Lecture 10

Anti-debugger techniques

Anti-debuggers

- Making reverse-engineering and disassembly painful
  - Polymorphism
  - Encryption
  - Interrupt disabling
  - Debugger detection
    - Behavior modification
    - Crashing debugger
- Links
  - http://www.textfiles.com/virus/adebgtut.txt
  - http://anti-debugging.qarchive.org
  - http://rdist.root.org/2007/04/19/anti-debugger-techniques-are-overrated
  - http://www.openrce.org/reference_library/anti_reversing
  - http://reconstructer.org

Modify Interrupts

- Overwrite Interrupt Vector of INT 1/3 to point to garbage code
  - INT 1 = Debug Single Step
  - INT 3 = Debug Breakpoint
  - Point to garbage so debugger crashes
  - Must skip instructions to counter
- Run long loop of INT 3 instructions
  - Slows down AV or debugging process
  - Must NOP out INT 3 to counter
- Turn off keyboard interrupts

```
IN AL,20h
OR AL,02
OUT AL,20
<virus code>
IN AL,20
AND AL, NOT 2
OUT AL,20
```

Modify interrupts

- Replace INT 1/3 with another valid interrupt
- Detect INT 1 tracing by stack inspection
  - INT 1 overwrites 6 bytes below SP with return values for IP, CS, and Flags

```
PUSH AX
POP AX
DEC SP
DEC SP
POP BX
CMP AX,BX
JNE CODE_IS_TRACED
```

- Force INT 1/INT 3 tracing to disable code
  - Hide critical value (i.e. decryption key) on stack directly without modifying stack pointer
  - Debugger will overwrite value if it runs

Protecting anti-debug code

- Use anti-debug routines as decryption key
  - Retrieve a byte from routines to modify decryption routine
  - Prevents AV and anti-debugger from “NOP”ing out your traps
- Running line
  - Hook INT 1
  - Decrypt each instruction just before it is run
  - Re-encrypt each instruction just after it is run
  - Only one instruction at a time is decrypted in memory

Detection tricks

- `IsDebuggerPresent()`
  - Returns 0 if current process is not running in the context of a debugger
  - Searches PEB (Process Environment Block) for the field `IsDebugged`
  - `CheckRemoteDebuggerPresent()` checks if debugger is a separate, parallel process that is attached to process
  - Windows NT
    - NTGlobalFlag for WindowsNT
    - NtQueryInformationProcess
  - Can be easily bypassed with breakpoint scripts, patching of the IAT, or directly modifying the PEB

Detection tricks

- Detect presence of INT 41/INT 2dh
  - Used by Windows debugging interface
- Detect INT 68h hooking (SoftICE)
- Detect files and registry keys created by debugger
  - szOllyKey, szIsSICEKey
  - Attempt to create SoftICE driver to detect presence
  - Look for \system32\drivers\WINICE.dat
  - \\.\SICE, \\.\SIWDEBUG, etc.
- Detect debug processes or hooks
  - Check for NTice service via OpenService()
  - OllyInvisible hooking of CsrGetProcessId
  - Olly HideDebugger detour modifications to OpenProcess()
  - Scan every running process and read out memory address 0x004B064B (where OllyDB stores some of its strings)

Detection tricks

- Scan and detect 0xCC opcode (int 3)
- Scan low memory for specific names installed by SoftICE such as “WINICE.BR”, “SOFTICE1”
- Examine timing
  - Use rdtsc to measure cycles
  - Measure system calls to detect virtualization or syscall hijacks
    - Vmware, Sebek, UML, etc.
    - VMEDetect using rdtsc trick
  - Debugger instruction stepping
  - Obfuscate calls to rdtsc by using trampoline within existing ntdll.dll (ReleaseBufferLocation())
- Detecting VMware
  - Execute VMware backdoor I/O function call
  - Check location of IDT

Detection tricks

- Check registers and flags used for debug purposes
  - TRAP bit in EFLAGS register
  - DR0-DR7
    - Change registers in exception handler
    - Set handler apriori and cause an error in code (divide by zero)
    - When context switching to handler, debug registers saved on user stack
    - Read and write these values
  - PEB ProcessHeap flag
- Execute exception with Trap flag set
  - No debugger = SEH will occur
  - Debugger = SEH will not occur
- Check PROCESSINFOCLASS for ProcessDebugPort or SYSTEMINFOCLASS for SystemDebuggerInformation

Crashing tricks (landmines)

- Exploit software errors in debuggers
  - Confuse LordPE and ProcDump dumping by reporting inconsistent process image size
  - Crash SoftICE with poorly formatted strings
  - Crash SoftICE with illegal form of instructions
  - Crash OllyDBG with format string vulnerability
    - OutputDebugString()
  - Crash OllyDBG with bogus PE Headers

Confusion tricks

- OllyDBG does not handle instruction prefixes well
  - Insert prefix one byte before SEH
  - OllyDBG skips it, but SEH runs with no debugger
- OllyDBG mishandling INT 3
  - Set an SEH before INT3
- OllyDBG memory handling
  - PAGE_GUARD used as memory break-point
  - Set SEH and execute PAGE_GUARD exception
    - With OllyDBG, goes to MemBpx and continues execution
    - Without OllyDBG, SEH code executed