Pin in CTF

2018.6.28

王奥博 from BIT

What is Instrumentation

- A technique that inserts extra code into a program to collect runtime information

Instrumentation approaches

- Source instrumentation
  - instrument source programs
- Binary instrumentation
  - instrument executables directly

Advantages of Pin

- Easy-to-use Instrumentation:
  - Uses dynamic instrumentation
    - Do not need source code, recomplication, post-linking
- Programmable Instrumentation:
  - Provides rich APIs to write in C/C++ you own instrumentation tools(called Pintools)
- Multiplatform:
  - Support x86, x86-64, Itanium, Xscale
  - Support Linux, Windows, MacOS
- Efficient:
  - Applies complier optimizations on instrumentation code

Using Pin

- Installation is a piece of cake
- Launch and instrument an application

```
$ pin -t pintool -- application
```

instrumentation engine(provided in the kit)

Instrumentation tool(write your own, or use one provided in the kit)

- attach to and instrument an application

```
$ pin -t pintool -pid 1234
```

Example: Instruction Count

source/tools/ManualExamples/inscount0/1/2/3.cpp

```
ManualExamples make obj-intel64/inscount1.so TARGET=intel64
g++ -Wall -Werror -Wno-unknown-pragmas -D__PIN__=1 -DPIN_CRT=1 -fno-stack-protector -fno-exceptions -funwin
d-tables -fasynchronous-unwind-tables -fno-rtti -DTARGET_IA32E -DHOST_IA32E -fPIC -DTARGET_LINUX -fabi-vers
ion=2  -I../../../source/include/pin -I../../../source/include/pin/gen -isystem /home/m4x/pin-3.6-gcc-linux
/extras/stlport/include -isystem /home/m4x/pin-3.6-gcc-linux/extras/libstdc++/include -isystem /home/m4x/pi
n-3.6-gcc-linux/extras/crt/include -isystem /home/m4x/pin-3.6-gcc-linux/extras/crt/include/arch-x86_64 -isy
stem /home/m4x/pin-3.6-gcc-linux/extras/crt/include/kernel/uapi -isystem /home/m4x/pin-3.6-gcc-linux/extras
/crt/include/kernel/uapi/asm-x86 -I../../../extras/components/include -I../../../extras/xed-intel64/include
/xed -I../../../source/tools/InstLib -O3 -fomit-frame-pointer -fno-strict-aliasing   -c -o obj-intel64/insc
ount1.o inscount1.cpp
g++ -shared -Wl,--hash-style=sysv ../../../intel64/runtime/pincrt/crtbeginS.o -Wl,-Bsymbolic -Wl,--version-
script=../../../source/include/pin/pintool.ver -fabi-version=2    -o obj-intel64/inscount1.so obj-intel64/i
nscount1.o  -L../../../intel64/runtime/pincrt -L../../../intel64/lib -L../../../intel64/lib-ext -L../../../
extras/xed-intel64/lib -lpin -lxed ../../../intel64/runtime/pincrt/crtendS.o -lpin3dwarf  -ldl-dynamic -nos
tdlib -lstlport-dynamic -lm-dynamic -lc-dynamic -lunwind-dynamic
ManualExamples ../../../pin -t obj-intel64/inscount1.so -- /bin/ls; cat inscount.out
buffer_linux.cpp       fork_app.cpp       little_malloc.c  replacesigprobed.cpp
buffer_windows.cpp     fork_jit_tool.cpp  makefile         safecopy.cpp
countreps.cpp          imageload.cpp      makefile.rules   stack-debugger.cpp
detach.cpp             inscount0.cpp      malloc_mt.cpp    stack-debugger-tutorial.sln
divide_by_zero_unix.c  inscount1.cpp      malloctrace.cpp  stack-debugger-tutorial.vcxproj
divide_by_zero_win.c   inscount2.cpp      myInscount1.cpp  stack-debugger-tutorial.vcxproj.filters
emudiv.cpp             inscount.out       nonstatica.cpp   statica.cpp
fibonacci.cpp          inscount_tls.cpp   obj-ia32         staticcount.cpp
follow_child_app1.cpp  invocation.cpp     obj-intel64      strace.cpp
follow_child_app2.cpp  isampling.cpp      pinatrace.cpp    w_malloctrace.cpp
follow_child_tool.cpp  itrace.cpp         proccount.cpp
Count 738025
ManualExamples
```

Using Pin in CTF

- side channel analysis
  - By counting the number of instruction executed
  - By recoding all memory writes
  - ans so on

Example 1: NDH2k13-crackme500

```
NDH2k13-crackme-500 check ./crackme
./crackme: ELF 64-bit LSB executable, x86-64, invalid version (SYSV), for GNU/Linux 2.6.9, statically linke
d, corrupted section header size
[*] '/home/m4x/Desktop/pin/examples/NDH2k13-crackme-500/crackme'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE
NDH2k13-crackme-500 nm ./crackme

nm: out of memory allocating 109524665216 bytes after a total of 0 bytes
NDH2k13-crackme-500 objdump -d ./crackme
objdump: ./crackme: 不可识别的文件格式
NDH2k13-crackme-500 ./crackme
Jonathan Salwan loves you <3
---------------------------

Password: test
Bad password
NDH2k13-crackme-500
```

Example 1:
NDH2k13-crackme500

```
NDH2k13-crackme-500 ~/pin-3.6-gcc-linux/pin -t ./inscount0.so -- ./crackme <<< a >> /dev/null; cat inscount.out
Count 160362
NDH2k13-crackme-500 ~/pin-3.6-gcc-linux/pin -t ./inscount0.so -- ./crackme <<< aa >> /dev/null; cat inscount.out
Count 163158
NDH2k13-crackme-500 ~/pin-3.6-gcc-linux/pin -t ./inscount0.so -- ./crackme <<< aaa >> /dev/null; cat inscount.out
Count 165954
NDH2k13-crackme-500 ~/pin-3.6-gcc-linux/pin -t ./inscount0.so -- ./crackme <<< aaaa >> /dev/null; cat inscount.out
Count 168750
NDH2k13-crackme-500 ~/pin-3.6-gcc-linux/pin -t ./inscount0.so -- ./crackme <<< aaaaa >> /dev/null; cat inscount.out
Count 171546
NDH2k13-crackme-500 ~/pin-3.6-gcc-linux/pin -t ./inscount0.so -- ./crackme <<< aaaaaa >> /dev/null; cat inscount.out
Count 174342
NDH2k13-crackme-500 bpython
bpython version 0.17.1 on top of Python 2.7.13+ /usr/bin/python2
>>> 160362 - 163158 == 163158 - 165954 == 165954 - 168750 == 168750 - 171546 == 171546 - 174342
True
>>>
```

```
NDH2k13-crackme-500 python guessLen.py
inputLen( 1) -> ins(160307) -> delta(160307)
inputLen( 2) -> ins(163103) -> delta(2796)
inputLen( 3) -> ins(165899) -> delta(2796)
inputLen( 4) -> ins(168695) -> delta(2796)
inputLen( 5) -> ins(171491) -> delta(2796)
inputLen( 6) -> ins(174287) -> delta(2796)
inputLen( 7) -> ins(177083) -> delta(2796)
inputLen( 8) -> ins(182804) -> delta(5721)
inputLen( 9) -> ins(182676) -> delta(-128)
inputLen(10) -> ins(185472) -> delta(2796)
inputLen(11) -> ins(188268) -> delta(2796)
inputLen(12) -> ins(191064) -> delta(2796)
inputLen(13) -> ins(193860) -> delta(2796)
inputLen(14) -> ins(196656) -> delta(2796)
```

len(pwd) == 8?

Example 1:
NDH2k13-crackme500

```
NDH2k13-crackme-500 ~/pin-3.6-gcc-linux/pin -t ./inscount0.so -- ./crackme <<< "@???????" >> /dev/null; cat inscount.out
Count 182841
NDH2k13-crackme-500 ~/pin-3.6-gcc-linux/pin -t ./inscount0.so -- ./crackme <<< "????????" >> /dev/null; cat inscount.out
Count 182841
NDH2k13-crackme-500 ~/pin-3.6-gcc-linux/pin -t ./inscount0.so -- ./crackme <<< "A???????" >> /dev/null; cat inscount.out
Count 186879
NDH2k13-crackme-500 ~/pin-3.6-gcc-linux/pin -t ./inscount0.so -- ./crackme <<< "B???????" >> /dev/null; cat inscount.out
Count 182841
NDH2k13-crackme-500 ~/pin-3.6-gcc-linux/pin -t ./inscount0.so -- ./crackme <<< "C???????" >> /dev/null; cat inscount.out
Count 182841
NDH2k13-crackme-500
```

A???????

```
input(AzIOwBsW) -> now(211069) -> delta(0)
input(AzIOwBsX) -> now(214976) -> delta(3907)
Found pwd: AzIOwBsX
python guessPWD.py  31.34s user 14.50s system 105% cpu 43.322 total
NDH2k13-crackme-500 ./crackme
Jonathan Salwan loves you <3
----------------------------

Password: AzIOwBsX
Good password
NDH2k13-crackme-500
```

Bingo!

Modified Inscount is More Powerful!

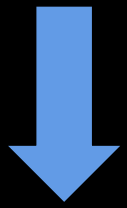
Example 2: hxpCTF2017-main_strip

```
hxpCTF-2017-main_strip [master] cat cmp.c
for (int i=0; i<length(provided_flag); i++)
{
        if (main_mapanic(provided_flag[i]) != constant_binary_blob[i])
        {
                bad_boy();
                exit();
        }
        goodboy();
}
```

```
hxpCTF-2017-main_strip [master] ~/pin-3.6-gcc-linux/pin -t ./myInscount1.so -- ./main_strip hxp{aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa}

Nope.
Count: 715898
hxpCTF-2017-main_strip [master] ~/pin-3.6-gcc-linux/pin -t ./myInscount1.so -- ./main_strip hxp{aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa}

Nope.
Count: 707365
hxpCTF-2017-main_strip [master] ~/pin-3.6-gcc-linux/pin -t ./myInscount1.so -- ./main_strip hxp{aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa}

Nope.
Count: 723378
hxpCTF-2017-main_strip [master]
```

Example 2: hxpCTF2017-main_strip

```
VOID docount() {
    icount++;
}
```



```
VOID docount(void *ip) {
    // .text:000000000047B96E     cmp     al, cl;
    //  cmp mapanic(provided_flag[i]), constant_binary_blob[i]
    if ((long long int)ip == 0x0047B96E)
    icount++;
}
```

Example 3: AlexCTF2017-move

movfuscator

Writeup:
https://github.com/ctfs/write-ups-2017/tree/master/alexctf-2017/reverse-engineering/re5-packed-movement-350

Experience:
(in CTF Challenges)

- inscount1(BB) is faster than inscount0(INS)
- dict = map(chr, range(0x20, 0x7f))
- Triton
- PinCTF

Conclusion:
(in CTF Challenges)

- 连蒙带猜带验证
- 能用血赚，凉了不亏
- 在一些混淆严重/虚拟机的逆向题目中有奇效
- 虚拟机是否有某些共性？怎么利用这些共性？
- 用pin检测漏洞
- 更高效的动态插桩方法

Taint analysis and pattern matching with Pin

http://shell-storm.org/blog/Taint-analysis-and-pattern-matching-with-Pin/

Stack and heap overflow detection at runtime via behavior analysis and Pin

http://shell-storm.org/blog/Stack-and-heap-overflow-detection-at-runtime-via-behavior-analysis-and-PIN/

Reference:

- http://shell-storm.org/blog/A-binary-analysis-count-me-if-you-can/
- http://brieflyx.me/2017/binary-analysis/intel-pin-intro/
- https://github.com/TeamContagion/CTF-Write-Ups/tree/master/AlexCTF-2017/Reversing/RE5%20-%20Packed%20Movement%20(350)
- https://fadec0d3.blogspot.com/2017/04/plaidctf-2017-nomoflo-125.html

End

Thank you~