CIS 5930/4930
Offensive Computer Security
Spring 2014

What is an Incident?

- Refers to a security breach or attack
  - DoS
  - Data leaks
    - Confidential
    - PII
    - IP
    - Secret
  - Sabotage
    - data corruption
  - Malware

What is Incident Response?

- Is an organized approach to addressing and remediating the aftermath of a security breach / attack.

The Goal(s):

- To limit the damage of the incident
- To limit the recovery time
- To limit the costs incurred by the incident

Common Challenges:

- Budgets, resources, limited personnel
- Bureaucracy, Share/Stakeholders

Incident Responder Roles

The following roles must be part of an effective IR team:

Balance between responsibility and authority is key

- **Incident Coordinator**
  - Keep track of everything, address expectations, understands bureaucracy, understands laws/regs
- **Incident Manager**
  - someone with strong social skills, knows bosses, SME's
- **Incident Responders**
  - capable, well-informed, and technically skilled
- **Subject Matter Experts (SME's)**
  - perhaps consultants (usually IR team budgets cannot afford SME's as full time)
- **Zeus (Ultimate Authority)**
  - You need someone who can move the bureaucratic mountains and oceans - may be an executive / stakeholder
    - *"Why you ask? Because we just got hacked, do what I say, or else"*

Keys to efficient incident response

- Clear leadership
  - clear division of responsibilities & authorities
  - Established plan & processes
  - keep morale up, always learning from mistakes
  - Address stakeholder's expectations, and keep them informed
- Incident responders who ask good questions
- Plan is not public

Dealing with stakeholder's effectively

Bosses / Stakeholders will be impatient.

- Not understand the situation
- Can make things worse
- Can slow things down
  - Proper incident response != profit to them

Nuances of Incident Response

- Record keeping is key
  - **NOT STORED ON EXTERNAL SYSTEMS**
    GOOGLE DOCS or public documents
    - Usually attackers are targeting PII, or stuff that shouldn't be anywhere OUTSIDE of your networks
  - Need-to-know basis, Sensitive information
- Record the mistakes the team made
  - not to be used against the team, but to learn from
  - mistakes are common
- Lots of 4am decisions + coffee
- **Usually law enforcement gets involved + chain of**

Incident Response Phases

0. Preparation
   - Establishing a IR plan / team (very complicated)
1. Triage
   - Identify the mortally wounded systems, focus on the ones you can save
2. Containment
   - Scramble to understand the problem, communicate (quickly) what is known
   - Goal is to get to a point where the incident is no longer a direct threat
     i. limit the scope of the incident
     ii. stop the bleeding / infection

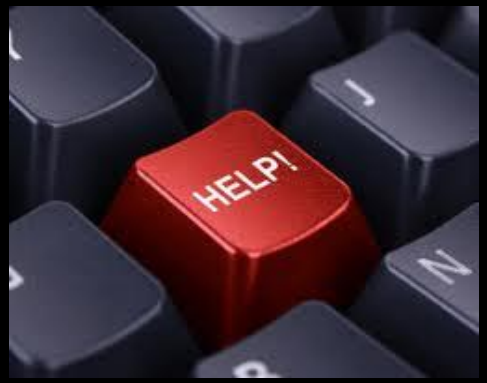
Incident Response Phases

3. Response
   - Fix the problems (easier said than done)
4. Resolution
   - root cause analysis
     i. root cause may be deep-seeded in organization, may be political & beyond scope of IR team
   - IR report
   - **Aftermath:**
     i. Someone may get fired, goto jail, or get demoted
     ii. Usually few details are disclosed

When do you respond?

How are incidents identified?

- What's suitable for just an IT ticket
- What's suitable for a full fledged incident response?
- False positives / False negatives
- No perfect way



Indicators of Compromise

Indicator of Compromise = is a forensic artifact or remnant of an intrusion that can be identified on a host or network.

- *Used to communicate threat intelligence among defenders*
- Depends on attacker
  - Insider threat
  - Outside hacker
    - or hybrid
- Depends on attack vector
  - over the network
  - malicious USBs

Indicators of Compromise

Straightforward indicators:

- Anonymous dumps your corporate emails
- Your corporate secrets are on Wikileaks
- Audit reveals $$$$$ is missing

Not-so-straightforward indicators:

- You find out from the news
  - Whistle blower?
    - legitimacy? Imposter?
  - Leak?
    - of future product plans / IP
    - of mergers
    - of quarterly performance

Indicators of Compromise

General Examples:

- Database tables missing
- Systems crashing
- Strange traffic on the network
- User machines abnormally slow?
- IDS alerts

Indicators of Compromise

Realistic examples:

- Combinations of suspicious metadata on a *victim's* system plus complex malicious code (Say a packed .dll or .exe)
- creation of suspicious registry keys + mutexes

IOC standards

The Incident Object Description Exchange Format

- http://www.openioc.org/
- http://www.ietf.org/rfc/rfc5070.txt

Response / Containment

After identifying the vector(s), perhaps:

- Fix firewall rules
- add malware signature to IDS/AV
- identify full extent of compromise

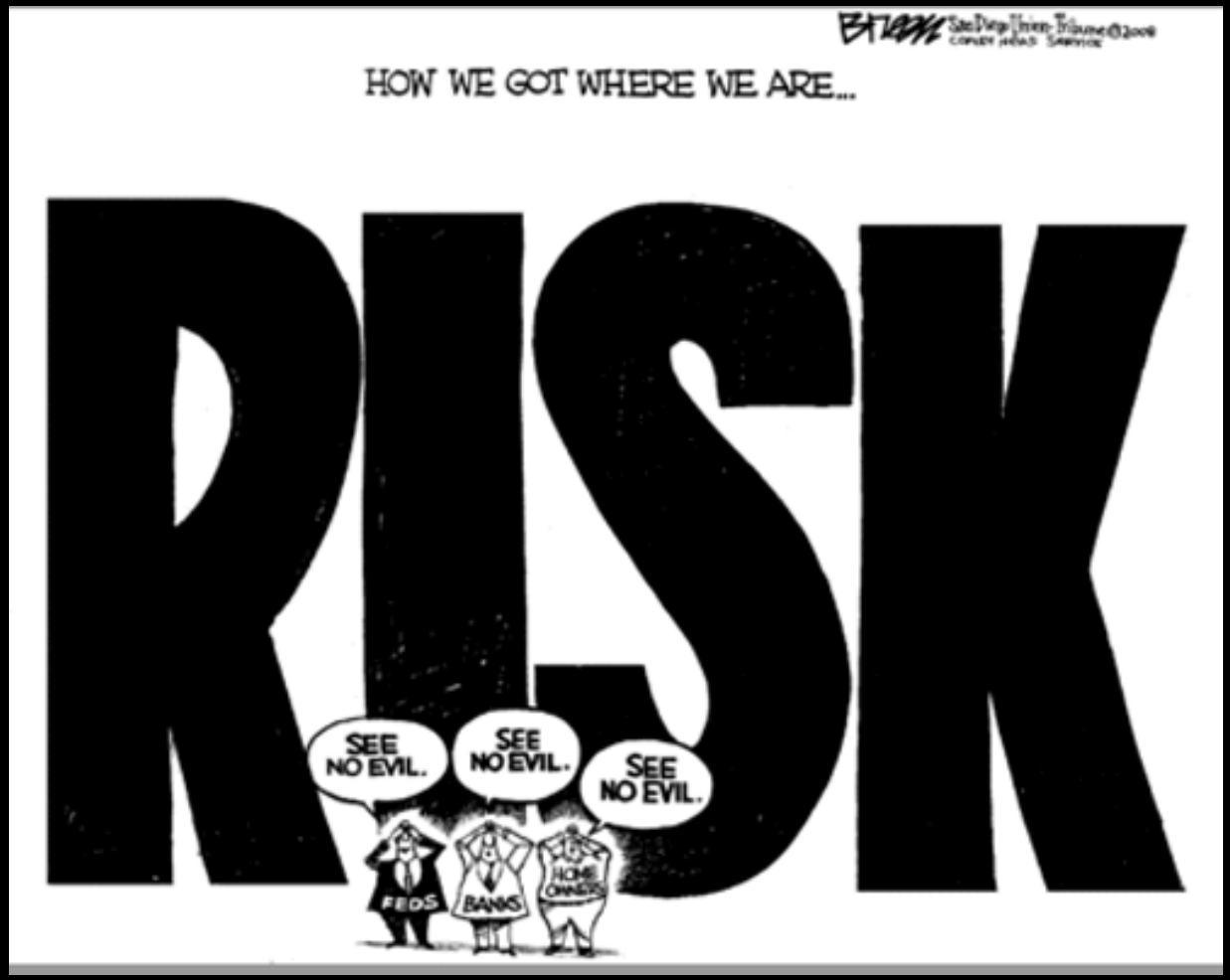
Results from a proper Incident Response

- Damage contained / stopped
- Attacker's vector identified / stopped
  - Hopefully patched, or in the process of being secured
- Incident response report
  - Impact of breach
    - details of the scope & damage done
  - Details of breach
  - How you addressed the incident
    - with IR budget
    - technical steps
    - indicators of compromise
    - How it was fixed & steps to take in the future

RISK

Company's risk:

- should be reviewed
- perhaps updated



IR toolkit

(Not comprehensive)

Depending on what you can afford

EnCase ($$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$)

Commercial IDS/IPS

The Sleuth Kit

Volatility

SysInternals Suite

IDA pro

A debugger

- Immunity, Ollydbg, WinDbg, etc...

(Not comprehensive list)

Volatility

a framework for extracting digital artifacts from RAM samples (virtual memory dumps)

- 32bit / 64bit Windows XP, 2003, Vista, 7

Has two interfaces

- single-command command line binary
- the interactive volshell

intermission to look at the volatility cheat sheet

Getting a memory dump for Volatility

A memory dump = binary file containing complete contents of systems memory

I use MoonSol's memory dump tools to get my memory dumps

http://www.moonsols.com/windows-memory-toolkit/

Using moonsols

win32dd (for 32bit) win64dd (for 64bit)

```
C:\>win32dd -d /f c:\memory.dmp
  win32dd - 1.3.1.20100417 - (Community Edition)
  Kernel land physical memory acquisition
  Copyright (C) 2007 - 2010, Matthieu Suiche (http://www.msuiche.net)
  Copyright (C) 2009 - 2010, MoonSols (http://www.moonsols.com)

    Name                        Value
    ----                        -----
    File type:                  Microsoft memory crash dump file
    Acquisition method:         PFN Mapping
    Content:                    Memory manager physical memory block

    Destination path:           c:\memory.dmp

    O.S. Version:               Microsoft Windows XP Home Edition (build 2600)
    Computer name:              TEST-POGZ2DOLZ7

    Physical memory in use:         19%
    Physical memory size:       1244656 Kb (   1215 Mb)
    Physical memory available:  1007556 Kb (    983 Mb)

    Paging file size:           2972492 Kb (   2902 Mb)
    Paging file available:      2888736 Kb (   2821 Mb)

    Virtual memory size:        2097024 Kb (   2047 Mb)
    Virtual memory available:   2084016 Kb (   2035 Mb)

    Extented memory available:        0 Kb (      0 Mb)

    Physical page size:         4096 bytes
    Minimum physical address:   0x0000000000001000
    Maximum physical address:   0x000000004BFEF000

    Address space size:         1275002880 bytes (1245120 Kb)

    --> Are you sure you want to continue? [y/n] _
```

Using moonsols

General output

```
--> Are you sure you want to continue? [y/n] y

Acquisition started at:     [27/3/2013 (DD/MM/YYYY) 22:14:20 (UTC)]

Processing....Done.

Acquisition finished at:  [2013-03-27 (YYYY-MM-DD) 22:14:30 (UTC)]
Time elapsed:             0:09 minutes:seconds (9 secs)

Created file size:          1274601472 bytes (   1215 Mb)

NtStatus (troubleshooting):   0x00000000
Total of written pages:        311182
Total of inacessible pages:         0
Total of accessible pages:     311182

Physical memory in use:         19%
Physical memory size:       1244656 Kb (   1215 Mb)
Physical memory available:  1007432 Kb (    983 Mb)

Paging file size:           2972492 Kb (   2902 Mb)
Paging file available:      2888692 Kb (   2820 Mb)

Virtual memory size:        2097024 Kb (   2047 Mb)
Virtual memory available:   2084016 Kb (   2035 Mb)

Extented memory available:        0 Kb (      0 Mb)

Physical page size:         4096 bytes
Minimum physical address:   0x0000000000001000
Maximum physical address:   0x000000004BFEF000

Address space size:         1275002880 bytes (1245120 Kb)
```

Using Volatility

Given a memory dump, we can analyze:

- process list / thread list
- process memory
- connections
- sockets
- dlls
- malware / backdoors in memory
  - *which may leave zero forensic evidence on disk!*

helpful volatility options

- -h for help
- -v for verbose

Volatility nuances

- silently fails given bad commands/options
- volatility's options are mostly community-written plugins
  - fail in weird ways
  - don't always work
  - don't work for all versions of windows
  - don't work for IPv6
- process memory dumps on malware/backdoors will often trigger your AV

Demo time

Volatility + Ida + yara

Scenario

I've exploited a vulnerable windows system with a meterpreter payload. I ran:

- getsystem
- execute -f calc.exe
- migrate (to calc.exe)

Then on the victim I acquired a memory dump at this moment afterwards.

Using volatility to find bad stuff

`malfind` plugin helps find hidden / injected code blocks in user mode memory.

- Based off of VAD tag and page permissions
- Can't detect DLL's injected into a process using CreateRemoteThead->LoadLibrary

Still helpful...

- Common tactic of hackers, malware
- Not a silver bullet
  - YMMV

A site note on finding injected/hidden DLLs

malfind

idrmodules

dlllist

impscan

...

Example of malfind to detect Zeus malware family: (not demo related)

```
$ python vol.py -f zeus.vmem malfind -p 1724
Volatile Systems Volatility Framework 2.1_alpha

Process: explorer.exe Pid: 1724 Address: 0x1600000
Vad Tag: VadS Protection: PAGE_EXECUTE_READWRITE
Flags: CommitCharge: 1, MemCommit: 1, PrivateMemory: 1, Protection: 6

0x01600000  b8 35 00 00 00 e9 cd d7 30 7b b8 91 00 00 00 e9   .5......0{......
0x01600010  4f df 30 7b 8b ff 55 8b ec e9 ef 17 c1 75 8b ff   O.0{..U......u..
0x01600020  55 8b ec e9 95 76 bc 75 8b ff 55 8b ec e9 be 53   U....v.u..U....S
0x01600030  bd 75 8b ff 55 8b ec e9 d6 18 c1 75 8b ff 55 8b   .u..U......u..U.

0x1600000 b835000000       MOV EAX, 0x35
0x1600005 e9cdd7307b       JMP 0x7c90d7d7
0x160000a b891000000       MOV EAX, 0x91
0x160000f e94fdf307b       JMP 0x7c90df63
0x1600014 8bff             MOV EDI, EDI
0x1600016 55               PUSH EBP

Process: explorer.exe Pid: 1724 Address: 0x15d0000
Vad Tag: VadS Protection: PAGE_EXECUTE_READWRITE
Flags: CommitCharge: 38, MemCommit: 1, PrivateMemory: 1, Protection: 6

0x015d0000  4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00   MZ..............
0x015d0010  b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00   ........@.......
0x015d0020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x015d0030  00 00 00 00 00 00 00 00 00 00 00 00 d0 00 00 00   ................

0x15d0000 4d               DEC EBP
0x15d0001 5a               POP EDX
0x15d0002 90               NOP
0x15d0003 0003             ADD [EBX], AL
0x15d0005 0000             ADD [EAX], AL
0x15d0007 000400           ADD [EAX+EAX], AL
0x15d000a 0000             ADD [EAX], AL
```

From: https://code.google.com/p/volatility/wiki/CommandReferenceMal22

Ok to the demo

With the memory dump of the victim machine,

we're going to do:

- Triage/Containment
- Response
- Resolution

```
D:\volatility>volatility-2.1.standalone.exe -f memory.dmp malfind
Volatile Systems Volatility Framework 2.1
Process: csrss.exe Pid: 560 Address: 0x7f6f0000
Vad Tag: Vad  Protection: PAGE_EXECUTE_READWRITE
Flags: Protection: 6

0x7f6f0000  c8 00 00 00 00 01 00 00 ff ee ff ee 08 70 00 00   ..............p..
0x7f6f0010  08 00 00 00 00 fe 00 00 00 00 10 00 00 20 00 00   .................
0x7f6f0020  00 02 00 00 00 20 00 00 8d 01 00 00 ff ef fd 7f   .................
0x7f6f0030  03 00 08 06 00 00 00 00 00 00 00 00 00 00 00 00   .................

0x7f6f0000 c8000000         ENTER 0x0, 0x0
0x7f6f0004 0001             ADD [ECX], AL
0x7f6f0006 0000             ADD [EAX], AL
0x7f6f0008 ff               DB 0xff
0x7f6f0009 ee               OUT DX, AL
0x7f6f000a ff               DB 0xff
0x7f6f000b ee               OUT DX, AL
0x7f6f000c 087000           OR [EAX+0x0], DH
0x7f6f000f 0008             ADD [EAX], CL
0x7f6f0011 0000             ADD [EAX], AL
0x7f6f0013 0000             ADD [EAX], AL
0x7f6f0015 fe00             INC BYTE [EAX]
0x7f6f0017 0000             ADD [EAX], AL
0x7f6f0019 0010             ADD [EAX], DL
0x7f6f001b 0000             ADD [EAX], AL
0x7f6f001d 2000             AND [EAX], AL
0x7f6f001f 0000             ADD [EAX], AL
0x7f6f0021 0200             ADD AL, [EAX]
0x7f6f0023 0000             ADD [EAX], AL
0x7f6f0025 2000             AND [EAX], AL
0x7f6f0027 008d010000ff     ADD [EBP-0xffffff], CL
0x7f6f002d ef               OUT DX, EAX
0x7f6f002e fd               STD
0x7f6f002f 7f03             JG 0x7f6f0034
0x7f6f0031 0008             ADD [EAX], CL
0x7f6f0033 06               PUSH ES
0x7f6f0034 0000             ADD [EAX], AL
0x7f6f0036 0000             ADD [EAX], AL
0x7f6f0038 0000             ADD [EAX], AL
0x7f6f003a 0000             ADD [EAX], AL
0x7f6f003c 0000             ADD [EAX], AL
0x7f6f003e 0000             ADD [EAX], AL

Process: Foxit Reader.ex Pid: 2008 Address: 0x21c0000
Vad Tag: VadS Protection: PAGE_EXECUTE_READWRITE
Flags: CommitCharge: 184, MemCommit: 1, PrivateMemory: 1, Protection: 6

0x021c0000  4d 5a e8 00 00 00 00 5b 52 45 55 89 e5 81 c3 37   MZ.....[REU....7
0x021c0010  15 00 00 ff d3 89 c3 57 68 04 00 00 00 50 ff d0   .......Wh....P..
0x021c0020  68 f0 b5 a2 56 68 05 00 00 00 50 ff d3 00 00 00   h...Vh....P.....
0x021c0030  00 00 00 00 00 00 00 00 00 00 00 00 e0 00 00 00   .................

0x21c0000 4d                DEC EBP
0x21c0001 5a                POP EDX
0x21c0002 e800000000        CALL 0x21c0007
0x21c0007 5b                POP EBX
0x21c0008 52                PUSH EDX
```

Results from malfind

malfind -D C:\output\directory\...

```
D:\volatility>volatility-2.1.standalone.exe -f memory.dmp malfind -D D:\volatility\
Volatile Systems Volatility Framework 2.1
Process: csrss.exe Pid: 560 Address: 0x7f6f0000
Vad Tag: Vad  Protection: PAGE_EXECUTE_READWRITE
Flags: Protection: 6

0x7f6f0000  c8 00 00 00 00 01 00 00 ff ee ff ee 08 70 00 00   .............p..
0x7f6f0010  08 00 00 00 00 fe 00 00 00 00 10 0
0x7f6f0020  00 02 00 00 00 20 00 00 8d 01 00 0
0x7f6f0030  03 00 08 06 00 00 00 00 00 00 00 0

0x7f6f0000 c8000000          ENTER 0x0, 0x0
0x7f6f0004 0001              ADD [ECX], AL
0x7f6f0006 0000              ADD [EAX], AL
0x7f6f0008 ff                DB 0xff
0x7f6f0009 ee                OUT DX, AL
0x7f6f000a ff                DB 0xff
0x7f6f000b ee                OUT DX, AL
0x7f6f000c 087000            OR [EAX+0x0], DH
0x7f6f000f 0008              ADD [EAX], CL
0x7f6f0011 0000              ADD [EAX], AL
0x7f6f0013 0000              ADD [EAX], AL
0x7f6f0015 fe00              INC BYTE [EAX]
0x7f6f0017 0000              ADD [EAX], AL
0x7f6f0019 0010              ADD [EAX], DL
0x7f6f001b 0000              ADD [EAX], AL
0x7f6f001d 2000              AND [EAX], AL
0x7f6f001f 0000              ADD [EAX], AL
0x7f6f0021 0200              ADD AL, [EAX]
0x7f6f0023 0000              ADD [EAX], AL
0x7f6f0025 2000              AND [EAX], AL
0x7f6f0027 008d010000ff      ADD [EBP-0xffffff]
0x7f6f002d ef                OUT DX, EAX
```

Computer ▸ Local Disk (D:) ▸ volatility

Organize ▾ Open New folder

.splunk
.VirtualBox
Contacts
Desktop
Downloads
Dropbox
Favorites
Links
My Documents
My Music
My Pictures
My Videos
Saved Games
Searches
VirtualBox VMs
BT5 R3
CTF4
Hacking Art o

Name	Date modifi
be2.vmem	3/27/2013 4
conns	3/27/2013 9
coreflood.vmem	3/27/2013 4
laqma.vmem	3/27/2013 4
memory.dmp	3/27/2013 2
process.0x885d62d0.0x7f6f0000.dmp	3/27/2013 9
process.0x8846e020.0x21c0000.dmp	3/27/2013 9
process.0x8846e020.0x2280000.dmp	3/27/2013 9
process.0x8846e020.0x2550000.dmp	3/27/2013 9
process.0x8846e020.0x2630000.dmp	3/27/2013 9
process.0x8855d020.0x890000.dmp	3/27/2013 9
process.0x8855d020.0xa50000.dmp	3/27/2013 9
process.0x8855d020.0xc60000.dmp	3/27/2013 9
process.0x8855d020.0xd40000.dmp	3/27/2013 9
volatility-2.1.standalone	3/27/2013 4
x	3/27/2013 9

Output files from malfind are highlighted

.dmp files are binary data

Some are valid PE files...

Investigation w/ IDA

Since when does calc.exe need "priv_elevate_getsystem"?

```
xor     edx, edx
test    eax, eax
setnle  dl
lea     eax, [edx+edx-1]
test    eax, eax
jnz     loc_1000750F
```

```
000742E:
    eax, byte ptr [ecx+1]
    edx, byte ptr [esi+1]
    eax, edx
    short loc_1000744D
```

Line 2 of 2

Strings window

Address	Length	Type	String
.rdata:1...	0000001D	C	priv_fs_blank_directory_mace
.rdata:1...	00000018	C	priv_fs_blank_file_mace
.rdata:1...	00000020	C	priv_fs_set_file_mace_from_file
.rdata:1...	00000016	C	priv_fs_set_file_mace
.rdata:1...	00000016	C	priv_fs_get_file_mace
.rdata:1...	0000001B	C	priv_passwd_get_sam_hashes
.rdata:1...	00000017	C	priv_elevate_getsystem
.rdata:1...	0000000A	C	lsass.exe
.rdata:1...	00000011	C	SeDebugPrivilege
.rdata:1...	00000005	C	FREE
.rdata:1...	00000009	C	wcstombs
.rdata:1...	0000000A	C	ntdll.dll
.rdata:1...	00000007	C	memcpy
.rdata:1...	00000005	C	free

Line 13 of 268

Why would calc.exe or any other process have these strings????

in the .rdata

Hex View-A

```
.rdata:100135E0  F9 81 E7 77 66 C8 E7 77  B1 79 E7 77 F9 3F E7 77  ·ütwf+tw¦ytw·?tw
.rdata:100135F0  00 00 00 00 DD 32 BF 76  0B 1E BF 76 00 00 00 00  ....¦2+v▯+v....
.rdata:10013600  FF CB D4 77 00 00 00 00  00 00 00 00 00 00 00 00   -+w............
.rdata:10013610  00 00 00 00 E5 8B D4 00  D6 C4 D4 00 1C D4 D4 00  ....sï+.+-+.++.
.rdata:10013620  B7 DC D4 00 00 00 00 00  00 00 00 00 CB 09 D5 00  +_+..........-+..
.rdata:10013630  CD D4 D4 00 00 00 00 00  00 00 00 00 00 00 00 00  -++.............
.rdata:10013640  70 72 69 76 5F 66 73 5F  62 6C 61 6E 6B 5F 64 69  priv_fs_blank_di
.rdata:10013650  72 65 63 74 6F 72 79 5F  6D 61 63 65 00 00 00 00  rectory_mace....
.rdata:10013660  70 72 69 76 5F 66 73 5F  62 6C 61 6E 6B 5F 66 69  priv_fs_blank_fi
.rdata:10013670  6C 65 5F 6D 61 63 65 00  70 72 69 76 5F 66 73 5F  le_mace.priv_fs_
.rdata:10013680  73 65 74 5F 66 69 6C 65  5F 6D 61 63 65 5F 66 72  set_file_mace_fr
.rdata:10013690  6F 6D 5F 66 69 6C 65 00  70 72 69 76 5F 66 73 5F  om_file.priv_fs_
.rdata:100136A0  73 65 74 5F 66 69 6C 65  5F 6D 61 63 65 00 00 00  set_file_mace...
.rdata:100136B0  70 72 69 76 5F 66 73 5F  67 65 74 5F 66 69 6C 65  priv_fs_get_file
.rdata:100136C0  5F 6D 61 63 65 00 00 00  70 72 69 76 5F 70 61 73  _mace...priv_pas
.rdata:100136D0  73 77 64 5F 67 65 74 5F  73 61 6D 5F 68 61 73 68  swd_get_sam_hash
.rdata:100136E0  65 73 00 00 70 72 69 76  5F 65 6C 65 76 61 74 65  es..priv_elevate
.rdata:100136F0  5F 67 65 74 73 79 73 74  65 6D 00 00 6C 73 61 73  _getsystem..lsas
.rdata:10013700  73 2E 65 78 65 00 00 00  53 65 44 65 62 75 67 50  s.exe...SeDebugP
.rdata:10013710  72 69 76 69 6C 65 67 65  00 00 00 00 46 52 45 45  rivilege....FREE
.rdata:10013720  00 00 00 00 53 41 4D 00  77 63 73 74 6F 6D 62 73  ....SAM.wcstombs
.rdata:10013730  00 00 00 00 6E 74 64 6C  6C 2E 64 6C 6C 00 00 00  ....ntdll.dll...
.rdata:10013740  6D 65 6D 63 70 79 00 00  66 72 65 65 00 00 00 00  memcpy..free....
.rdata:10013750  72 65 61 6C 6C 6F 63 00  6D 61 6C 6C 6F 63 00 00  realloc.malloc..
.rdata:10013760  6D 73 76 63 72 74 2E 64  6C 6C 00 00 4C 73 61 43  msvcrt.dll..LsaC
.rdata:10013770  6C 6F 73 65 00 00 00 00  4C 73 61 51 75 65 72 79  lose....LsaQuery
.rdata:10013780  49 6E 66 6F 72 6D 61 74  69 6F 6E 50 6F 6C 69 63  InformationPolic
.rdata:10013790  79 00 00 00 4C 73 61 4F  70 65 6E 50 6F 6C 69 63  y...LsaOpenPolic
.rdata:100137A0  79 00 00 00 61 64 76 61  70 69 33 32 2E 64 6C 6C  y...advapi32.dll
.rdata:100137B0  00 00 00 00 53 61 6D 72  43 6C 6F 73 65 48 61 6E  ....SamrCloseHan
.rdata:100137C0  64 6C 65 00 53 61 6D 49  46 72 65 65 5F 53 41 4D  dle.SamIFree_SAM
.rdata:100137D0  50 52 5F 45 4E 55 4D 45  52 41 54 49 4F 4E 5F 42  PR_ENUMERATION_B
.rdata:100137E0  55 46 46 45 52 00 00 00  53 61 6D 49 46 72 65 65  UFFER...SamIFree
.rdata:100137F0  5F 53 41 4D 50 52 5F 55  53 45 52 5F 49 4E 46 4F  _SAMPR_USER_INFO
.rdata:10013800  5F 42 55 46 46 45 52 00  53 61 6D 72 45 6E 75 6D  _BUFFER.SamrEnum
.rdata:10013810  65 72 61 74 65 55 73 65  72 73 49 6E 44 6F 6D 61  erateUsersInDoma
.rdata:10013820  69 6E 00 00 53 61 6D 72  51 75 65 72 79 49 6E 66  in..SamrQueryInf
```

Indicator of Compromise

These two strings shouldn't exist in calc.exe, or really any process

- Strong IoC
- Strong signature too!

Strings window

Address	Length	Type	String
.rdata:1...	0000001D	C	priv_fs_blank_directory_mace
.rdata:1...	00000018	C	priv_fs_blank_file_mace
.rdata:1...	00000020	C	priv_fs_set_file_mace_from_file
.rdata:1...	00000016	C	priv_fs_set_file_mace
.rdata:1...	00000016	C	priv_fs_get_file_mace
.rdata:1...	0000001B	C	priv_passwd_get_sam_hashes
.rdata:1...	00000017	C	priv_elevate_getsystem
.rdata:1...	0000000A	C	lsass.exe
.rdata:1...	00000011	C	SeDebugPrivilege
.rdata:1...	00000005	C	FREE
.rdata:1...	00000009	C	wcstombs
.rdata:1...	0000000A	C	ntdll.dll
.rdata:1...	00000007	C	memcpy
.rdata:1...	00000005	C	free

Line 13 of 268

Containment

Given these strings, it is possible to write a signature to detect this

- on Host-based-IDS
- on AV
- with custom RAT (perhaps)

Here's a Yara rule to detect meterpreter in memory:

```
rule MeterpreterDetected
{
    strings:
        $a = "priv_elevate_getsystem"
        $b = "priv_passwd_get_sam_hashes"
    condition:
        $a and $b
}
```

Containment

First test the signature to see how many false positives occur - Building a whitelist to exclude processes may be necessary.

```
rule MeterpreterDetected
{
    strings:
        $a = "priv_elevate_getsystem"
        $b = "priv_passwd_get_sam_hashes"
    condition:
        $a and $b
}
```

Containment

```
rule MeterpreterDetected
{
    strings:
        $a = "priv_elevate_getsystem"
        $b = "priv_passwd_get_sam_hashes"
        $whitelist1 = "smss.exe"
    condition:
        ($a and $b) and not $whitelist1
}
```

If all is good, establish a whitelist, then update HIDS systems with this Yara rule, with the action to kill any matching process.

All open meterpreter sessions will be killed (hopefully)

A side note

"priv_elevate_getsystem" and "priv_passwd_get_sam_hashes"

are part of the meterpreter standard api dll - which is loaded every time by default

- does not indicate that the attacker has compromised a SYSTEM token

Alternate rules

Strings for signatures can be text and/or hexadecimal. Also wildcards..

example:

```
$hex_string = { A2 34 ?? C8 A? FF }
```

Who uses Yara

- VirusTotal Intelligence
- jsunpack-n
- FireEye
- We Watch Your Website
- ClamAV (with a yara extension)
- ...
- Volatility!!!
  - **yarascan :D**

But only two of these are a HIDS / AV

Response

Further investigation is needed to determine how the attack happened

We can use *yarascan* to automate the detection of the attackers in any other memory dumps, with our yara rule

- (not going to demo this... just a fun fact)

Response

We want to:

1. Identify where the attacker is coming from
2. Identify whether the attacker compromised the SYSTEM token
   - if so, then there's a higher chance of a rootkit

apihooks

```
D:\volatility>volatility-2.1.standalone.exe -f memory.dmp apihooks
Volatile Systems Volatility Framework 2.1
**************************************************************************
Hook mode: Usermode
Hook type: Inline/Trampoline
Process: 972 (svchost.exe)
Victim module: MPRAPI.dll (0x76d40000 - 0x76d56000)
Function: MPRAPI.dll!MprAdminMIBServerDisconnect at 0x76d42717
Hook address: 0x77cc4008
Hooking module: RPCRT4.dll

Disassembly(0):
0x76d42717 8d442404          LEA EAX, [ESP+0x4]
0x76d4271b 50                PUSH EAX
0x76d4271c ff151c12d476      CALL DWORD [0x76d4121c]
0x76d42722 c20400            RET 0x4
0x76d42725 6a18              PUSH 0x18
0x76d42727 6820f3d476        PUSH DWORD 0x76d4f320
0x76d4272c e8                DB 0xe8
0x76d4272d 2f                DAS
0x76d4272e ec                IN AL, DX

Disassembly(1):
0x77cc4008 56                PUSH ESI
0x77cc4009 57                PUSH EDI
0x77cc400a e843000000        CALL 0x77cc4052
0x77cc400f 84c0              TEST AL, AL
0x77cc4011 8b7c240c          MOV EDI, [ESP+0xc]
0x77cc4015 0f8582010000      JNZ 0x77cc419d
0x77cc401b 83                DB 0x83
0x77cc401c 3d                DB 0x3d
0x77cc401d d8f0              FDIV ST0, ST0
0x77cc401f d2                DB 0xd2

**************************************************************************
Hook mode: Usermode
Hook type: Inline/Trampoline
Process: 972 (svchost.exe)
Victim module: MPRAPI.dll (0x76d40000 - 0x76d56000)
Function: MPRAPI.dll!MprAdminServerDisconnect at 0x76d42717
Hook address: 0x77cc4008
Hooking module: RPCRT4.dll

Disassembly(0):
0x76d42717 8d442404          LEA EAX, [ESP+0x4]
0x76d4271b 50                PUSH EAX
```

apihooks...

```
*****************************************************************
Hook mode: Usermode
Hook type: Inline/Trampoline
Process: 2008 (Foxit Reader.ex)
Victim module: OLEPRO32.DLL (0x5edd0000 - 0x5edea000)
Function: OLEPRO32.DLL!OleCreatePictureIndirect at 0x5ede0930
Hook address: 0x7717a248
Hooking module: OLEAUT32.dll

Disassembly(0):
0x5ede0930 ff255434de5e      JMP DWORD [0x5ede3454]
0x5ede0936 90                NOP
0x5ede0937 90                NOP
0x5ede0938 90                NOP
0x5ede0939 90                NOP
0x5ede093a 90                NOP
0x5ede093b 90                NOP
0x5ede093c 90                NOP
0x5ede093d 90                NOP
0x5ede093e 90                NOP
0x5ede093f 90                NOP
0x5ede0940 ff255834de5e      JMP DWORD [0x5ede3458]
0x5ede0946 90                NOP
0x5ede0947 90                NOP

Disassembly(1):
0x7717a248 53                PUSH EBX
0x7717a249 55                PUSH EBP
0x7717a24a 8b6c2418          MOV EBP, [ESP+0x18]
0x7717a24e 689c000000        PUSH DWORD 0x9c
0x7717a253 c7450000000000    MOV DWORD [EBP+0x0], 0x0
0x7717a25a e80f89faff        CALL 0x77122b6e
0x7717a25f 83                DB 0x83

*****************************************************************
Hook mode: Usermode
Hook type: Inline/Trampoline
Process: 2008 (Foxit Reader.ex)
Victim module: OLEPRO32.DLL (0x5edd0000 - 0x5edea000)
Function: OLEPRO32.DLL!OleCreatePropertyFrame at 0x5ede0950
Hook address: 0x771755f9
Hooking module: OLEAUT32.dll

Disassembly(0):
0x5ede0950 ff256434de5e      JMP DWORD [0x5ede3464]
0x5ede0956 90                NOP
```

apihooks....

```
*****************************************************************
Hook mode: Usermode
Hook type: Inline/Trampoline
Process: 1964 (wuauclt.exe)
Victim module: MPRAPI.dll (0x76d40000 - 0x76d56000)
Function: MPRAPI.dll!MprAdminMIBServerDisconnect at 0x76d42717
Hook address: 0x77cc4008
Hooking module: RPCRT4.dll

Disassembly(0):
0x76d42717 8d442404          LEA EAX, [ESP+0x4]
0x76d4271b 50                PUSH EAX
0x76d4271c ff151c12d476      CALL DWORD [0x76d4121c]
0x76d42722 c20400            RET 0x4
0x76d42725 6a18              PUSH 0x18
0x76d42727 6820f3d476        PUSH DWORD 0x76d4f320
0x76d4272c e8                DB 0xe8
0x76d4272d 2f                DAS
0x76d4272e ec                IN AL, DX

Disassembly(1):
0x77cc4008 56                PUSH ESI
0x77cc4009 57                PUSH EDI
0x77cc400a e843000000        CALL 0x77cc4052
0x77cc400f 84c0              TEST AL, AL
0x77cc4011 8b7c240c          MOV EDI, [ESP+0xc]
0x77cc4015 0f8582010000      JNZ 0x77cc419d
0x77cc401b 83                DB 0x83
0x77cc401c 3d                DB 0x3d
0x77cc401d d8f0              FDIV ST0, ST0
0x77cc401f d2                DB 0xd2

*****************************************************************
Hook mode: Usermode
Hook type: Inline/Trampoline
Process: 1964 (wuauclt.exe)
Victim module: MPRAPI.dll (0x76d40000 - 0x76d56000)
Function: MPRAPI.dll!MprAdminServerDisconnect at 0x76d42717
Hook address: 0x77cc4008
Hooking module: RPCRT4.dll

Disassembly(0):
0x76d42717 8d442404          LEA EAX, [ESP+0x4]
0x76d4271b 50                PUSH EAX
0x76d4271c ff151c12d476      CALL DWORD [0x76d4121c]
0x76d42722 c20400            RET 0x4
```

Analysis

We've detected API hooks present in:

- svchost.exe (SYSTEM process)
- Foxit Reader.exe (user process)
- wuauclt.exe (user process)
  - windows autoupdate client

What could these indicators mean?

Analyzing the open connections

```
D:\volatility>volatility-2.1.standalone.exe -f memory.dmp connscan
Volatile Systems Volatility Framework 2.1
Offset(P)  Local Address             Remote Address            Pid
---------- ------------------------- ------------------------- ---
0x08bb0008 192.168.1.10:1056         192.168.1.1:49152         1208
0x08c0be68 192.168.1.10:5000         192.168.1.1:38170         1208
0x08c1fe18 192.168.1.10:1049         192.168.1.1:49152         1208
0x08c346a0 192.168.1.10:1028         192.168.1.161:443         2008
0x08dd0240 192.168.1.10:1050         192.168.1.1:49152         1208
```

`connscan` can be very slow, but we see 4 connections open, among two IP addrs

- 192.168.1.10 - 192.168.1.1 (gateway)
- 192.168.1.10 - 192.168.1.161

Normally have to sift through lots of connections

Conclusion

We used Volatility to find:

- The backdoor
- The compromised process
- The attacker's IP
- An indicator that the attack vector was a Foxit Reader.exe exploit
- and an indicator that the attacker compromised a SYSTEM token
  - API hooks in SYSTEM process svchost.exe

In many scenarios attackers will leave zero disk forensic evidence --- and the only evidence will be in volatile memory

Volshell

This is a interactive (but limited) shell in the volatility framework given a memory dump.

- Built on top of Python interpreter
  - can leverage everything in Python

>volatility-2.1.standalone.exe -f be2.vmem --profile=WinXPSP2x86 volshell

or

$ python volatility volshell -f xp-laptop-2005-07-04-1430.img

For help:

- hh()
- http://moyix.blogspot.com/2008/08/indroducing-volshell.html

\x00

Questions?