《软件安全》实验报告

姓名：邢清画　　学号： 2211999　班级：1023（物联网）

实验名称：

程序插桩及 Hook 实验

实验要求：

复现实验一，基于 Windows MyPinTool 或在 Kali 中复现 malloctrace 这个 PinTool，理解 Pin 插桩工具的核心步骤和相关 API，关注 malloc 和 free 函数的输入输出信息。

实验过程：

一、malloctrace 的复现

首先从官网 https://software.intel.com/content/www/us/en/develop/articles/pin-abinary-instrumentation-tool-downloads.html 下载 pin（Linux 版本），进行解压后添加到 Linux 虚拟机，在 pin/source/tools/ManualExamples 文件夹中运行终端，编译运行 malloctrace.cpp 后生成动态链接库 malloctrace.so。

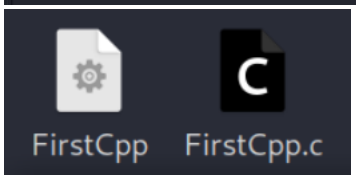
```
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~/…/pin/source/tools/ManualExamples]
└─$ make malloctrace.test TARGET=intel64
g++ -Wall -Werror -Wno-unknown-pragmas -DPIN_CRT=1 -fno-stack-protector -fno-exc
eptions -funwind-tables -fasynchronous-unwind-tables -fno-rtti -DTARGET_IA32E -D
HOST_IA32E -fPIC -DTARGET_LINUX -fabi-version=2 -faligned-new -I../../../source/
include/pin -I../../../source/include/pin/gen -isystem /home/kali/Downloads/pin/
extras/cxx/include -isystem /home/kali/Downloads/pin/extras/crt/include -isystem
 /home/kali/Downloads/pin/extras/crt/include/arch-x86_64 -isystem /home/kali/Dow
nloads/pin/extras/crt/include/kernel/uapi -isystem /home/kali/Downloads/pin/extr
as/crt/include/kernel/uapi/asm-x86 -I../../../extras/components/include -I../../
../extras/xed-intel64/include/xed -I../../../source/tools/Utils -I../../../sourc
e/tools/InstLib -O3 -fomit-frame-pointer -fno-strict-aliasing  -Wno-dangling-poi
nter -c -o obj-intel64/malloctrace.o malloctrace.cpp
g++ -shared -Wl,--hash-style=sysv  ../../../intel64/runtime/pincrt/crtbeginS.o -W
l,-Bsymbolic -Wl,--version-script=../../../source/include/pin/pintool.ver -fabi-
version=2   -o obj-intel64/malloctrace.so obj-intel64/malloctrace.o -L../../../i
ntel64/runtime/pincrt -L../../../intel64/lib -L../../../intel64/lib-ext -L../../
../extras/xed-intel64/lib -lpin -lxed  ../../../intel64/runtime/pincrt/crtendS.o
-lpindwarf -ldwarf -ldl-dynamic -nostdlib -lc++ -lc++abi -lm-dynamic -lc-dynamic
 -lunwind-dynamic
../../../pin   -t obj-intel64/malloctrace.so --  ../../../source/tools/Utils/obj-
intel64/cp-pin.exe makefile obj-intel64/malloctrace.makefile.copy \
  > obj-intel64/malloctrace.out 2>&1
cmp makefile obj-intel64/malloctrace.makefile.copy
rm obj-intel64/malloctrace.makefile.copy
rm obj-intel64/malloctrace.out
```

编写 hello world! 的 c 语言文件，并用 gcc 指令编译成可执行文件。

```
File  Edit  View  Search  Terminal  Help
#include <stdio.h>

void main()
{
printf("hello world!");
}
```

```
kali@kali: ~/Downloads/testCPP
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~/Downloads/testCPP]
└─$ gcc -o FirstCpp FirstCpp.c
```



用刚刚编译生成的动态链接库 malloctrace.so 执行生成的可执行文件 FirstCpp

```
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~/Downloads/pin]
└─$ ./pin -t ./source/tools/ManualExamples/obj-intel64/malloctrace.so -
CPP/FirstCpp
hello world!
```

观察输出结果：

malloc(0x)表示调用了 malloctrace 的 tool 并分配了 0xa 字节的内存空间，其中 a 代表常数；

```
malloc(0×5a1)
  returns 0×7f719713e180
malloc(0×4a1)
  returns 0×7f719713e730
malloc(0×10)
  returns 0×7f719713ebe0
malloc(0×9d)
  returns 0×7f719713ebf0
malloc(0×28)
  returns 0×7f719713ec90
malloc(0×140)
  returns 0×7f719713ecc0
malloc(0×20)
  returns 0×7f719713ee00
```

returns 0xbbbbbbbbbbbb 表示 malloctrace 分配了内存并返回了指向该内存的指针。

```
malloc(0×348)
  returns 0×7f7183cb65e0
malloc(0×90)
  returns 0×7f7183cb6930
malloc(0×410)
  returns 0×7f7183cb69c0
malloc(0×1088)
  returns 0×7f7183cb6dd0
malloc(0×110)
  returns 0×7f7183cb7e60
malloc(0×400)
malloc(0×400)
  returns 0×55a626b2b2a0
```

2. **Inscount0** 的复现

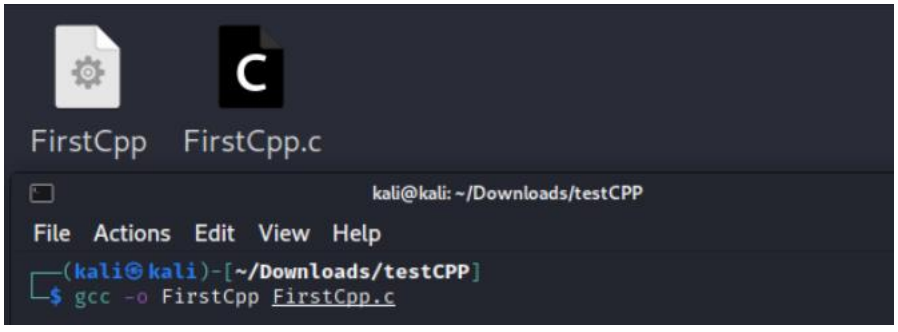
首先从官网 https://software.intel.com/content/www/us/en/develop/articles/pin-abinary-instrumentation-tool-downloads.html 下载 pin（Linux 版本），进行解压后添加到 Linux 虚拟机，在 pin/source/tools/ManualExamples 文件夹中运行终端，编译运行 inscount0 后生成动态链接库 inscount0.so。

```
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~/…/pin/source/tools/ManualExamples]
└─$ make inscount0.test TARGET=intel64
mkdir -p obj-intel64/
g++ -Wall -Werror -Wno-unknown-pragmas -DPIN_CRT=1 -fno-stack-protector -fno-exce
ptions -funwind-tables -fasynchronous-unwind-tables -fno-rtti -DTARGET_IA32E -DHO
ST_IA32E -fPIC -DTARGET_LINUX -fabi-version=2 -faligned-new -I../../../source/inc
lude/pin -I../../../source/include/pin/gen -isystem /home/kali/Downloads/pin/extr
as/cxx/include -isystem /home/kali/Downloads/pin/extras/crt/include -isystem /hom
e/kali/Downloads/pin/extras/crt/include/arch-x86_64 -isystem /home/kali/Downloads
/pin/extras/crt/include/kernel/uapi -isystem /home/kali/Downloads/pin/extras/crt/
include/kernel/uapi/asm-x86 -I../../../extras/components/include -I../../../extra
s/xed-intel64/include/xed -I../../../source/tools/Utils -I../../../source/tools/I
nstLib -O3 -fomit-frame-pointer -fno-strict-aliasing  -Wno-dangling-pointer -c -o
 obj-intel64/inscount0.o inscount0.cpp
g++ -shared -Wl,--hash-style=sysv  ../../../intel64/runtime/pincrt/crtbeginS.o -Wl
,-Bsymbolic -Wl,--version-script=../../../source/include/pin/pintool.ver -fabi-ve
rsion=2   -o obj-intel64/inscount0.so obj-intel64/inscount0.o -L../../../intel64/
runtime/pincrt -L../../../intel64/lib -L../../../intel64/lib-ext -L../../../extra
s/xed-intel64/lib -lpin -lxed  ../../../intel64/runtime/pincrt/crtendS.o -lpindwar
f -ldwarf -ldl-dynamic -nostdlib -lc++ -lc++abi -lm-dynamic -lc-dynamic -lunwind-
dynamic
make -C  ../../../source/tools/Utils dir obj-intel64/cp-pin.exe
make[1]: Entering directory '/home/kali/Downloads/pin/source/tools/Utils'
mkdir -p obj-intel64/
g++ -DTARGET_IA32E -DHOST_IA32E -DFUND_TC_TARGETCPU=FUND_CPU_INTEL64 -DFUND_TC_HO
STCPU=FUND_CPU_INTEL64 -DTARGET_LINUX -DFUND_TC_TARGETOS=FUND_OS_LINUX -DFUND_TC_
HOSTOS=FUND_OS_LINUX -I../../../source/tools/Utils -O3  -std=c++11 -o obj-intel64
/cp-pin.exe cp-pin.cpp -no-pie
make[1]: Leaving directory '/home/kali/Downloads/pin/source/tools/Utils'
../../../pin   -t obj-intel64/inscount0.so --  ../../../source/tools/Utils/obj-int
el64/cp-pin.exe makefile obj-intel64/inscount0.makefile.copy \
  > obj-intel64/inscount0.out 2>&1
cmp makefile obj-intel64/inscount0.makefile.copy
rm obj-intel64/inscount0.makefile.copy
rm obj-intel64/inscount0.out
```

接下来与 **malloctrace 的复现**实验过程相同，编写 hello world! 的 c 语言文件，并用 gcc 指令编译成可执行文件。

```
#include <stdio.h>

void main()
{
printf("hello world!");
}
~
```



用刚刚编译生成的动态链接库 inscount0.so 执行生成的可执行文件 FirstCpp，得到插桩后的

计数结果 count 192211

```
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~/Downloads/pin]
└─$ ./pin -t ./source/tools/ManualExamples/obj-intel64/inscount0.so -- ../testCP
P/FirstCpp
hello world!
```

```
File  Edit  Search  View  Document  Help
Count 192211
```

修改 inscount0.cpp 中的代码，生成新的 tool inscount00.cpp，并编译成动态链接库 inscount00.so。

修改后的 inscount00.cpp ：

```
VOID Instruction(INS ins, VOID *v)
{
 if (INS_Opcode(ins) == XED_ICLASS_MOV &&
 INS_IsMemoryRead(ins) &&
 INS_OperandIsReg(ins, 0) &&
 INS_OperandIsMemory(ins, 1))
 {
 icount++;
 }
}
```

（pin 工具每次遇到一个新指令都会调用该函数）

```
┌──(kali㉿kali)-[~/…/pin/source/tools/ManualExamples]
└─$ make inscount00.test TARGET=intel64                                    2 ×
g++ -Wall -Werror -Wno-unknown-pragmas -DPIN_CRT=1 -fno-stack-protector -fno-exc
eptions -funwind-tables -fasynchronous-unwind-tables -fno-rtti -DTARGET_IA32E -D
HOST_IA32E -fPIC -DTARGET_LINUX -fabi-version=2 -faligned-new -I../../../source/
include/pin -I../../../source/include/pin/gen -isystem /home/kali/Downloads/pin/
extras/cxx/include -isystem /home/kali/Downloads/pin/extras/crt/include -isystem
 /home/kali/Downloads/pin/extras/crt/include/arch-x86_64 -isystem /home/kali/Dow
nloads/pin/extras/crt/include/kernel/uapi -isystem /home/kali/Downloads/pin/extr
as/crt/include/kernel/uapi/asm-x86 -I../../../extras/components/include -I../../
../extras/xed-intel64/include/xed -I../../../source/tools/Utils -I../../../sourc
e/tools/InstLib -O3 -fomit-frame-pointer -fno-strict-aliasing  -Wno-dangling-poi
nter -c -o obj-intel64/inscount00.o inscount00.cpp
g++ -shared -Wl,--hash-style=sysv  ../../../intel64/runtime/pincrt/crtbeginS.o -W
l,-Bsymbolic -Wl,--version-script=../../../source/include/pin/pintool.ver -fabi-
version=2   -o obj-intel64/inscount00.so obj-intel64/inscount00.o -L../../../int
el64/runtime/pincrt -L../../../intel64/lib -L../../../intel64/lib-ext -L../../..
/extras/xed-intel64/lib -lpin -lxed  ../../../intel64/runtime/pincrt/crtendS.o -l
pindwarf -ldwarf -ldl-dynamic -nostdlib -lc++ -lc++abi -lm-dynamic -lc-dynamic -
lunwind-dynamic
../../../pin   -t obj-intel64/inscount00.so --  ../../../source/tools/Utils/obj-i
ntel64/cp-pin.exe makefile obj-intel64/inscount00.makefile.copy \
  > obj-intel64/inscount00.out 2>&1
cmp makefile obj-intel64/inscount00.makefile.copy
rm obj-intel64/inscount00.makefile.copy
rm obj-intel64/inscount00.out
```

执行可执行文件，并得到修改后的插桩计数 1890

```
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~/Downloads/pin]
└─$ ./pin -t ./source/tools/ManualExamples/obj-intel64/inscount00.so -- ../testC
PP/FirstCpp
hello world!
```

```
File  Edit  Search  View  Document  Help
Count 1890
```

（此处根据实际操作过程，留下具体操作步骤、附加一些自己的理解，即可）

心得体会：

通过本次程序插桩及 Hook 实验学习了如何进行使用 pin 工具的动态二进制插桩操作，同时在复现实验和实现自己定义的插桩工具过程中，进一步熟悉 Linux 环境下的指令操作，学习了不同的插桩工具之间的联系和各自的作用。