Information Security Laboratory

08-Debugging

Sang Kil Cha
Dynamic Behavioral Analysis

• Detect polymorphic/metamorphic malware

• Two major types
  – Heuristic-based detection
  – Anomaly-based detection
How to Perform Dynamic Analysis?

• Debugging APIs

• Binary Instrumentation (no source code for malware)
  – Static binary instrumentation
  – Dynamic binary instrumentation (DBI)
What’s Next?

• Anti-debugging
• Anti-instrumentation
Anti- Debugging/Instrumentation

• Benign use: software copy protection

• Malicious use: malware
Software Copy Protection

How would you protect your software?
Example of Copy Protection

Ask a question that only a valid user can answer:
• What is the xth word in page y of the manual?
• What is your serial number that is given at the time you purchased?
Example of Copy Protection (cont’d)

Check if a program is running on a registered device
• IMEI of a smartphone
• IP address, Mac address, user ID, etc.
Example of Copy Protection (cont’d)

A phone-based activation
• Only a registered phone number can be used
• You will not share your license (or serial) with many people
Altering Software?

You can easily bypass all such protections by simply modifying the program executables.

```c
// ...
if ( phone_activation() == SUCCESS )
    return VALID_USER;
// ...
```

Typically one-byte change in binary
Software Cracking

• Remove or disable features
  − Copy protection routines
  − Advertisement

• Reversing is crucial: no source code for COTS software

This is Illegal!
Anti Debugging
How Does Debugger Work?

• OS provides debugging API functions
  – Linux: ptrace system call
  – Windows: WaitForDebugEvent, ContinueDebugEvent, etc.

• OS passes the control to debugger
Introduction to PTRACE

#include <sys/ptrace.h>

long ptrace(enum __ptrace_request request, pid_t pid, void *addr, void *data);

man ptrace
Introduction to PTRACE

Debugee process

```c
ptrace(PTRACE_TRACEME, 0, 0, 0);
execve("/bin/ls", args /* arguments */ , 0);
```
Introduction to PTRACE

Debugger process

```c
int status;
waitpid(pid, &status, 0);
while (WIFSTOPPED(status)) {
    ptrace(PTRACE_SINGLESTEP, pid, 0, 0);
    // Do something
    waitpid(pid, &status, 0);
}
```
Writing an Instruction Tracer

Count every instruction executed in a target process
Breakpoints?

• Software breakpoints
  – int3 instruction (0xcc) replacement
  – Unlimited

• Hardware breakpoints
  – DR registers on x86
  – Limited to 4 (on x86)
Software Breakpoint

4004d6:  55  push  rbp
4004d7:  48 89  e5  mov  rbp,rsp
4004da:  b8 00 00 00 00  mov  eax,0x0
4004df:  5d  pop  rbp
4004e0:  c3  ret
Software Breakpoint

1. SIGTRAP at 4004da
2. Replace the byte at 4004da with the original byte (b8)
3. Modify the program counter (EIP/RIP)
4. Resume
Anti-Debugging (1)

```c
if (ptrace(PTRACE_TRACEME, 0, 0, 0) < 0) {
    return 1;
}
```
Anti-Debugging (2)

/proc/$PPID/status

Check the parent’s name!
Anti-Debugging (3)

```c
signal(SIGTRAP, handler); // Implicit control flow
```
memchr(code, 0xcc, size);
Debugger without PTRACE?

• Emulator-based debugging

• Instrumentation-based
Red Pill and Blue Pill

Image from https://en.wikipedia.org/wiki/Red_pill_and_blue_pill
Red Pill = Detect Virtualization

- `/proc/ide/hd*/model`
- `dmidecode`
- Timing channel
- Etc.
Question?