



Reconstructing the Scene of the Crime

METASPLOIT AUTOPSY



Who are they?

STEVE DAVIS

- Security Consultant /
Researcher
at MANDIANT

PETER SILBERMAN

- Engineer / Researcher
at MANDIANT

Agenda

- 1/2 Demo
 - Pop it like its *hotttt*
- Problem / Solution
- Process Acquisition
- Metasploit
- Meterpreter Communication
- Metasploit Forensic Framework (MSFF)
- 1/2 Demo
 - Reconstructing it like its *hotttt*

Demo Part 1

- Box Windows XP Fresh SP3

*Same box that our slides are running from...
Oh noes!*

- MS08-067 meterpreter bind tcp

5

Back to our regularly scheduled slides...

Problem

- Meterpreter
 - Traditional disk forensics is helpless
 - Attack vector may never touch disk
 - No way to determine what happened
- Goal
 - Reconstruct attacker's Meterpreter sessions with as much reliability as possible

Solution

- Acquire exploited processes' address space
- Parse out meterpreter protocol from acquired memory sections
 - Reconstruct meterpreter sessions

MANDIANT Memoryze

ENUMERATION

- All running processes
 - Handle table
 - Memory sections
 - Ports
 - Strings
- Drivers
 - Including layered ones
- Certain kernel hooks

ACQUISITION

- Physical memory image
- Running process's memory space
 - Binary
 - Loaded DLL's
 - Stacks
 - Heaps
 - Data sections
- Drivers

MANDIANT Memoryze

- Can analyze memory live, or from image
 - Live analysis can use paging file for a more complete picture of memory
- Supported platforms
 - 32-bit Windows 2000, XP, 2003 Server
 - Beta support for Vista
- Download at
 - <http://www.mandiant.com/>

10

Process Acquisition

Why Process Acquisition?

- Acquisition was originally used mostly for malware analysis
 - Acquire packed binaries running in memory
 - Usually utilized debuggers
 - Can defeat most packers
- Acquisition has other uses:
 - Acquire unknown binaries for Virustotal
 - Acquire memory to look for protocol strings
 - Encrypted strings are unencrypted in memory

Classic Process Acquisition

- Current Methodology
 - Open handle to process, OR
 - Attach to process
 - `ReadProcessMemory(hProc, ImageBase, buffer, ImageSize, BytesRead)`
- Current drawbacks
 - Requires “touching” a process
 - Detecting debuggers is trivial
 - Gives an incomplete picture of memory

Process Acquisition: Memoryze

RELIES ON

- Physical memory access
- Virtual to physical address translation

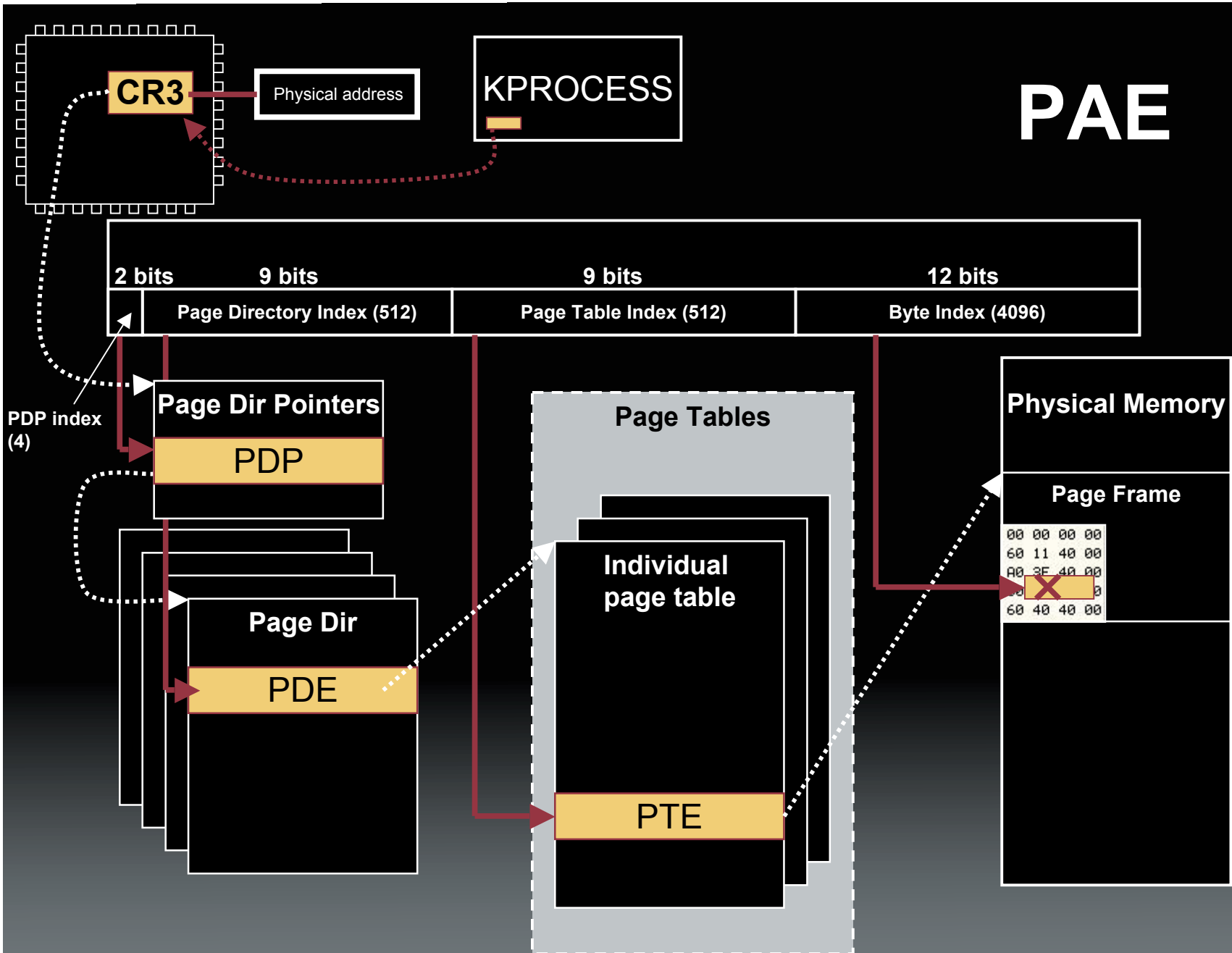
DOES NOT RELY ON

- Attaching to a process with a debugger
- Opening handles to processes or threads
- API calls
- The OS's Virtual Memory Manager

Memoryze: Process Acquisition

- Accessing Physical Memory
 - Live analysis
 - Acquisition
- `\Device\PhysicalMemory`
 - Section object exposed by Windows
 - Reading from handle allows application to read physical memory
 - Every virtual address must be translated to a physical offset within the section object

PAE



Memoryze: Process Acquisition

- Map physical memory into buffer
- Acquisition:
 - Write buffer to disk (dd)
- Analysis:
 - Scan buffer for known signatures of kernel structures, e.g. EPROCESS

New Process Acquisition

- Find all processes (EPROCESS) in physical memory
 - VadRoot within the EPROCESS structure
 - The VadRoot is the top node of a tree of Memory Manager Virtual Address Descriptor (MMVAD) entries
 - MMVAD entries contain the virtual start address and size of each memory section within a process
 - MMVAD entries containing mapped DLL's or EXE's will have a pointer to the path of the binary
 - Helps manage process' virtual address space

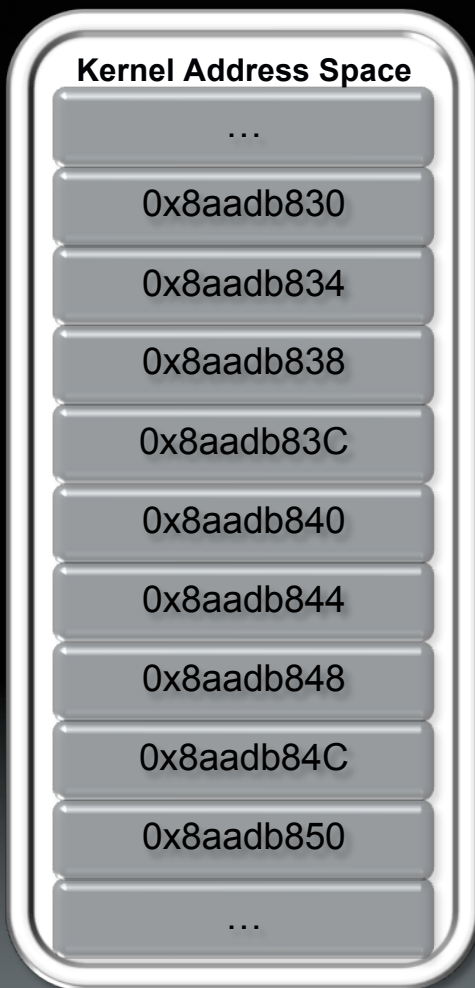
Memoryze: Process Acquisition

- OllyDbg's memory map view shows the different sections

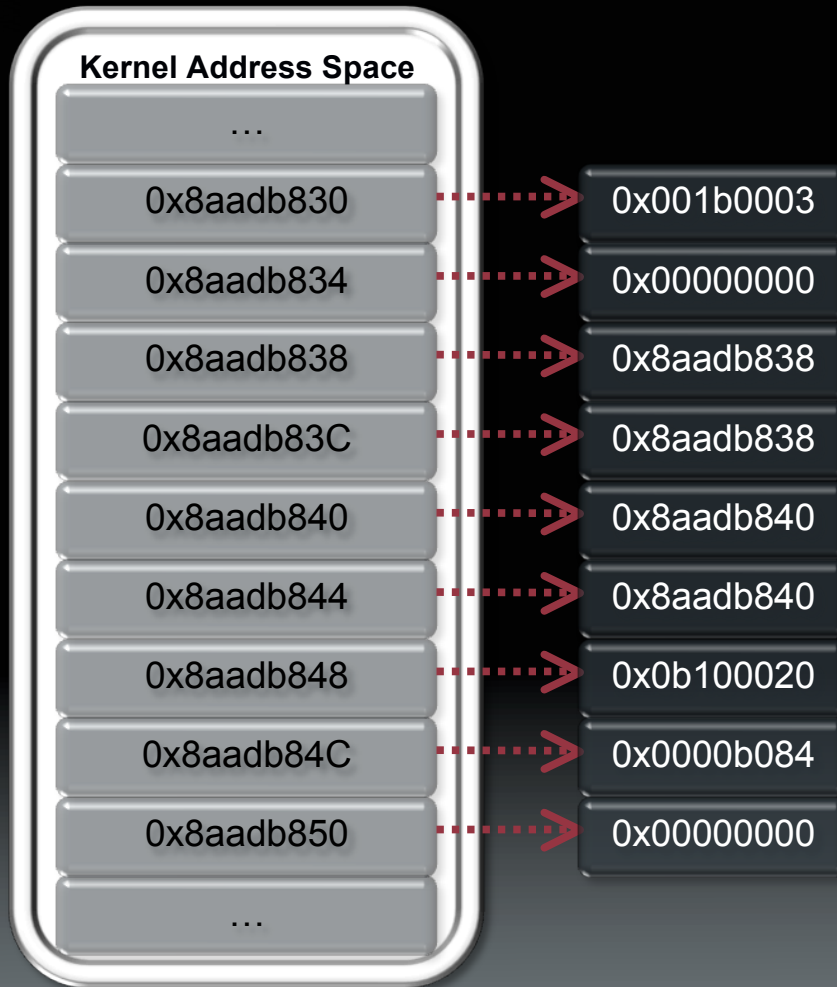
Address	Size	Owner	Section	Contains	Type	Access	Initial
00010000	00001000				Priv	Rw	Rw
00020000	00001000				Priv	Rw	Rw
00030000	00001000				Priv	Rw	Rw
0007B000	00001000				Priv	Rw	Rw
0007C000	00004000			stack of ma	Priv	Rw	Rw
00080000	00003000				Map	R	R
00090000	00002000				Map	R	R
000A0000	00010000				Priv	Rw	Rw
001A0000	00006000				Priv	Rw	Rw
001B0000	00003000				Map	Rw	Rw

- Each address range is an entry in VadRoot, represented by a MMVAD structure
- Enumeration of VadRoot allows access to heaps, stacks, and binary images

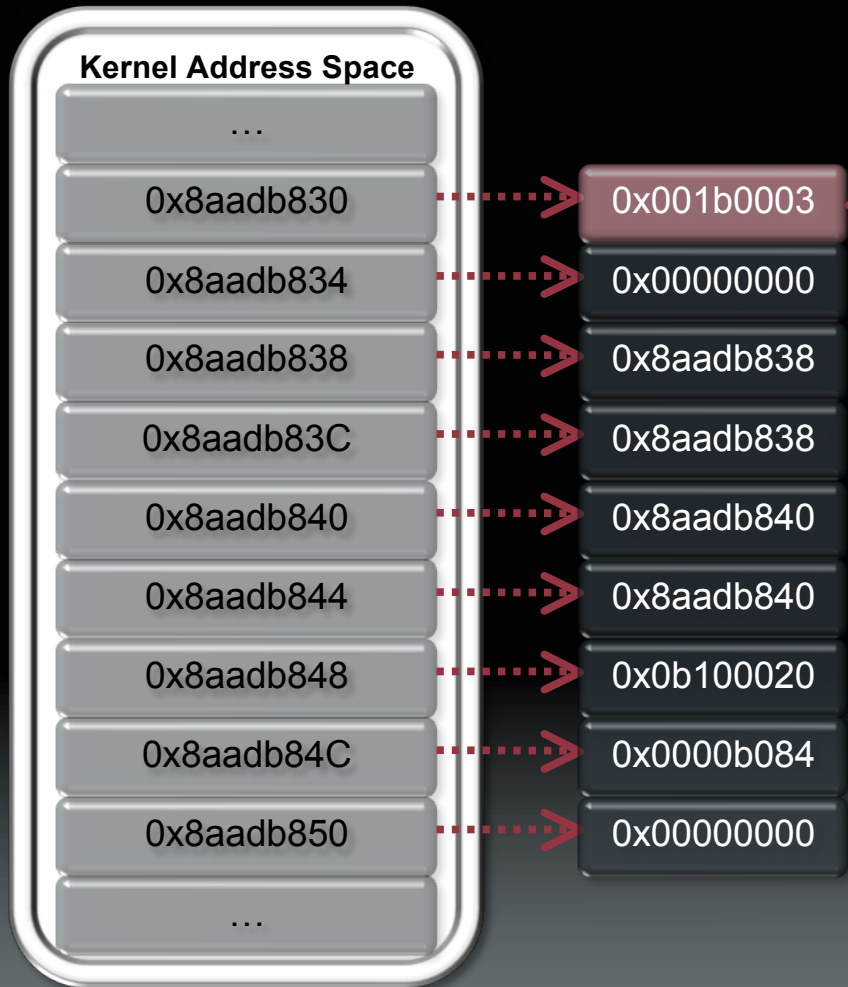
Finding Processes



Finding Processes

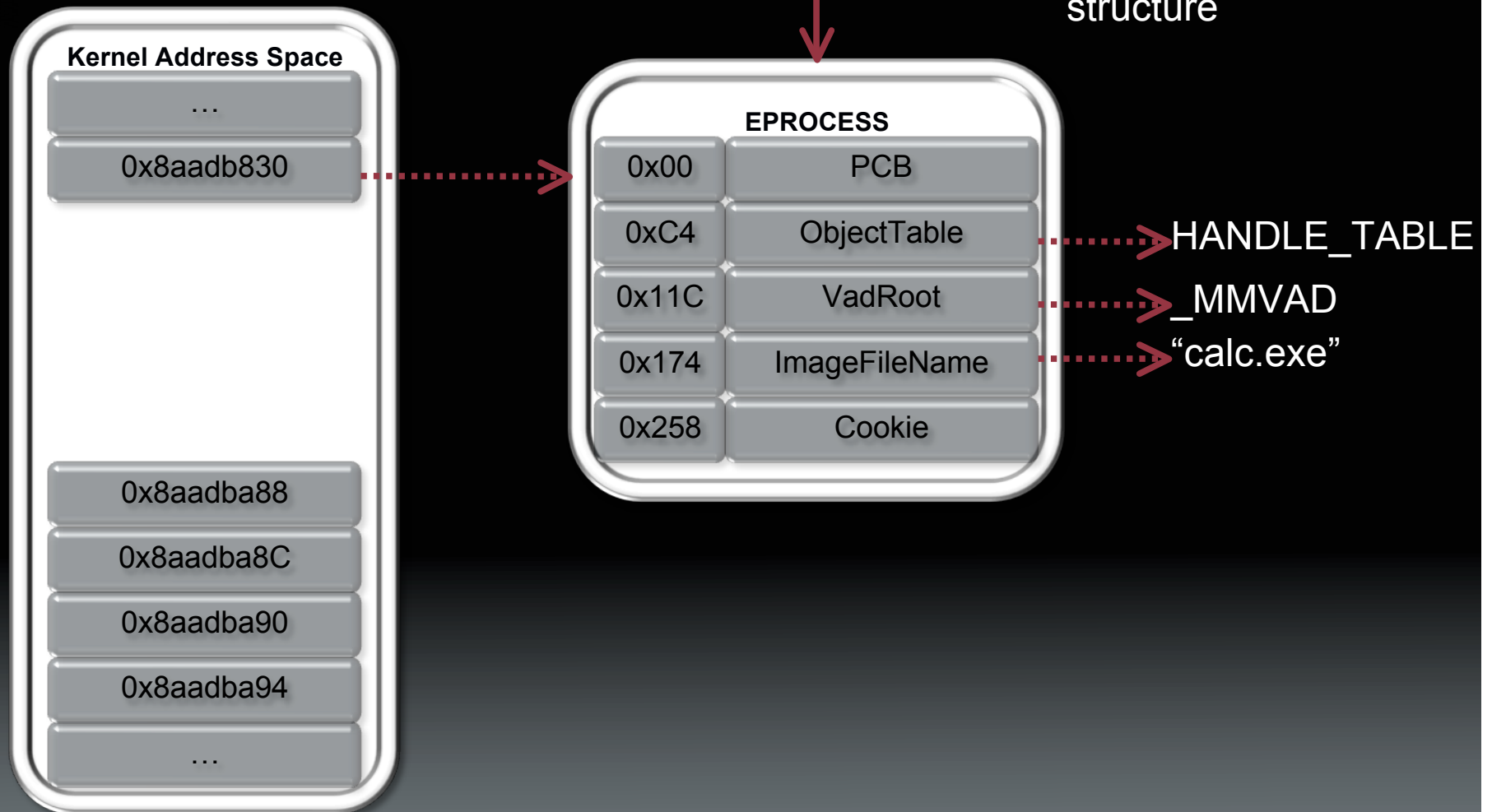


Finding Processes

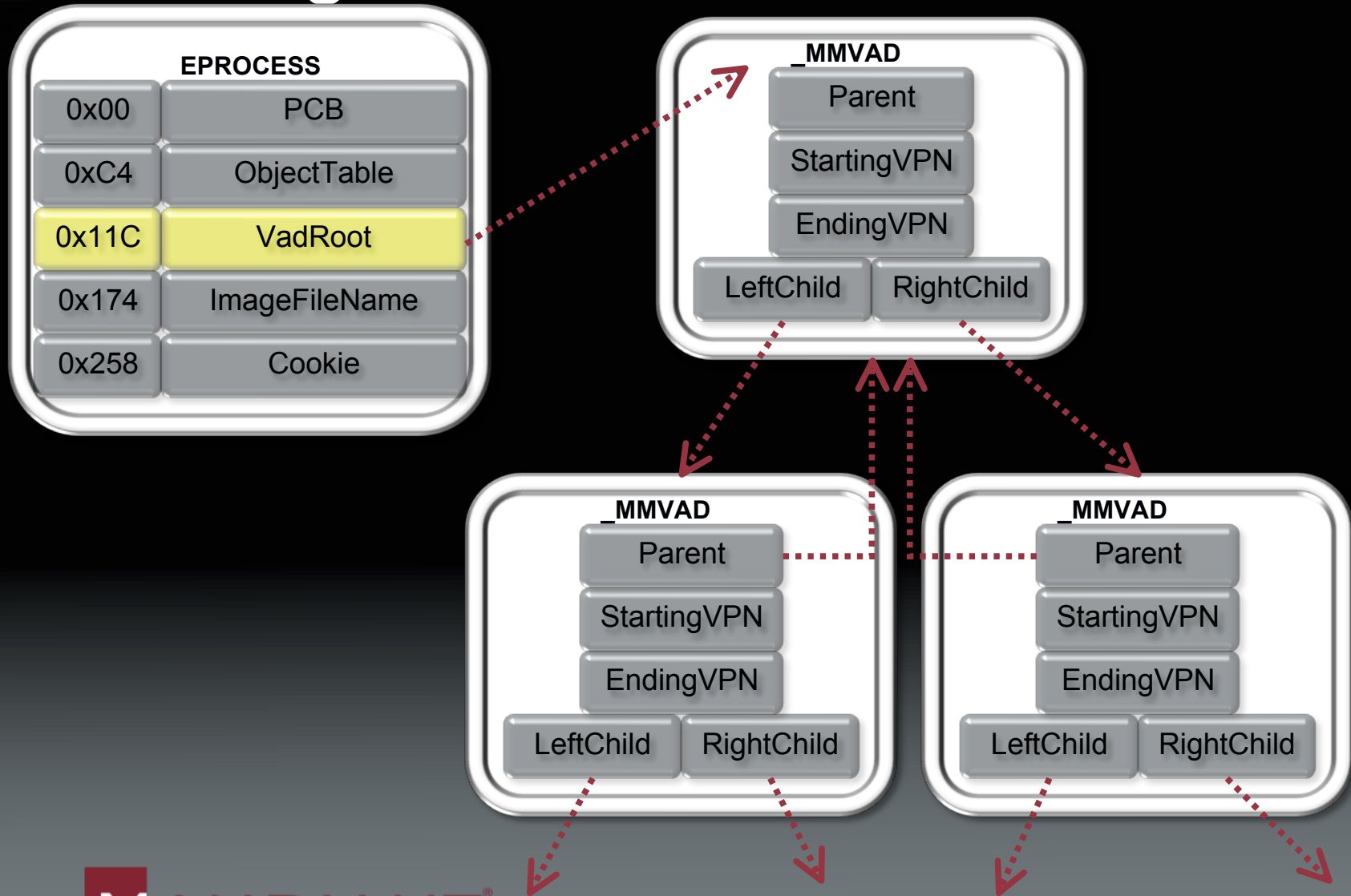


Indicates EPROCESS,
DISPATCH_HEADER,
further checks are
needed

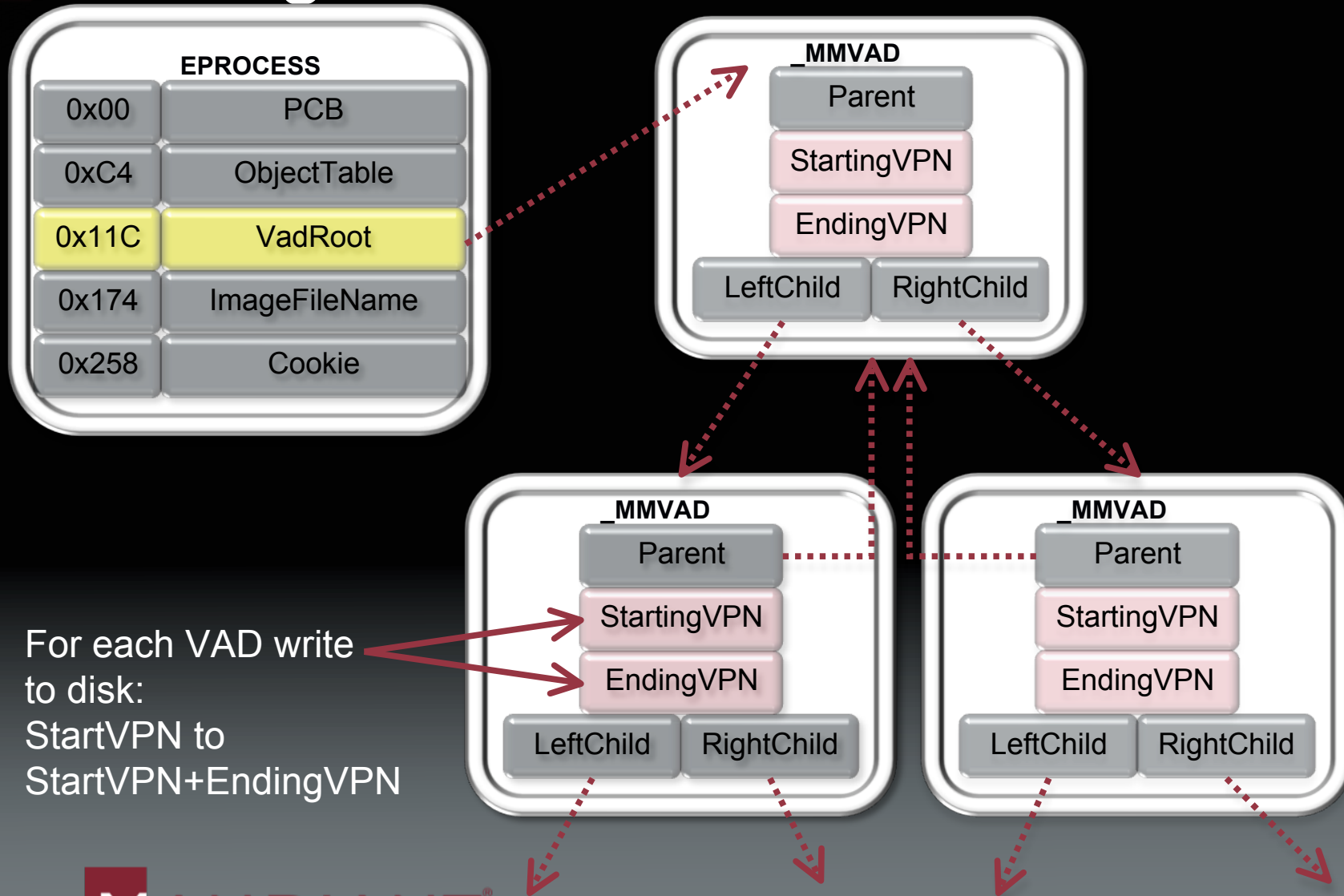
Finding Processes

























































































Parsing MMVAD



Writing VADs to disk



 976_%5cWINDOWS%5csystem32%5cws2_32.dll	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01a80
 976_%5cWINDOWS%5csystem32%5cws2help.dll	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01a90
 976_%5cWINDOWS%5csystem32%5cwshtcpip.dll	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01aa0
 976_%5cWINDOWS%5csystem32%5cwssock32.dll	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01ac0
 976_%5cWINDOWS%5csystem32%5cwsapi32.dll	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01ad0
 976_%5cWINDOWS%5csystem32%5cwuaueng.dll	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01ae0
 976_%5cWINDOWS%5csystem32%5cwuauuser.dll	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01af0
 976_%5cWINDOWS%5csystem32%5cwups2.dll	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01b00
 976_%5cWINDOWS%5csystem32%5cwups.dll	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01b10
 976_%5cWINDOWS%5csystem32%5cwzcsvc.dll	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01b20
 976_%5cWINDOWS%5csystem32%5cactsrv.dll	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01b30
 976_%5cWINDOWS%5cWinSxS%5cx86_Microsoft.Windows.Common-Controls_6595b64144cf1df_6.0.10.0_x-ww_f7fb5805%5ccomctl32.dll	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01bb0
 BatchResult.xml	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01cd0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00a0000-0x00a3ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01cf0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00a40000-0x00a7ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01d00
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00a80000-0x00a81fff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01d40
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00a90000-0x00a95fff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01e40
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00aa0000-0x00aaffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01e80
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00ab0000-0x00ab0fff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x01ec0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00ac0000-0x00ac1fff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x03cf00
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00ad0000-0x00ad0fff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x03df00
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00b10000-0x00b1ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x03f500
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00b20000-0x00b21fff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x004d0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00b30000-0x00b3ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x004e0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00c30000-0x00c3ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x005a0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00cb0000-0x00ceffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x005e0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00cf0000-0x00d2ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x006b0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00d30000-0x00d3ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x006f00
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00d40000-0x00d41fff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x7ff600
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00d50000-0x00e4ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x7ff7b0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00e50000-0x00e8ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x7ff7c0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00e90000-0x00ecffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x7ff7d0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00ed0000-0x00f0ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x7ff7e0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00f10000-0x00f4ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x7ff7f0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00f50000-0x00f8ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x7ff8a0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00f90000-0x00fcffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x7ff8c0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00fd0000-0x00fe7fff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x7ff8d0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x00ff0000-0x00ff0fff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x7ff8e0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x001a0000-0x001b5fff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x7ff9a0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x01a10000-0x01a4ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x7ff9b0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x01a50000-0x01a5ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x7ff9c0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x01a60000-0x01a6ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x7ff9d0
 C%3a%5cWINDOWS%5cSystem32%5c976_0x01a70000-0x01a7ffff.VAD	 C%3a%5cWINDOWS%5cSystem32%5c976_0x7ff9e0

New Process Acquisition

- Allows dumping of full address space
- Overcomes most binary packing
- Captures communication protocol strings
- Bypasses any anti-debugging techniques
- Acquire(s):
 - DLL's that are only in memory
 - Code corresponding to injected threads or shellcode

Metasploit

Have YOU read the developer docs?

Metasploit

- Open source exploit framework originally developed in Perl (1.x, 2.x) by HD Moore et al.
 - Currently Ruby (3.x)
- Platform independent
- Multiple payloads

Meterpreter

- The next generation of post-exploitation payloads
 - Forget `/bin/sh` and `cmd.exe`
 - Limited to `stdin, stderr, stdout`
 - Non-interactive
- Full functioning client → server interpreter
 - File upload / download
 - Key logging
 - Simple extension addition
- Can be completely memory resident

Under the Meterpreter Hood

- DLL gets injected into exploited process
- Hooks LoadLibrary (on Windows)
 - Applies hook to Win32 API LoadLibrary
 - Changes lower level API's behavior to allow LoadLibrary to load a DLL from memory
- Hooked API's to allow loading of `met_srv.dll` from memory
 - `NtOpenSection`, `NtCreateSection`
 - `NtQueryAttributesFile`
 - `NtOpenFile`, `NtMapViewOfSection`

Meterpreter Communication

- TLV (really LTV) Structures
 - Provide communication protocol for meterpreter server and client
 - 32 bit Length and Type Fields
 - n bits Value Field

Meterpreter Communication



Attacker

Sends Exploit

Payload Meterpreter bind_tcp



Victim

Meterpreter Communication



Attacker

Attacker executes "getpid"



Victim

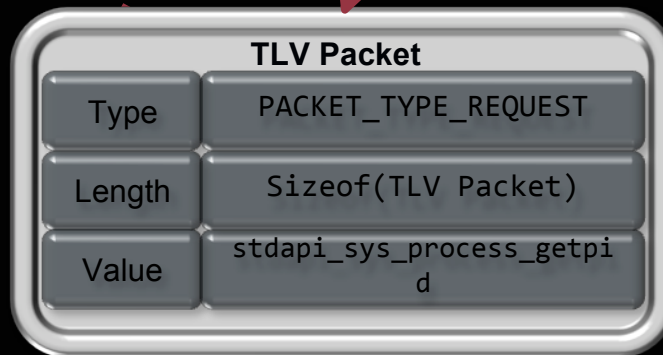
Meterpreter

Meterpreter Communication



Attacker

Request sent when attacker executes getpid



Victim

Meterpreter

Meterpreter Communication



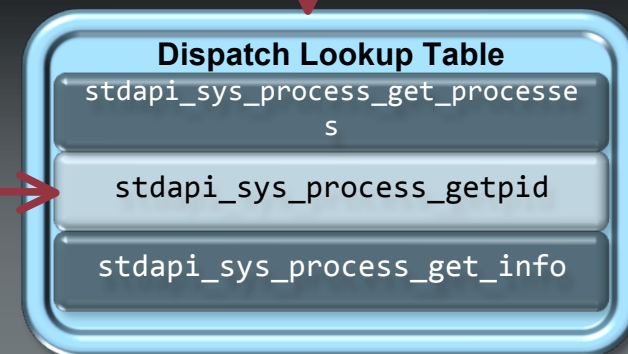
Attacker



Victim

Meterpreter does an internal lookup for the method requested:
`stdapi_sys_process_getpid`

Meterpreter



Meterpreter Communication



Attacker



Victim



Meterpreter builds a response on the heap;
response includes the result of
GetCurrentProcessId



Meterpreter Communication



Attacker

Response is sent back to the attacker



Victim

Response

Meterpreter

Response

Meterpreter Communication



Attacker



Victim

Response

Meterpreter

Response packet is freed
by meterpreter

Response

Response Packet Structure (1 of 4)

4)

Response Packet		
Length	sizeof(Response Packet)	
Type	PACKET_TLV_TYPE_PLAIN_RESPONSE	
Value	Length	sizeof(this tlv)
	Type	TLV_TYPE_METHOD
	Value	stdapi_sys_process_getpid

Response Packet Structure (2 of 4)

4)

Response Packet							
Length	sizeof(Response Packet)						
Type	PACKET_TLV_TYPE_PLAIN_RESPONSE						
Value	<table border="1"><tbody><tr><td>Length</td><td>sizeof(this tlv)</td></tr><tr><td>Type</td><td>TLV_TYPE_REQUEST_ID</td></tr><tr><td>Value</td><td>3164813846702899128916537536399</td></tr></tbody></table>	Length	sizeof(this tlv)	Type	TLV_TYPE_REQUEST_ID	Value	3164813846702899128916537536399
Length	sizeof(this tlv)						
Type	TLV_TYPE_REQUEST_ID						
Value	3164813846702899128916537536399						

Response Packet Structure (3 of 4)

4)

Response Packet							
Length	sizeof(Response Packet)						
Type	PACKET_TLV_TYPE_PLAIN_RESPONSE						
Value	<table border="1"><tbody><tr><td>Length</td><td>sizeof(this tlv)</td></tr><tr><td>Type</td><td>TLV_TYPE_PID</td></tr><tr><td>Value</td><td>0x000003EC</td></tr></tbody></table>	Length	sizeof(this tlv)	Type	TLV_TYPE_PID	Value	0x000003EC
Length	sizeof(this tlv)						
Type	TLV_TYPE_PID						
Value	0x000003EC						

Response Packet Structure (4 of 4)

4)

Response Packet	
Length	sizeof(Response Packet)
Type	PACKET_TLV_TYPE_PLAIN_RESPONSE
Value	
Length	sizeof(this tlv)
Type	TLV_TYPE_RESULT
Value	0x00000000

Response Packet Structure

Response Packet		
Length	sizeof(Response Packet)	
Type	PACKET_TLV_TYPE_PLAIN_RESPONSE	
Value	Length	sizeof(this tlv)
	Type	TLV_TYPE_METHOD
	Value	stdapi_sys_process_getpid
	Length	sizeof(this tlv)
	Type	TLV_TYPE_REQUEST_ID
	Value	3164813846702899128916537536399
	Length	sizeof(this tlv)
	Type	TLV_TYPE_PID
	Value	0x000003EC
	Length	sizeof(this tlv)
	Type	TLV_TYPE_RESULT
	Value	0x00000000

Response Packet from Memory

```

08 74 04 06 00 01 00 01 73 74 64 61 70 69 5F 73 ; .t.....stdapi_s
79 73 5F 70 72 6F 63 65 73 73 5F 67 65 74 70 69 ; ys_process_getpi
64 00 00 00 00 29 00 01 00 02 33 31 36 34 38 31 ; d,...)....316481
33 38 34 36 37 30 32 38 39 39 31 32 38 39 31 36 ; 3846702899128916
35 33 37 35 33 36 33 39 39 34 00 00 00 00 0C 00 ; 5375363994.....
02 08 FC 00 00 03 EC 00 00 00 0C 00 02 00 04 00 ; ..ü...i.....
00 00 00 01 48 05 98 01 0B 00 0E 00 C7 01 0E 00 ; ....H.".....Ç...

```

TLV Packet

Length	<i>Doesn't exist do to free()</i>	
Type:	TLV_TYPE_METHOD	0x00010001
Value:	stdapi_sys_process_getpid	

Response Packet from Memory

```

08 74 04 06 00 01 00 01 73 74 64 61 70 69 5F 73 ; .t.....stdapi_s
79 73 5F 70 72 6F 63 65 73 73 5F 67 65 74 70 69 ; ys_process getpi
64 00 00 00 00 29 00 01 00 02 33 31 36 34 38 31 ; d....)....316481
33 38 34 36 37 30 32 38 39 39 31 32 38 39 31 36 ; 3846702899128916
35 33 37 35 33 36 33 39 39 34 00 00 00 00 0C 00 ; 5375363994 .....
02 08 FC 00 00 03 EC 00 00 00 0C 00 02 00 04 00 ; ..ü...i.....
00 00 00 01 48 05 98 01 0B 00 0E 00 C7 01 0E 00 ; ....H.".....Ç...

```

TLV Packet	
Length	0x29
Type:	TLV_TYPE_REQUEST_ID 0x00010002
Value:	3164813846702899128916537536399

Response Packet from Memory

```

08 74 04 06 00 01 00 01 73 74 64 61 70 69 5F 73 ; .t.....stdapi_s
79 73 5F 70 72 6F 63 65 73 73 5F 67 65 74 70 69 ; ys_process_getpi
64 00 00 00 00 29 00 01 00 02 33 31 36 34 38 31 ; d....)....316481
33 38 34 36 37 30 32 38 39 39 31 32 38 39 31 36 ; 3846702899128916
35 33 37 35 33 36 33 39 39 34 00 00 00 00 0C 00 ; 5375363994.....
02 08 FC 00 00 03 EC 00 00 00 0C 00 02 00 04 00 ; ..ü...i.....
00 00 00 01 48 05 98 01 0B 00 0E 00 C7 01 0E 00 ; ....H.".....Ç...

```

TLV Packet		
Length	0x0C	
Type:	TLV_TYPE_PID	0x000208FC
Value:	0x000003EC	

Response Packet from Memory

```

08 74 04 06 00 01 00 01 73 74 64 61 70 69 5F 73 ; .t.....stdapi_s
79 73 5F 70 72 6F 63 65 73 73 5F 67 65 74 70 69 ; ys_process_getpi
64 00 00 00 00 29 00 01 00 02 33 31 36 34 38 31 ; d....)....316481
33 38 34 36 37 30 32 38 39 39 31 32 38 39 31 36 ; 3846702899128916
35 33 37 35 33 36 33 39 39 34 00 00 00 00 0C 00 ; 5375363994.....
02 08 FC 00 00 03 EC 00 00 00 0C 00 02 00 04 00 ; ..ü...i.....
00 00 00 01 48 05 98 01 0B 00 0E 00 C7 01 0E 00 ; ....H.".....Ç...

```

TLV Packet		
Length	0x0C	
Type:	TLV_TYPE_RESULT	0x00020004
Value:	0x00000000	

Meterpreter Communication

- The response packet is freed by meterpreter
- However...
- When Windows' memory manager frees memory, it is not *immediately* reused.
 - It can take hours for memory to be reclaimed after it has been freed.

49

Metasploit Forensic Framework

Finding one pwned system at a time

Metasploit Forensic Framework

- Scan acquired VADs looking for:
 - Strings containing meterpreter methods
 - This indicates a TLV response to a specific method
 - Parsing out the response TLV gives analysts the data attackers received
 - Also indicates what commands were executed on the machine

Conclusion

- Windows memory manager gives analysts a chance to see artifact memory
- Large impact for forensics
 - Not so large on Metasploit project
- Combining memory analysis with further research will lead to better and more effective projects

DEMO

Demo Part 3

- Acquire `svchost.exe`
 - *Remember attacker terminated connection roughly 30 minutes ago*
 - Run Metasploit Forensic Framework (`msff`)

Questions???

- stephen.davis@mandiant.com
- peter.silberman@mandiant.com