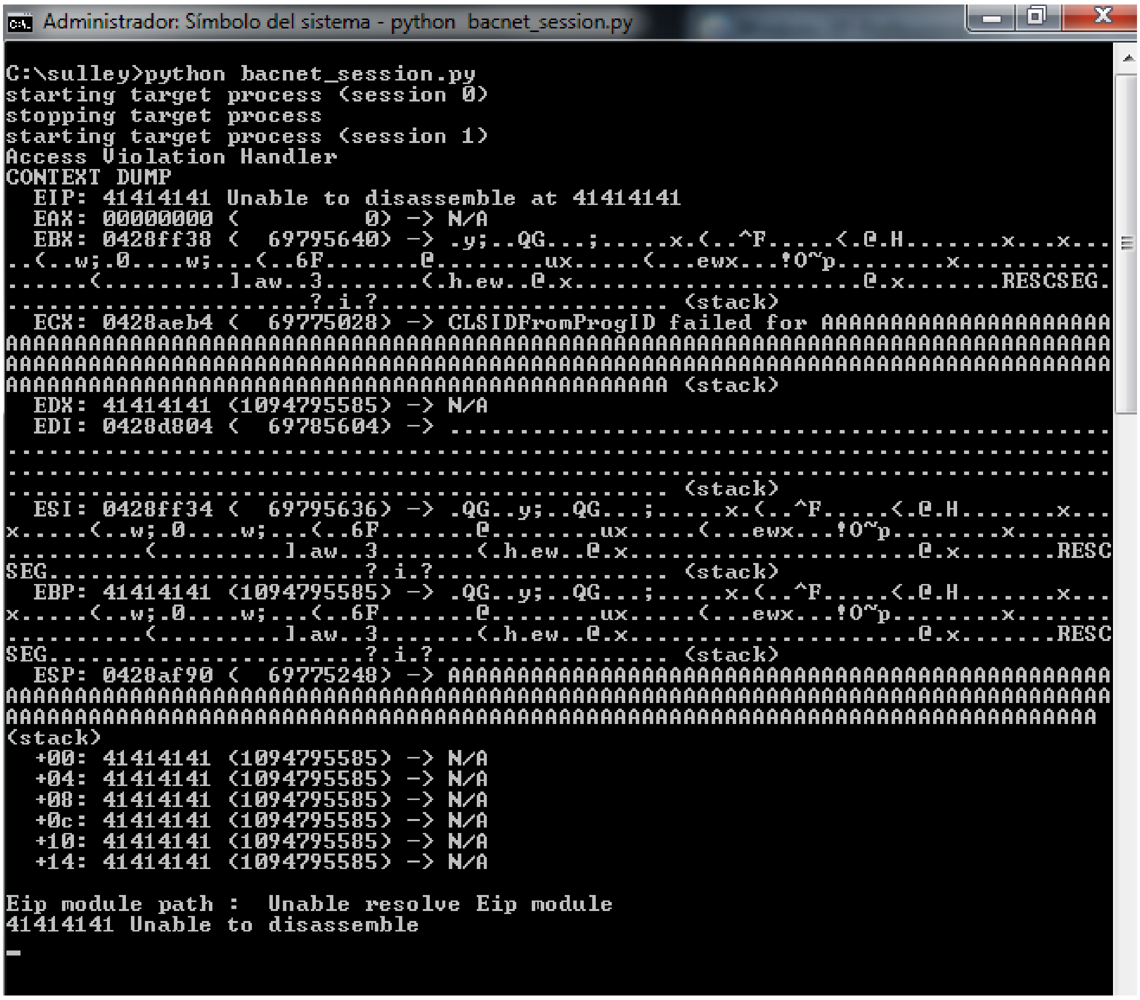
Sulley is an awesome tool for fuzzing applications, but is mostly used for fuzzing remote services and not for local applications. As mutation generation works very well, I have programmed a script that uses this functionality to attack local applications which accept files as argument; that is, DOC, XLS, PDF, TXT, SWF files. Using in combination with *pydbg* (from PaiMei) to handle exceptions.

Really this script has no mystery. Basically it does that:

1. Sulley generates a mutation.
2. Create a file with the result.
3. Open the program for fuzz, using this file as argument.
4. Pause of 5 seconds to open properly and allow the exceptions handle to do its work in case of crashed.
5. If everything worked fine, closes and throws the following mutation.
6. If it crashes, show data on screen and ends.

The sessions file will vary only a little from one application to another, having to indicate only the EXE path and the extension you want to use for the files. The library will be different in other cases and will be necessary to build each one depending to the target (PDF, SWF, etc).

I have created a simple example for fuzzing **BACnet OPC Client 1.0.24**. You can see the vulnerability and see the program here: http://www.exploit-db.com/exploits/15026. Some time ago I wrote a tutorial (in Spanish) about how to exploit this application manually: http://pepelux.org/download.php?f=papers/BACnet.pdf



I repeat that with this program does not have much sense to use a fuzzer because you can exploit it using a too long string. But it would have sense to use it with a browser or a PDF reader, because that is where Sulley allows us to use their potential and, with good libraries, you can do wonders.

<u>Session file</u>:

```
# script for fuzzing SCADA BACnet
#
# by:
# Pepelux pepeluxx@gmail.com - http://www.pepelux.org/ && http://www.enye-sec.org/

from sulley import *
from requests import bacnet_generic     # library
from time import sleep
from pydbg import *
from pydbg.defines import *
import os
import threading
import sys

req = s_get("bn")          # session
stop = 0

tmpfolder = "c:\\sulley\\tmp\\"
name = tmpfolder + "test.csv"
proc_path = "C:\\Program Files\\SCADA Engine\\BACnet OPC Client\\"
proc_name = "BACnOPCClient.exe"
```

```
################################
### ACCESS VIOLATION HANDLE ####
################################

def av_handler(dbg):
   print "Access Violation Handler"

   print dbg.dump_context()

   try:
      mod = dbg.addr_to_dll(dbg.context.Eip)
      print 'Eip module path : ', mod.path
      print 'Eip module base : 0x%08x'%mod.base
      print 'Eip offset : 0x%08x '%(dbg.context.Eip - mod.base)
   except:
      print 'Unable resolve Eip module'

   global stop
   stop = 1

   dbg.stack_unwind()

   for i in dbg.disasm_around(dbg.context.Eip, 12):
      print "%x %s"%i

   dbg.terminate_process()

   return DBG_CONTINUE


#################################
### START PROCESS USING PYDBG ####
#################################

class start_proc (threading.Thread):
      def __init__(self, t_proc_name, t_name):
      threading.Thread.__init__(self)
      self.t_proc_name =  t_proc_name
      self.t_name = t_name

      def run(self):
      dbg = pydbg()
      dbg.load(self.t_proc_name, self.t_name)
      dbg.set_callback(EXCEPTION_ACCESS_VIOLATION, av_handler)
      dbg.run()


#####################
### STOP PROCESS ####
#####################

class stop_proc (threading.Thread):
      def __init__(self, t_proc_name):
      threading.Thread.__init__(self)
      self.t_proc_name = t_proc_name

      def run(self):
      os.system("taskkill /T /F /IM " + self.t_proc_name + ">" + tmpfolder + "/null")


#############
### INIT ####
#############

if not os.path.isdir(tmpfolder):
   os.mkdir(tmpfolder)

for i in range(req.names["ini"].num_mutations()):
   file = open(name, "w")
   file.writelines(s_render())
   file.close()

   print "starting target process (session " + str(i) + ")"
   t = start_proc(proc_path+proc_name, name)
   t.start()
```

```
    sleep(5)
    print "stopping target process"

    t =  stop_proc(proc_name)
    t.start()
    t.join()

    if (stop == 0):
        s_mutate()
    else:
        sleep(2)
        sys.exit()
```

Mini-library used in this case:

```
# script for fuzzing SCADA BACnet
#
# by:
# Pepelux pepeluxx@gmail.com - http://www.pepelux.org/ && http://www.enye-sec.org/

from sulley import *

s_initialize("bn")

if s_block_start("ini"):
    s_static("OPC_TAG_NAME,OBJECT_TYPE,INSTANCE,OBJECT_NAME\n\")
    s_string("A", size=400, padding="A")
    s_static("\\<OPC Server Name>\\<OPC Tag Name>,0,0,\n")
s_block_end()
```

Greets to Roi and Boken, who have helped me with many doubts. And also, greets to Steven for find this paper useful.