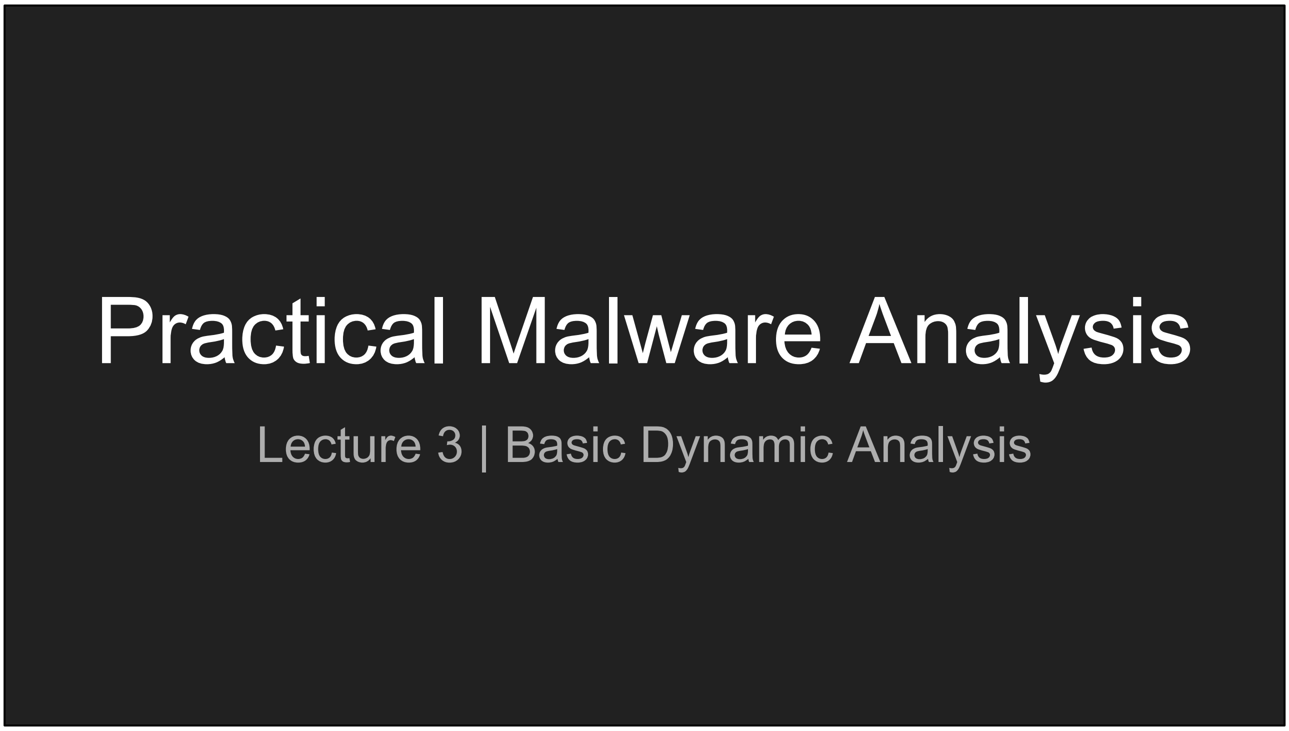


Practical Malware Analysis

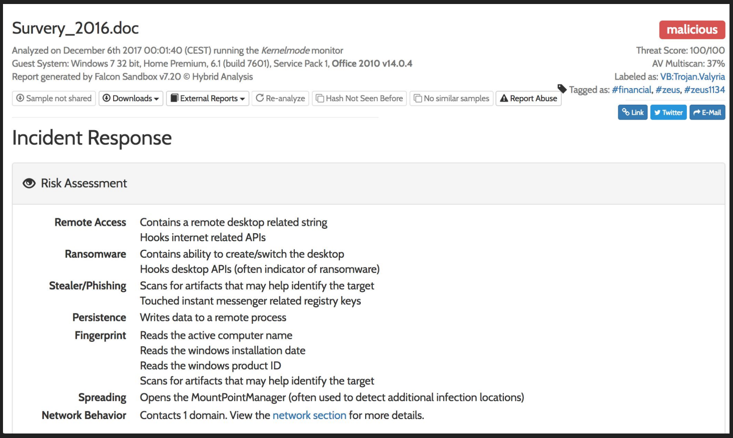
Lecture 3 | Basic Dynamic Analysis

Dynamic analysis

- After basic analysis dead-ends
- Monitor the malware as it runs and view changes to the system after execution

Sandbox Automation

- Public or private
- Quick



Survery_2016.doc

malicious

Analyzed on December 6th 2017 00:01:40 (CEST) running the *Kernelmode* monitor
Guest System: Windows 7 32 bit, Home Premium, 6.1 (build 7601), Service Pack 1, **Office 2010 v14.0.4**
Report generated by Falcon Sandbox v7.20 © Hybrid Analysis

Threat Score: 100/100
AV Multiscan: 37%
Labeled as: VB:Trojan.Valyria
Tagged as: #financial, #zeus, #zeus1134

Incident Response

Risk Assessment

Remote Access	Contains a remote desktop related string Hooks internet related APIs
Ransomware	Contains ability to create/switch the desktop Hooks desktop APIs (often indicator of ransomware)
Stealer/Phishing	Scans for artifacts that may help identify the target Touched instant messenger related registry keys
Persistence	Writes data to a remote process
Fingerprint	Reads the active computer name Reads the windows installation date Reads the windows product ID Scans for artifacts that may help identify the target
Spreading	Opens the MountPointManager (often used to detect additional infection locations)
Network Behavior	Contacts 1 domain. View the network section for more details.

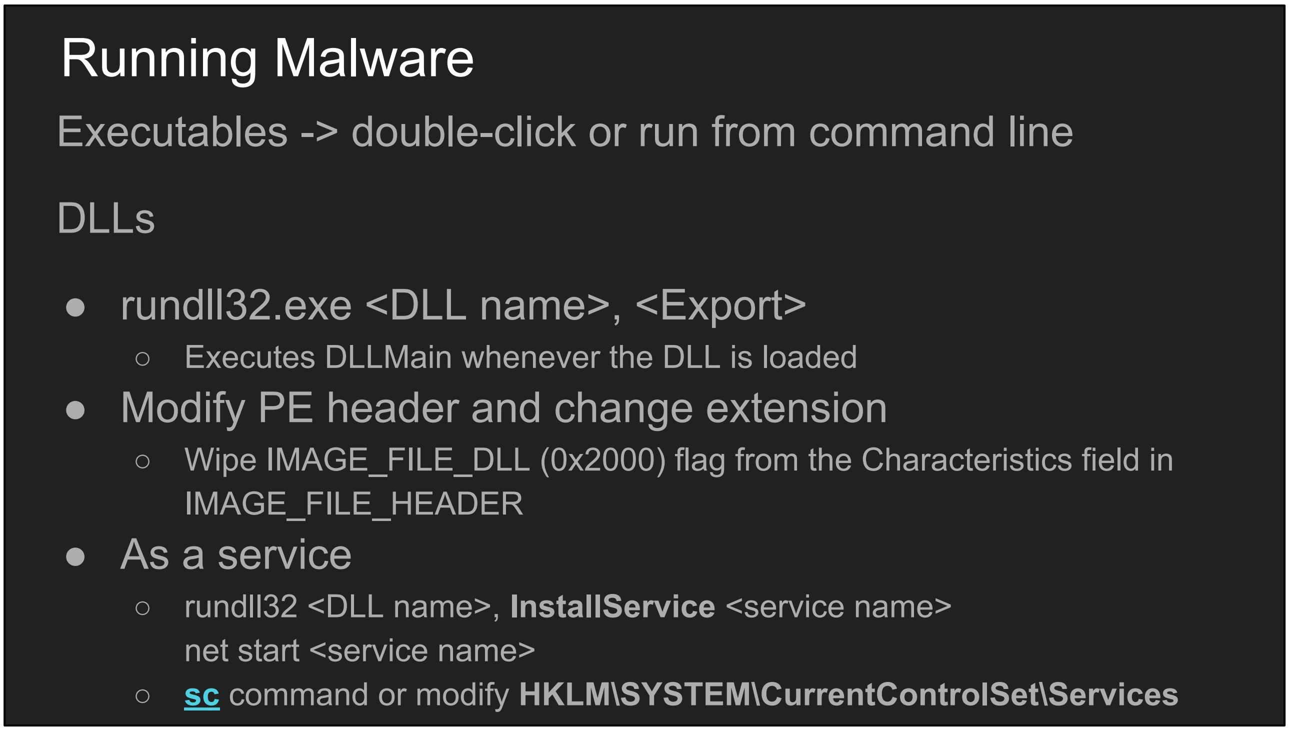
- Oversimplified
- Anti-VM detection exists

A quick and dirty way to perform some basic dynamic analysis is to run a sample through a sandboxed environment that is configured to simulate services and log changes to the filesystem made by the malware

You may upload samples to publically available sandboxing sites, or host a sandbox internally for analysing sensitive samples
Sandboxes are quick and easy, great for doing analysis in bulk or gathering low-hanging fruit

However, these environments are oversimplified and may not mimic the human interaction, network interaction, or environment that malware needs to execute properly

Malware may also detect that it is being executed within a sandbox and terminate or behave differently that it normally would



Running Malware

Executables -> double-click or run from command line

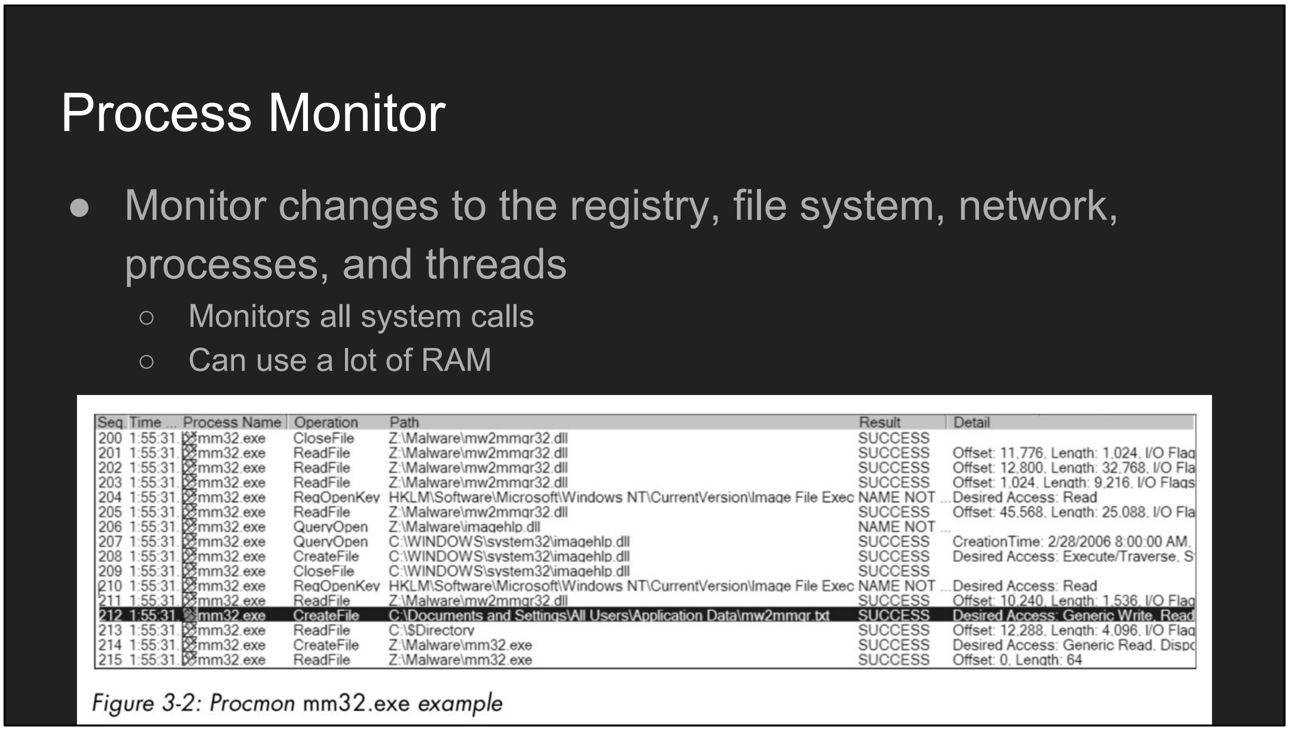
DLLs

- rundll32.exe <DLL name>, <Export>
  - Executes DLLMain whenever the DLL is loaded
- Modify PE header and change extension
  - Wipe IMAGE_FILE_DLL (0x2000) flag from the Characteristics field in IMAGE_FILE_HEADER
- As a service
  - rundll32 <DLL name>, **InstallService** <service name>
    net start <service name>
  - **sc** command or modify **HKLM\SYSTEM\CurrentControlSet\Services**

Running malware can be as easy as double-clicking a malicious executable, however, DLLs can be more complicated to run solo because Windows doesn't know how to automatically launch them

Windows has a container for running DLLs called rundll32 that can be called with the name of the DLL and the name or ordinal of the export to run, this calls DLLMain from the DLL entry point and runs whatever code is in the main function
You can also trick Windows into running a DLL as an executable by modifying the PE header and changing its extension

When a DLL needs to be installed as a service (a program that runs in the background) it will usually contain an export such as InstallService to install and start the service, if it does not you may need to manually install the service using the Windows 'sc' command or by modifying a registry entry for an unused service and then running it with the 'net start' command



Process Monitor

- Monitor changes to the registry, file system, network, processes, and threads
  - Monitors all system calls
  - Can use a lot of RAM

Seq.	Time ...	Process Name	Operation	Path	Result	Detail
200	1:55:31.	mm32.exe	CloseFile	Z:\Malware\mw2mmgr32.dll	SUCCESS	
201	1:55:31.	mm32.exe	ReadFile	Z:\Malware\mw2mmgr32.dll	SUCCESS	Offset: 11,776, Length: 1,024, I/O Flag
202	1:55:31.	mm32.exe	ReadFile	Z:\Malware\mw2mmgr32.dll	SUCCESS	Offset: 12,800, Length: 32,768, I/O Fla
203	1:55:31.	mm32.exe	ReadFile	Z:\Malware\mw2mmgr32.dll	SUCCESS	Offset: 1,024, Length: 9,216, I/O Flags
204	1:55:31.	mm32.exe	RegOpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Exec	NAME NOT ...	Desired Access: Read
205	1:55:31.	mm32.exe	ReadFile	Z:\Malware\mw2mmgr32.dll	SUCCESS	Offset: 45,568, Length: 25,088, I/O Fla
206	1:55:31.	mm32.exe	QueryOpen	Z:\Malware\imagehlp.dll	NAME NOT ...	
207	1:55:31.	mm32.exe	QueryOpen	C:\WINDOWS\system32\imagehlp.dll	SUCCESS	CreationTime: 2/28/2006 8:00:00 AM,
208	1:55:31.	mm32.exe	CreateFile	C:\WINDOWS\system32\imagehlp.dll	SUCCESS	Desired Access: Execute/Traverse, S
209	1:55:31.	mm32.exe	CloseFile	C:\WINDOWS\system32\imagehlp.dll	SUCCESS	
210	1:55:31.	mm32.exe	RegOpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Exec	NAME NOT ...	Desired Access: Read
211	1:55:31.	mm32.exe	ReadFile	Z:\Malware\mw2mmgr32.dll	SUCCESS	Offset: 10,240, Length: 1,536, I/O Flag
212	1:55:31.	mm32.exe	CreateFile	C:\Documents and Settings\All Users\Application Data\mw2mmgr.txt	SUCCESS	Desired Access: Generic Write, Read
213	1:55:31.	mm32.exe	ReadFile	C:\$Directory	SUCCESS	Offset: 12,288, Length: 4,096, I/O Flag
214	1:55:31.	mm32.exe	CreateFile	Z:\Malware\mm32.exe	SUCCESS	Desired Access: Generic Read, Dispo
215	1:55:31.	mm32.exe	ReadFile	Z:\Malware\mm32.exe	SUCCESS	Offset: 0, Length: 64

Figure 3-2: Procmon mm32.exe *example*

Monitor changes to the registry, file system, network, processes, and threads
Monitors all system calls
Can use a lot of RAM

Process Monitor

Filters are key

- On system calls (Operations)
- Process names
- Detail

GUI buttons to switch between Registry, File system, Process activity, and Network filters

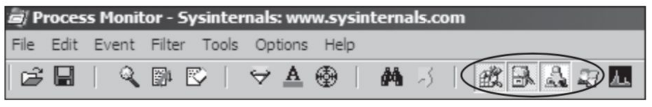

Process Monitor - Sysinternals: www.sysinternals.com
File Edit Event Filter Tools Options Help

Figure 3-4: Filter buttons for procmon

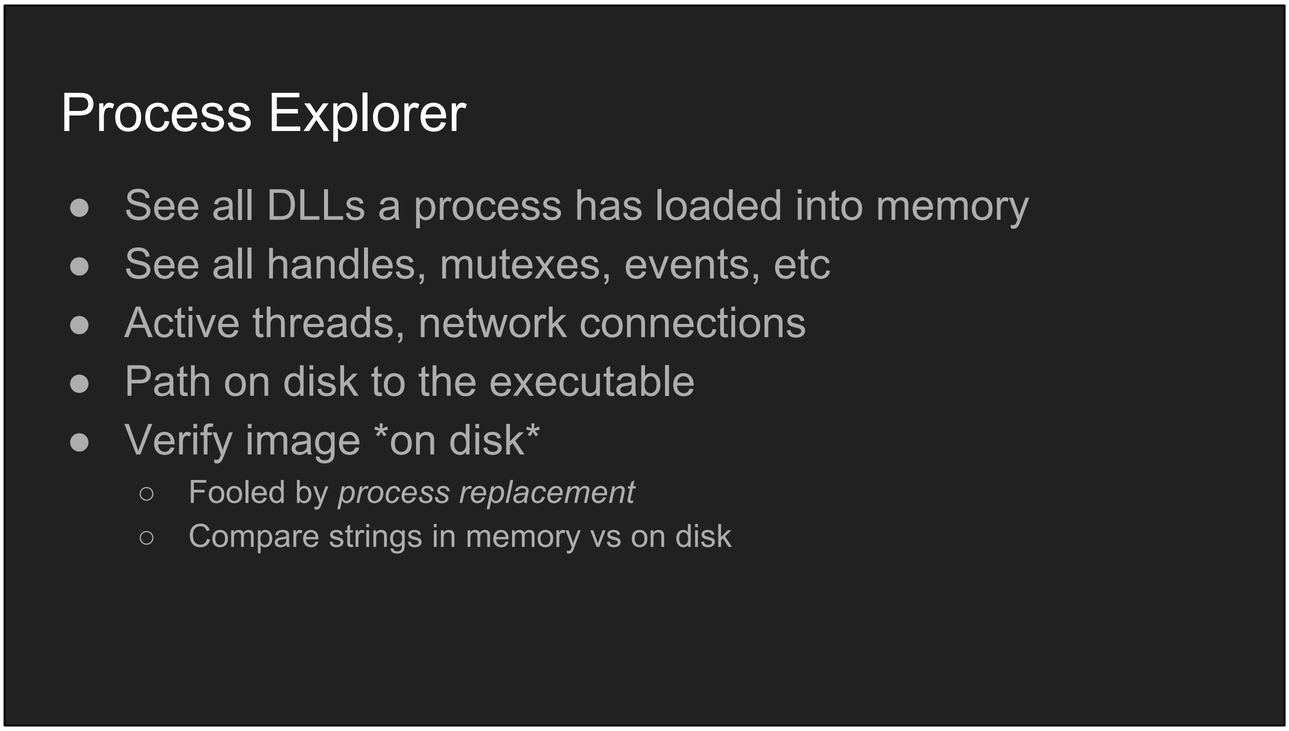
Filters are key
On system calls (Operations)
Process names
Detail
GUI buttons to switch between Registry, File system, Process activity, and Network filters

Process Explorer

(Task Manager ++) Lists active processes and their properties

- DLLs associated with a process
- Kill processes
- Log users out
- Launch and validate processes
  - Display updates every second
  - Pink = service, blue = process, green and red (temp) = new or killed

(Task Manager ++) Lists active processes and their properties
DLLs associated with a process
Kill processes
Log users out
Launch and validate processes
Display updates every second
Pink = service, blue = process, green and red (temp) = new or killed



Process Explorer

- See all DLLs a process has loaded into memory
- See all handles, mutexes, events, etc
- Active threads, network connections
- Path on disk to the executable
- Verify image *on disk*
  - Fooled by *process replacement*
  - Compare strings in memory vs on disk

See all DLLs a process has loaded into memory
See all handles, mutexes, events, etc
Active threats, network connections
Path on disk to the executable

Verify the image on disk (uses the WinVerifyTrust function, which uses certificate information to verify a file…). This can be fooled by process replacement, a technique where malicious code overwrites a processes’ memory space without touching the disk, also gaining the privileges of whatever process is being replaced. The image in memory will be different from the image on disk and comparing strings in the properties window will reflect this.

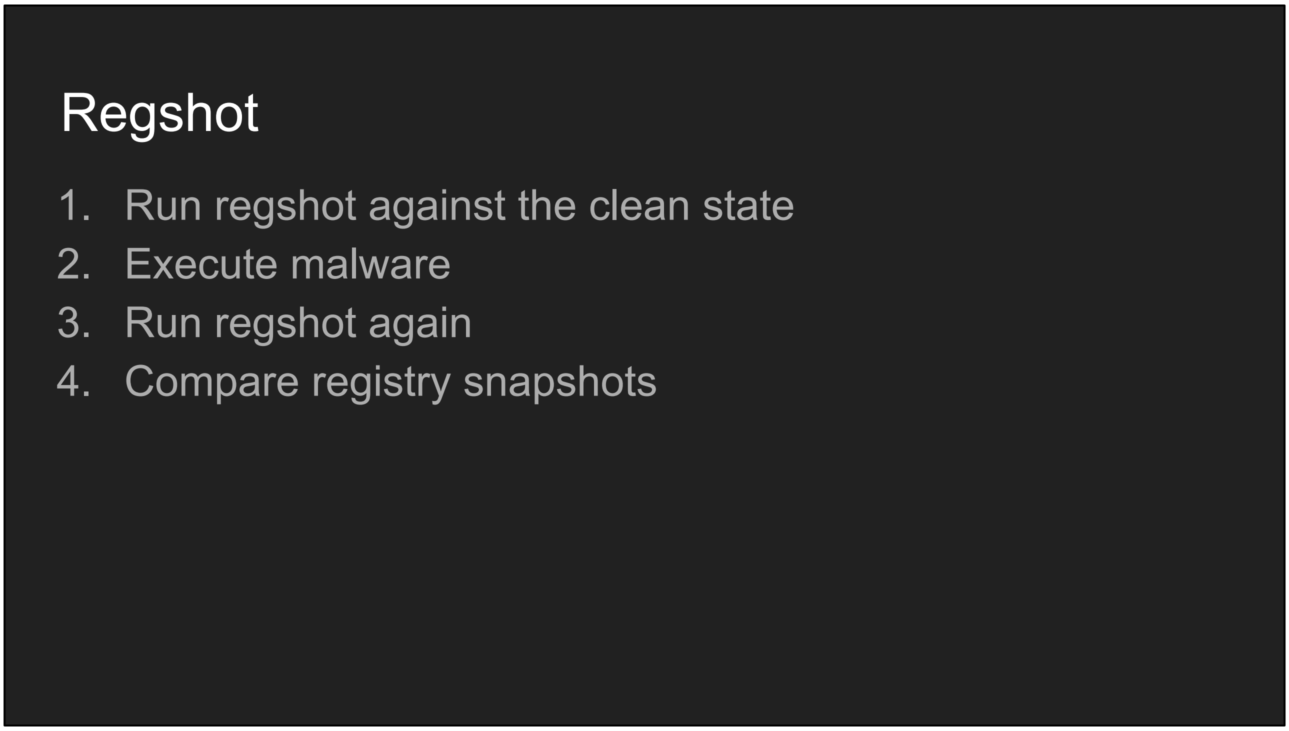
Process Explorer

- Launch Dependency Walker
- Search for DLLs / file handles
  - Compare loaded DLLs to imports in Dependency Walker

Watch for new processes that are launched after executing malware or opening malicious documents

Right-clicking on a process will give you the option to launch Dependency Walker against it
You can also Find > File handle or DLL to see DLLs loaded by any running process
    Comparing the DLL lists may reveal DLLs added to a process after load time

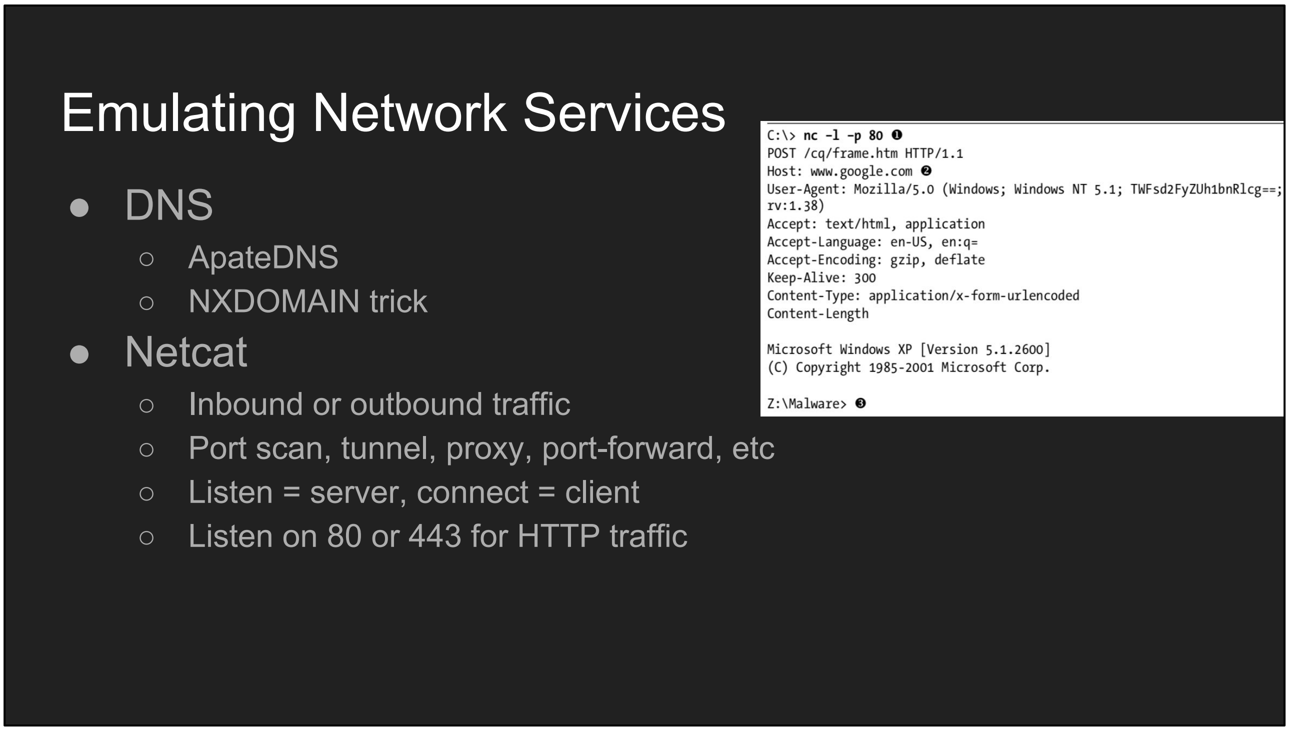
Watch for new processes that are launched after executing malware or opening malicious documents



Regshot

1. Run regshot against the clean state
2. Execute malware
3. Run regshot again
4. Compare registry snapshots

Regshot is a tool that can be used to compare snapshots of the Windows registry before and after executing malware in order to observe any changes to the registry



Emulating Network Services

- DNS
  - ApateDNS
  - NXDOMAIN trick
- Netcat
  - Inbound or outbound traffic
  - Port scan, tunnel, proxy, port-forward, etc
  - Listen = server, connect = client
  - Listen on 80 or 443 for HTTP traffic

```
C:\> nc -l -p 80 ❶
POST /cq/frame.htm HTTP/1.1
Host: www.google.com ❷
User-Agent: Mozilla/5.0 (Windows; Windows NT 5.1; TWFsd2FyZUh1bnRlcg==;
rv:1.38)
Accept: text/html, application
Accept-Language: en-US, en:q=
Accept-Encoding: gzip, deflate
Keep-Alive: 300
Content-Type: application/x-form-urlencoded
Content-Length

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

Z:\Malware> ❸
```

In order to allow malware to attempt to contact command and control infrastructure, we must simulate common network protocols for the malware to interact with.

Starting with DNS, ApateDNS is a tool that listens on UDP port 53 and answers any DNS queries with a configured IP address, and logs all queries it receives - handy for gathering network indicators of compromise

Because malware authors usually roll their command and control infrastructure many times to avoid being caught (or to remediate being caught), malware may be configured to check for multiple domains if the first one it sends a DNS query for has been compromised. In order to simulate this, ApateDNS can be configured to send back NXDOMAIN responses to queries to see if the malware tries another domain.

Netcat can be used to listen for HTTP connections (typically on 80 or 443 since those are commonly allowed through firewalls) and will output any received data to stdout

InetSim

- Linux tool (free)
- Simulates a lot of services
- Configurable responses
- Logs connections
- Dummy service

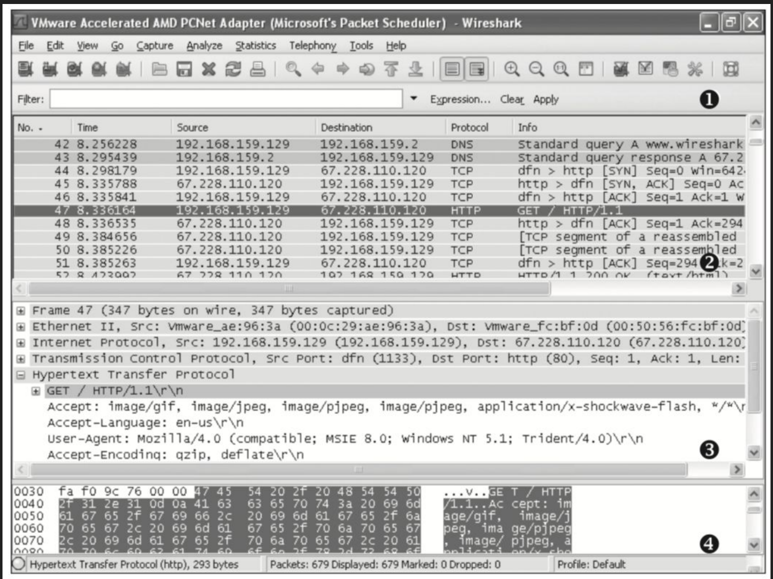
InetSim is a great tool for hosting on a networked Linux virtual machine to simulate a multitude of network services

Responses can be configured to return resources expected by malware

InetSim can log every connection made for analysis and even provides a dummy service to log all traffic sent by a client computer, regardless of port or protocol

Wireshark

- Protocol analyzer
- Follow TCP stream


VMware Accelerated AMD PCNet Adapter (Microsoft's Packet Scheduler) - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply
1
2
3
4

```
No. -  Time      Source           Destination      Protocol  Info
   42 8.256228   192.168.159.129  192.168.159.2    DNS       Standard query A www.wireshark
   43 8.295439   192.168.159.2    192.168.159.129  DNS       Standard query response A 67.2
   44 8.298179   192.168.159.129  67.228.110.120   TCP       dfn > http [SYN] Seq=0 Win=642
   45 8.335788   67.228.110.120   192.168.159.129  TCP       http > dfn [SYN, ACK] Seq=0 Ac
   46 8.335841   192.168.159.129  67.228.110.120   TCP       dfn > http [ACK] Seq=1 Ack=1 W
   47 8.336164   192.168.159.129  67.228.110.120   HTTP      GET / HTTP/1.1
   48 8.336535   67.228.110.120   192.168.159.129  TCP       http > dfn [ACK] Seq=1 Ack=294
   49 8.384656   67.228.110.120   192.168.159.129  TCP       [TCP segment of a reassembled
   50 8.385226   67.228.110.120   192.168.159.129  TCP       [TCP segment of a reassembled
   51 8.385263   192.168.159.129  67.228.110.120   TCP       dfn > http [ACK] Seq=294 Ack=2
   52 8.423992   67.228.110.120   192.168.159.129  HTTP      HTTP/1.1 200 OK  (text/html)
```

```
⊞ Frame 47 (347 bytes on wire, 347 bytes captured)
⊞ Ethernet II, Src: Vmware_ae:96:3a (00:0c:29:ae:96:3a), Dst: Vmware_fc:bf:0d (00:50:56:fc:bf:0d)
⊞ Internet Protocol, Src: 192.168.159.129 (192.168.159.129), Dst: 67.228.110.120 (67.228.110.120)
⊞ Transmission Control Protocol, Src Port: dfn (1133), Dst Port: http (80), Seq: 1, Ack: 1, Len:
⊟ Hypertext Transfer Protocol
  ⊞ GET / HTTP/1.1\r\n
    Accept: image/gif, image/jpeg, image/pjpeg, image/pjpeg, application/x-shockwave-flash, */*\r
    Accept-Language: en-us\r\n
    User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)\r\n
    Accept-Encoding: gzip, deflate\r\n
```

```
0030  fa f0 9c 76 00 00 47 45  54 20 2f 20 48 54 54 50   ...v..GE T / HTTP
0040  2f 31 2e 31 0d 0a 41 63  63 65 70 74 3a 20 69 6d   /1.1..Ac cept: im
0050  61 67 65 2f 67 69 66 2c  20 69 6d 61 67 65 2f 6a   age/gif,  image/j
0060  70 65 67 2c 20 69 6d 61  67 65 2f 70 6a 70 65 67   peg, ima ge/pjpeg
0070  2c 20 69 6d 61 67 65 2f  70 6a 70 65 67 2c 20 61   , image/ pjpeg, a
```

```
Hypertext Transfer Protocol (http), 293 bytes | Packets: 679 Displayed: 679 Marked: 0 Dropped: 0 | Profile: Default
```

Figure 3-10: Wireshark DNS and HTTP example

In order to get a full view of all network traffic on a system, we can use Wireshark to intercept and log packets
Start by choosing a network interface to capture on, letting the capture run for a bit, and then stopping it to analyze the captured packets
Filters are helpful for sifting through a lot of packets
Following a specific TCP stream is useful for viewing a single conversation between two hosts

Workflow

1. Set up network simulation
2. Start all monitoring tools
3. Execute malware
4. Stop tools and analyze output

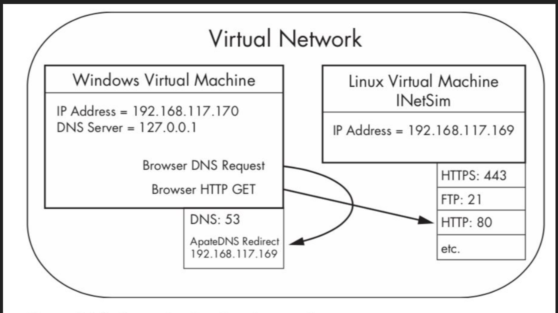


Figure 3-12: Example of a virtual network

Workflow:
Set up network simulation
Start all monitoring tools
Execute malware
Stop tools and analyze output

Resources

Hybrid Analysis provides a great example of a public sandbox

Process Monitor and Process Explorer from the Sysinternals suite for analyzing processes

Regshot for observing changes to the registry

ApateDNS, Netcat, and InetSim for simulating network services

Wireshark for packet capture and analysis