Cyber Security Technologies

Session 4 – Malware Analysis

Shawn Davis
ITMS 448 – Spring 2016

Slides contain original content from Davis, S. as well as from Sikorski M. and Honig A., Practical Malware Analysis; No Starch Press, Inc. 2012

Overview

- This week's lecture will consist of labs we will walk through together analyzing real malware from the wild
- Therefore, do NOT perform any of the labs in this slide deck on your RADISH VM or your personal or work computers.
- Go ahead and logon to your Win 8.1 physical workstation in the lab.
- Do not logon to RADISH

OVA

- We will not be using RADISH in class today
- We will be using an OVA file to import a Windows XP VM into VirtualBox on your physical desktop
- Make sure the following share is mapped to your lab computer:
  - \\coulson.otsads.iit.edu\itm448
- Copy the Win XP 32 Bit SP3 x48-1.ova file to your desktop

Note

- If you missed class and would like to perform the labs in this slide deck you will need to stop by the TS2033 lab during open hours. The itm448 share can be mapped there as well
- Then, only use VirtualBox and do not connect to RADISH to run through these labs

Overview

Part I – Creating an Isolated Analysis Environment

Part II – Basic Static Analysis

Part III – Advanced Static Analysis

Part IV – Basic Dynamic Analysis

Part V – Advanced Static Analysis

Definition of Malware Analysis

"The art of dissecting malware to understand how it works, how to identify it, and how to defeat or eliminate it."

(Sikorski and Honig, 2012, p. xxxviii)

Malware Analysis Techniques

- Static Analysis
  - Evaluate the executable without executing it.
- Dynamic Analysis
  - Evaluate the executable during execution.

Part I

Creating an Isolated Analysis Environment

Isolated Analysis Environment

- Especially needed for Dynamic Analysis
- Should also be used for Static Analysis in case of accidental execution

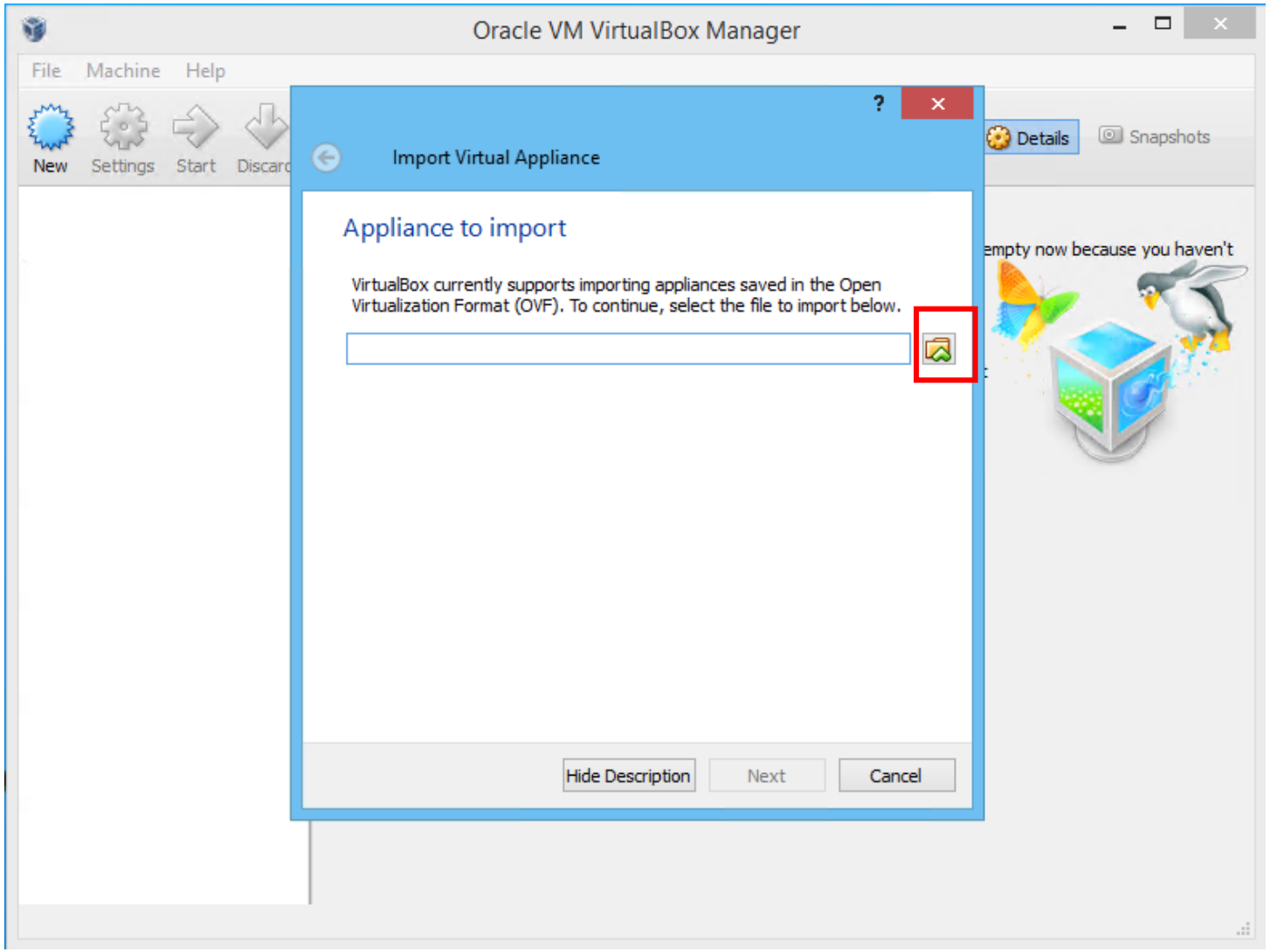
Creating an Isolated Analysis Environment

1. Create virtual machine(s)
   - May need multiple OSs since malware may behave differently in each
2. Isolate VM from host and/or internet.
3. Install static and dynamic analysis tools
4. Create a snapshot of the state of the VM at this point
   - Can revert to this state between tests

1. Create Virtual Machine

- We already have an XP VM ready for you to import.
- Open VirtualBox on your physical machine's desktop.
- Click "File" and "Import Appliance"

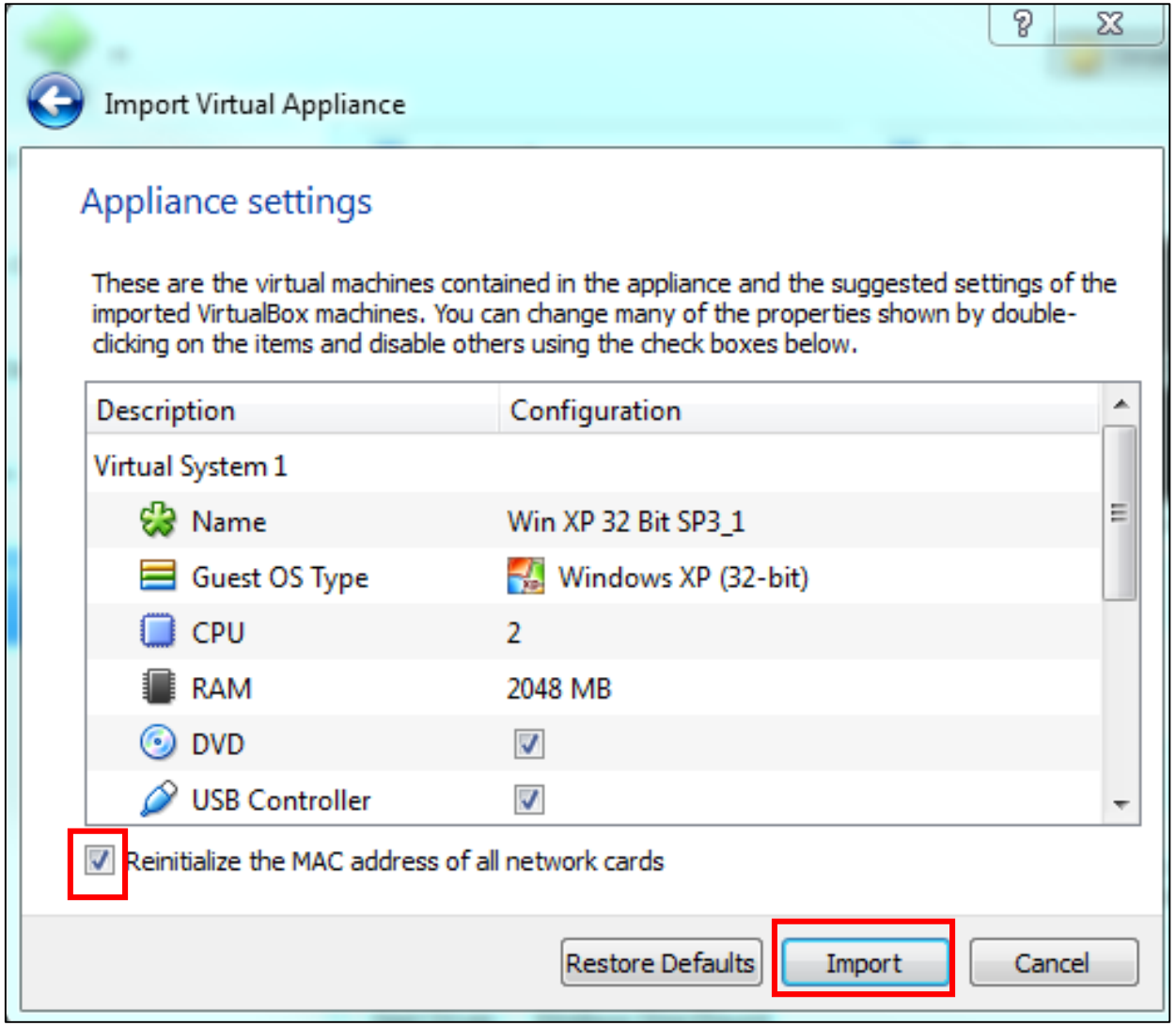
Import OVA


Oracle VM VirtualBox Manager
File
Machine
Help
New
Settings
Start
Discard
Details
Snapshots
Import Virtual Appliance
Appliance to import
VirtualBox currently supports importing appliances saved in the Open Virtualization Format (OVF). To continue, select the file to import below.
Hide Description
Next
Cancel
empty now because you haven't

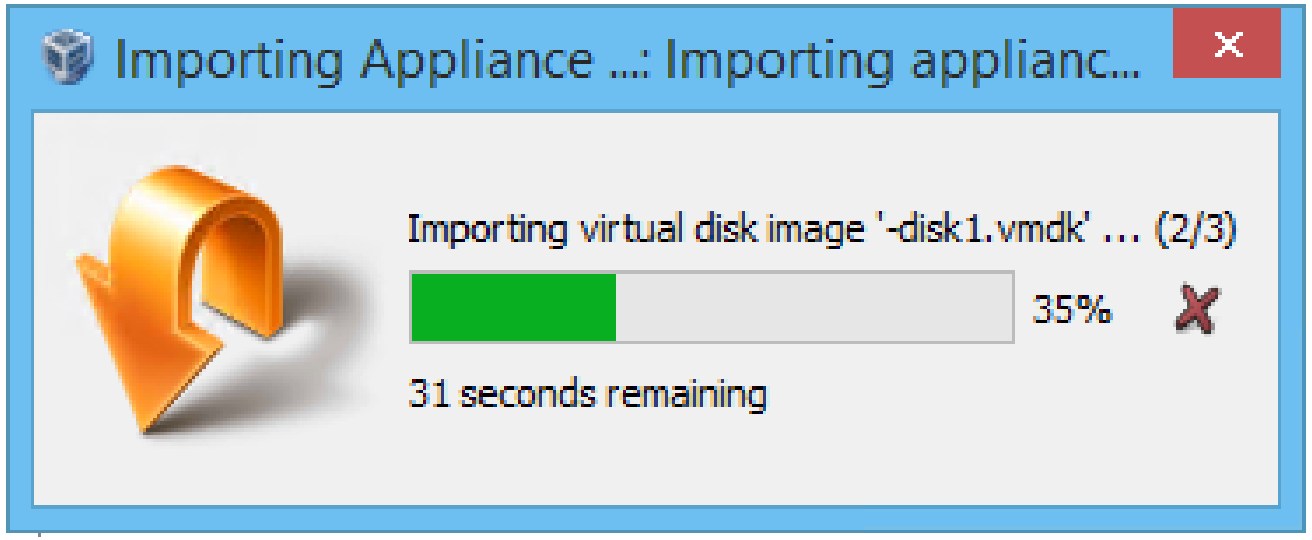
Import OVA

- Select the XP OVA file you copied to your desktop from the \\coulson.otsads.iit.edu\itm448 share in the beginning of class
- Hit "Open"
- Hit "Next"

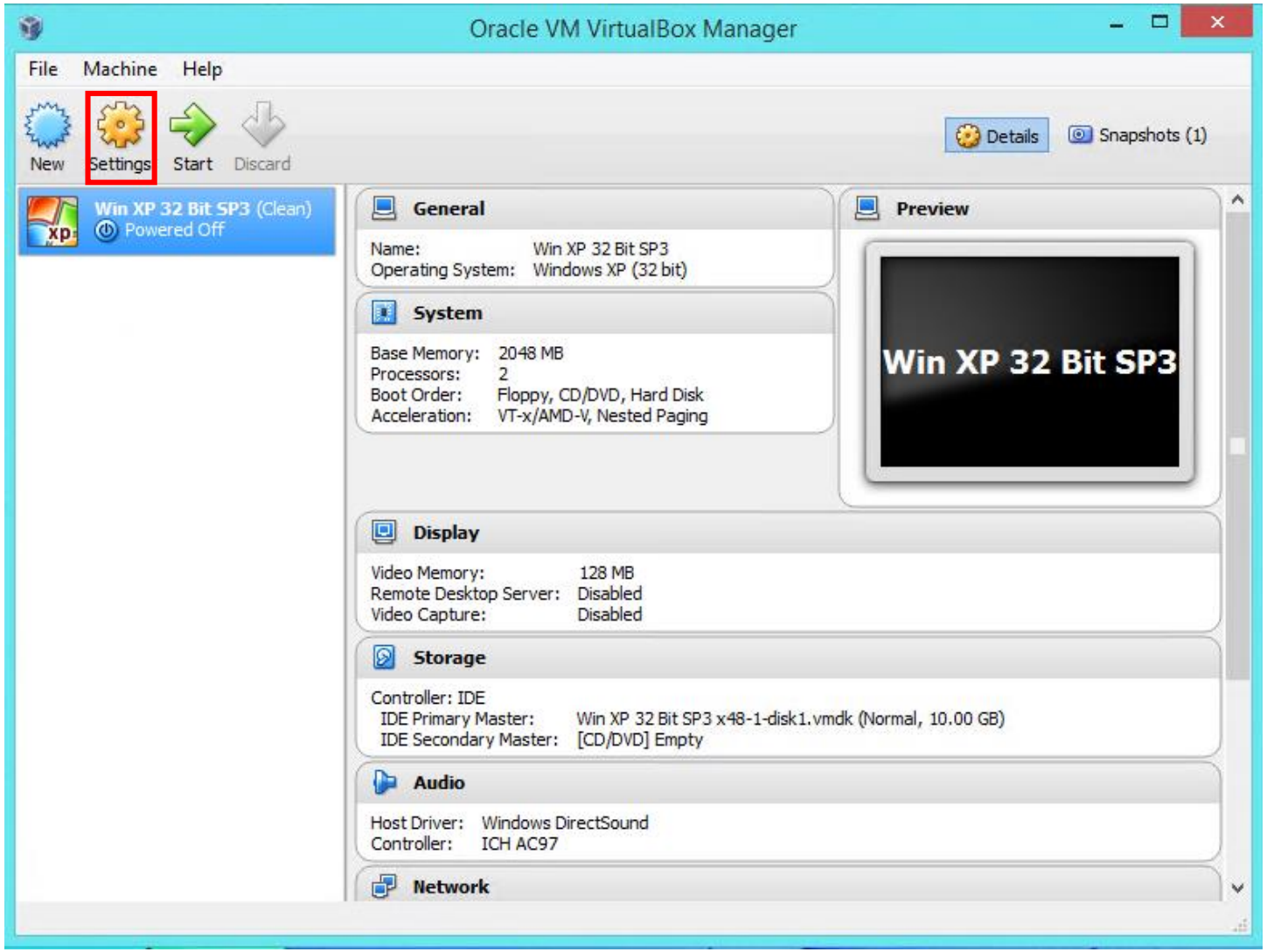
Import OVA


Import Virtual Appliance
Appliance settings
These are the virtual machines contained in the appliance and the suggested settings of the imported VirtualBox machines. You can change many of the properties shown by double-clicking on the items and disable others using the check boxes below.
Description
Configuration
Virtual System 1
Name
Win XP 32 Bit SP3_1
Guest OS Type
Windows XP (32-bit)
CPU
2
RAM
2048 MB
DVD
USB Controller
Reinitialize the MAC address of all network cards
Restore Defaults
Import
Cancel

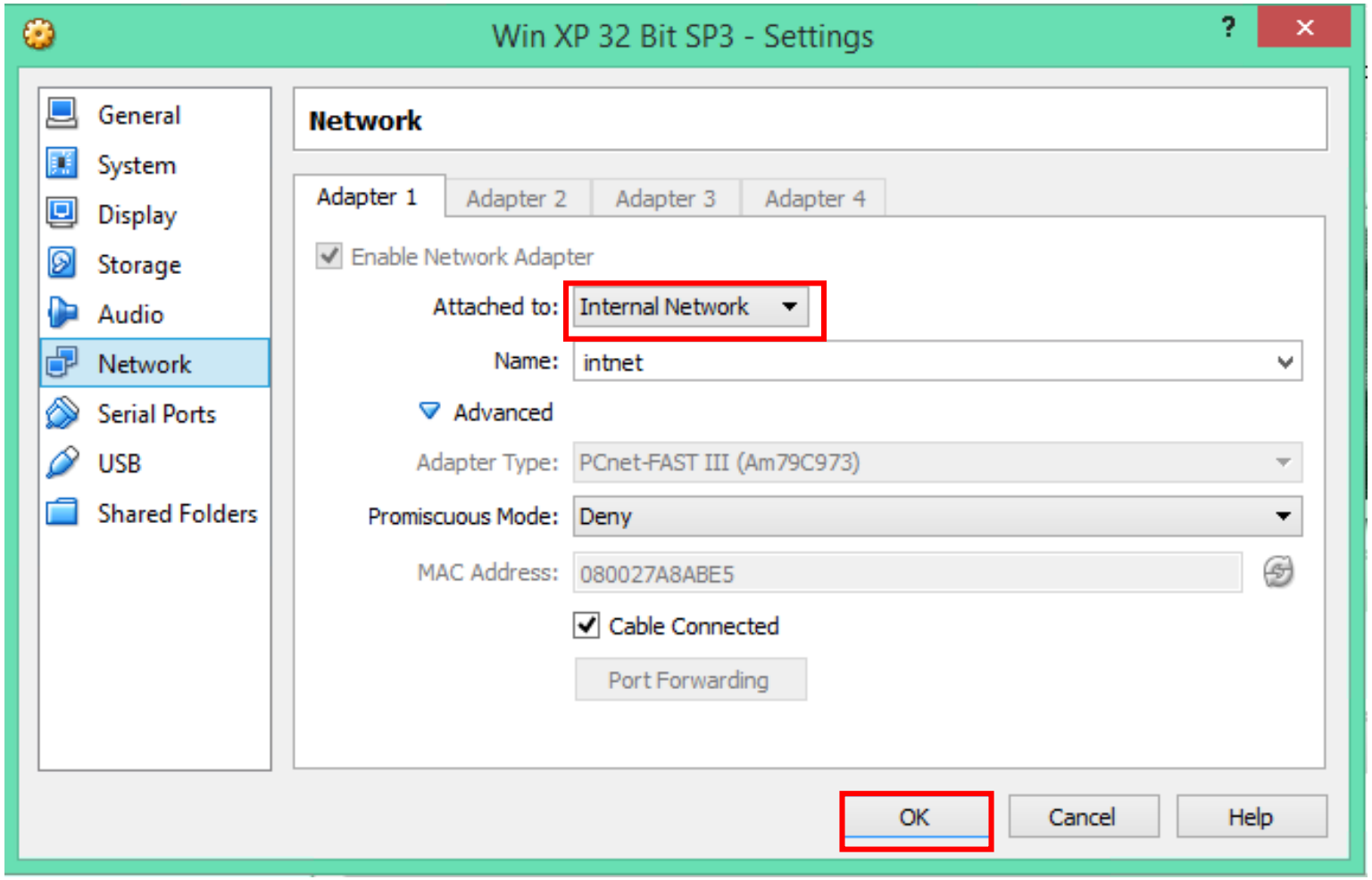
Import OVA


Importing Appliance ...: Importing applianc...
Importing virtual disk image '-disk1.vmdk' ... (2/3)
35%
31 seconds remaining

Import OVA


Oracle VM VirtualBox Manager
File
Machine
Help
New
Settings
Start
Discard
Details
Snapshots (1)
Win XP 32 Bit SP3 (Clean)
Powered Off
General
Name: Win XP 32 Bit SP3
Operating System: Windows XP (32 bit)
System
Base Memory: 2048 MB
Processors: 2
Boot Order: Floppy, CD/DVD, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging
Preview
Win XP 32 Bit SP3
Display
Video Memory: 128 MB
Remote Desktop Server: Disabled
Video Capture: Disabled
Storage
Controller: IDE
IDE Primary Master: Win XP 32 Bit SP3 x48-1-disk1.vmdk (Normal, 10.00 GB)
IDE Secondary Master: [CD/DVD] Empty
Audio
Host Driver: Windows DirectSound
Controller: ICH AC97
Network

Import OVA


Win XP 32 Bit SP3 - Settings
General
System
Display
Storage
Audio
Network
Serial Ports
USB
Shared Folders
Network
Adapter 1
Adapter 2
Adapter 3
Adapter 4
Enable Network Adapter
Attached to:
Internal Network
Name:
intnet
Advanced
Adapter Type:
PCnet-FAST III (Am79C973)
Promiscuous Mode:
Deny
MAC Address:
080027A8ABE5
Cable Connected
Port Forwarding
OK
Cancel
Help

2. Isolate VM from host and/or Internet

- Let's review some of the VirtualBox network options that are in the "Attached to" dropdown.

VirtualBox Network Modes (Internet)

- Bridged Adapter
  - Physical Host and VM are on same network and can connect to each other
  - VM can reach internet
- NAT
  - Physical Host and a single VM are on different networks
  - Physical Host cannot initiate connection to VM but VM can initiate connection to Physical Host.
  - VM can reach internet

VirtualBox Network Modes (Internet)

- NAT Network
  - Same as NAT but multiple VMs can share the same network.
  - VM can reach internet

VirtualBox Network Modes (No internet)

- Host Only Adapter
  - Physical Host and VM(s) are on same private network
  - VM(s) cannot reach internet
- Internal Network
  - Physical Host and VM(s) are on different networks
  - No DHCP (would need to assign static IPs)
  - VM cannot reach internet
- Not Attached
  - Completed disconnected

VirtualBox Network Modes Conclusion

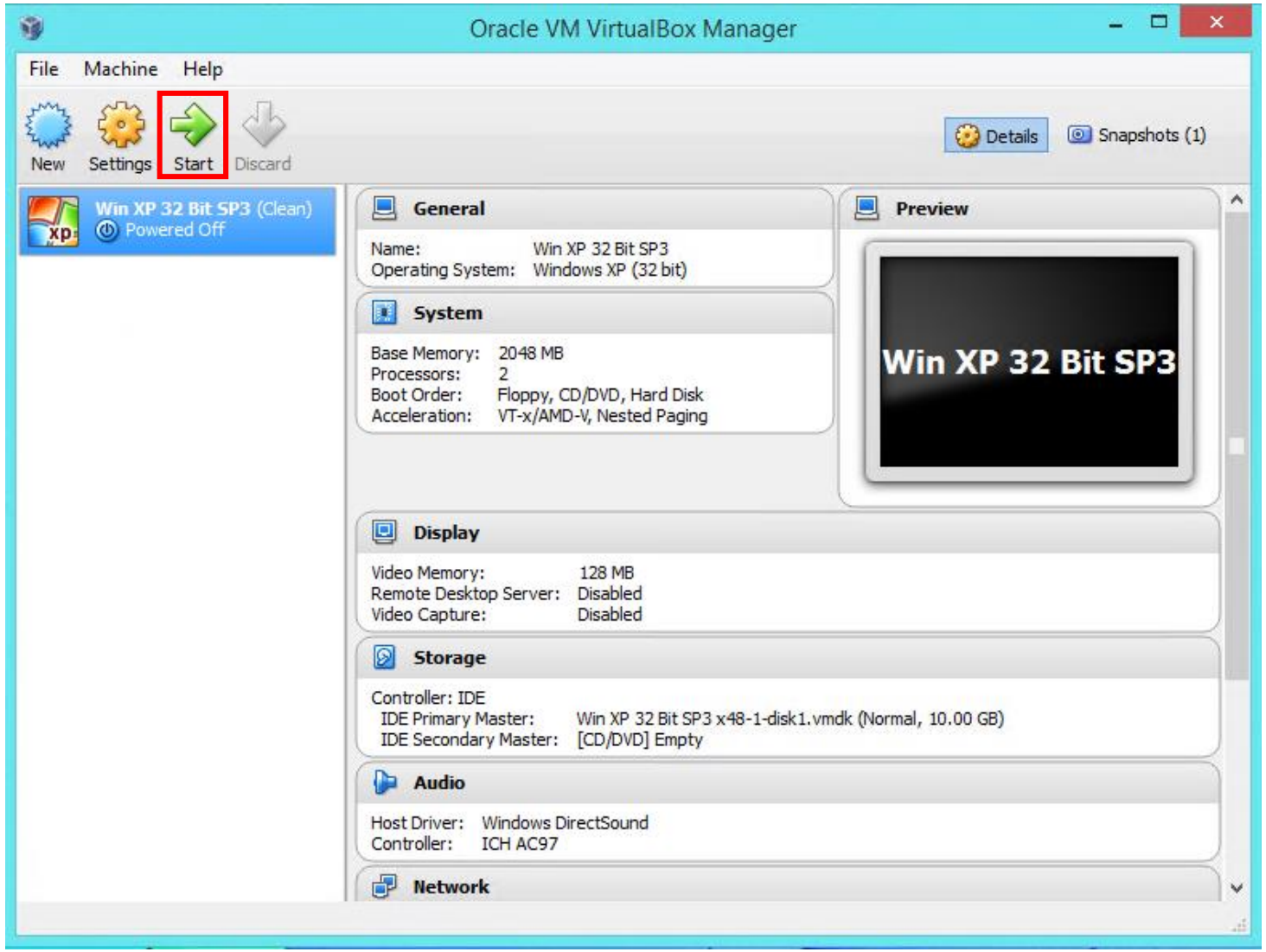
- Could infect your entire LAN:
  - Bridged Adapter
- Could infect your Physical Host:
  - Use NAT, NAT, Network, Host Only Adapter
- Will not infect your Physical Host or LAN:
  - Internal Network, Not Attached

Best Solution

- If you need to reach the internet and not infect your Physical Host:
  - Install a second network adapter and lease a separate internet connection (dirty pipe)

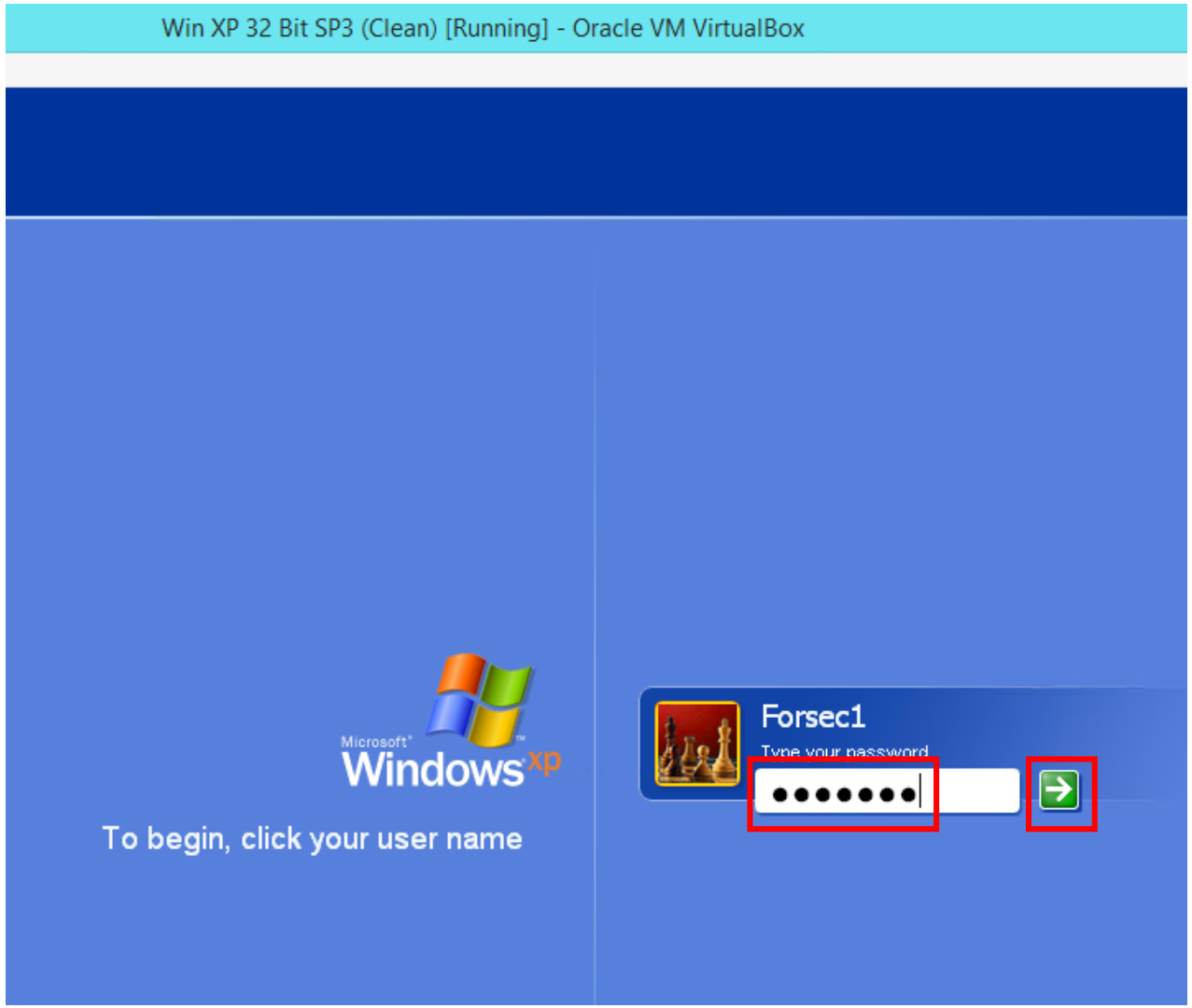
(In here, we will just use the “Internal Network” for our malware analysis labs)

Start XP VM


Oracle VM VirtualBox Manager
File
Machine
Help
New
Settings
Start
Discard
Details
Snapshots (1)
Win XP 32 Bit SP3 (Clean)
Powered Off
General
Name: Win XP 32 Bit SP3
Operating System: Windows XP (32 bit)
System
Base Memory: 2048 MB
Processors: 2
Boot Order: Floppy, CD/DVD, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging
Preview
Win XP 32 Bit SP3
Display
Video Memory: 128 MB
Remote Desktop Server: Disabled
Video Capture: Disabled
Storage
Controller: IDE
IDE Primary Master: Win XP 32 Bit SP3 x48-1-disk1.vmdk (Normal, 10.00 GB)
IDE Secondary Master: [CD/DVD] Empty
Audio
Host Driver: Windows DirectSound
Controller: ICH AC97
Network

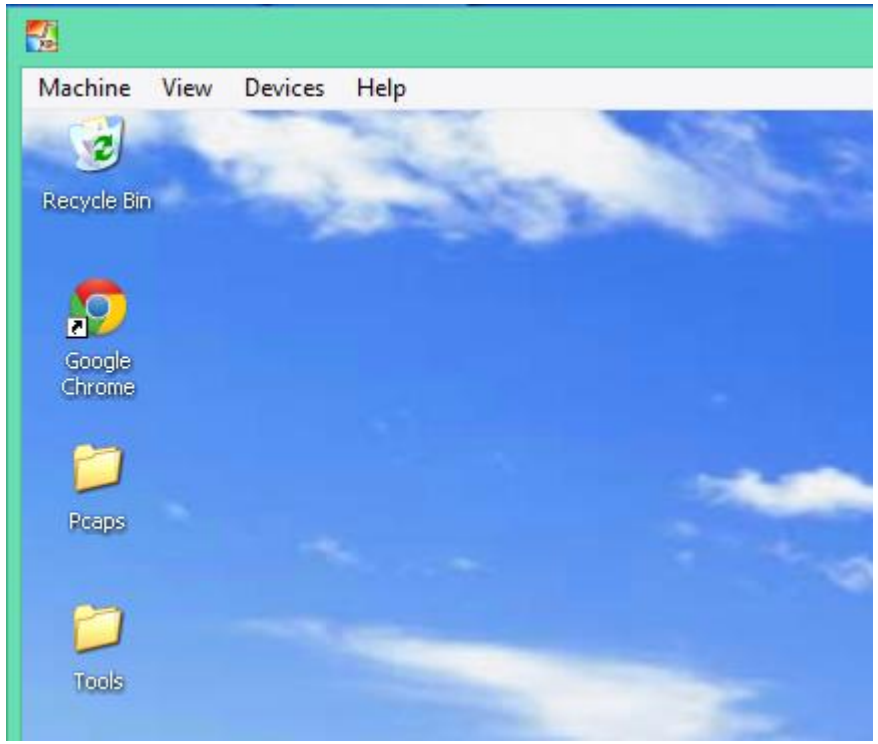
Start XP VM

Password = Forsec1

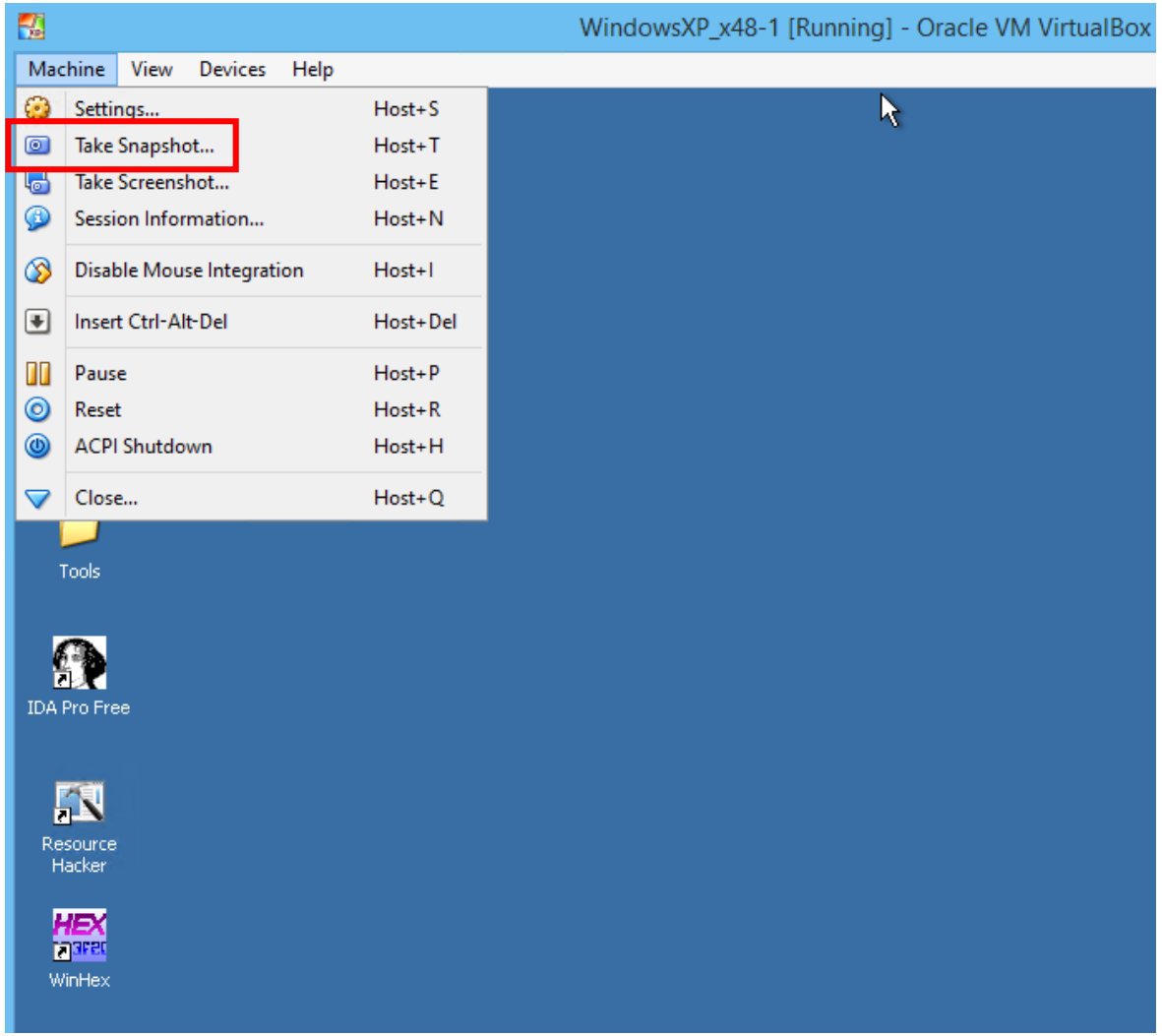


3. Install Static and Dynamic Analysis Tools

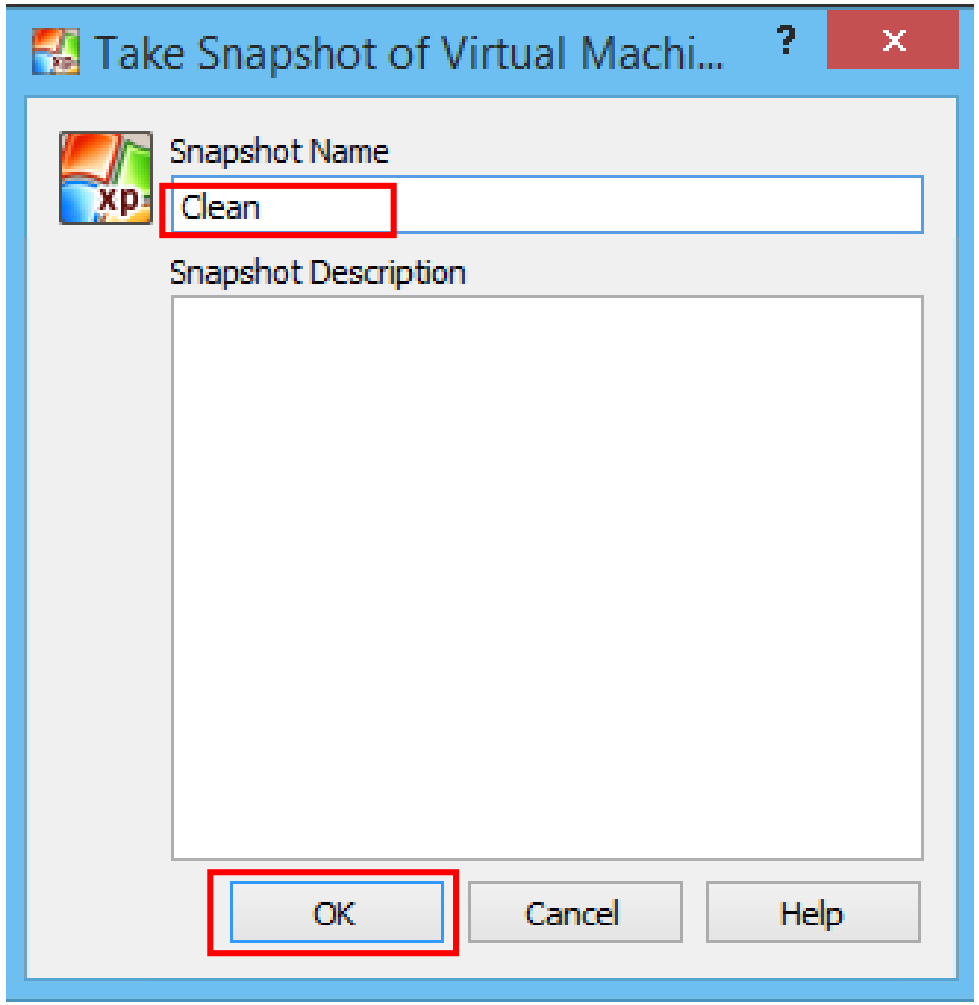
- Tools already placed in Tools folder on desktop
- Pcaps – Crimeboss Pcap
  - Will use for in class lab



4. Create a snapshot of the state of the VM at this point



Create a snapshot of the state of the VM at this point (Cont.)

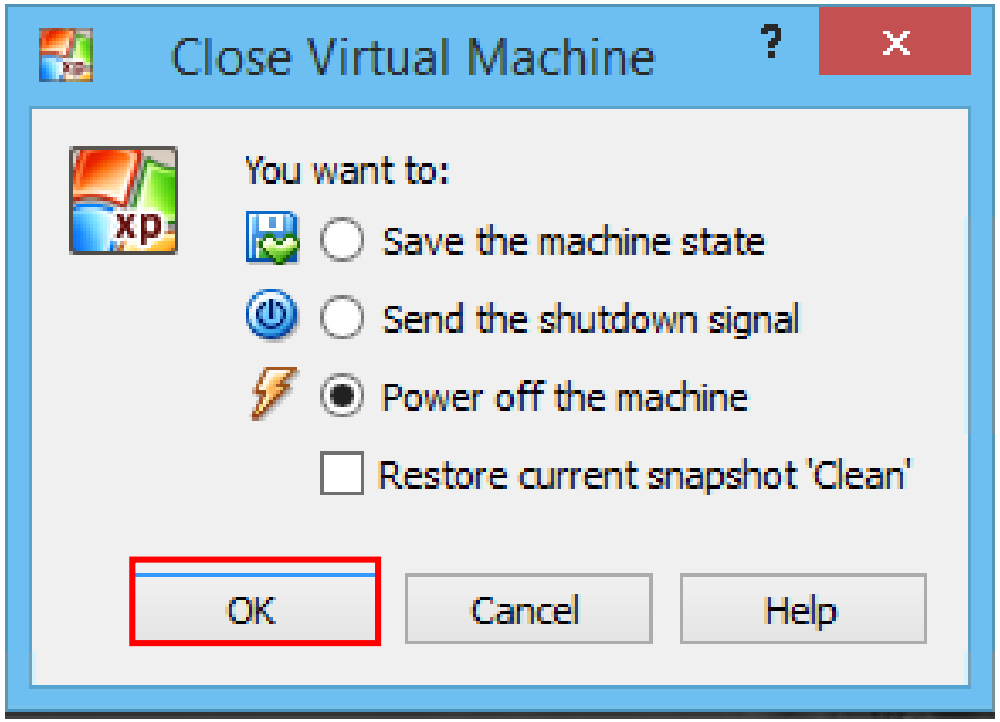


Clean Snapshot

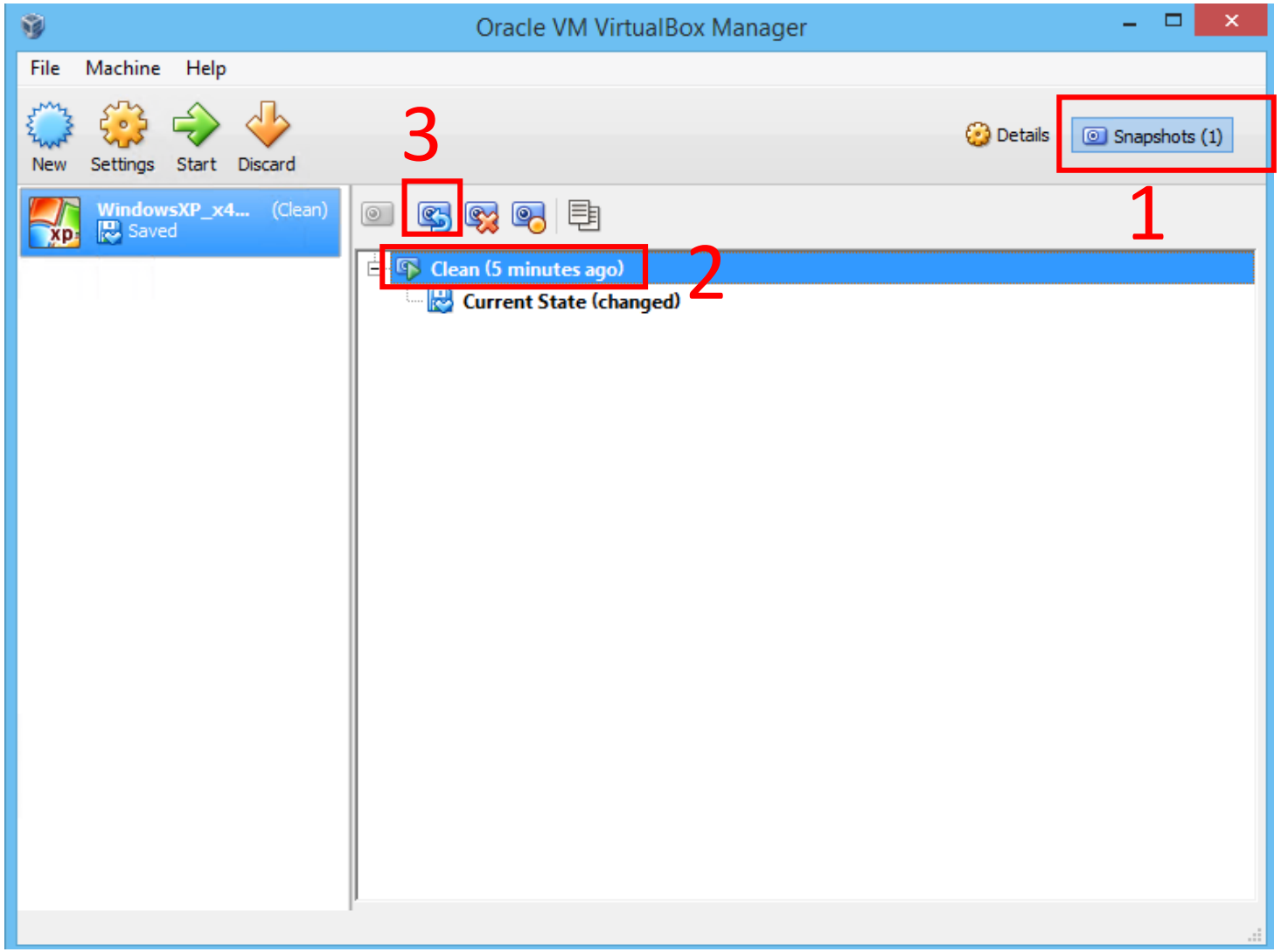
- Now, you can revert back to a clean state after running Malware.
- Let's practice reverting.
- First, right click properties and change the desktop background to something else.

Reverting to a Clean Snapshot

- Click the Red X in the top right cover of the XP VM.

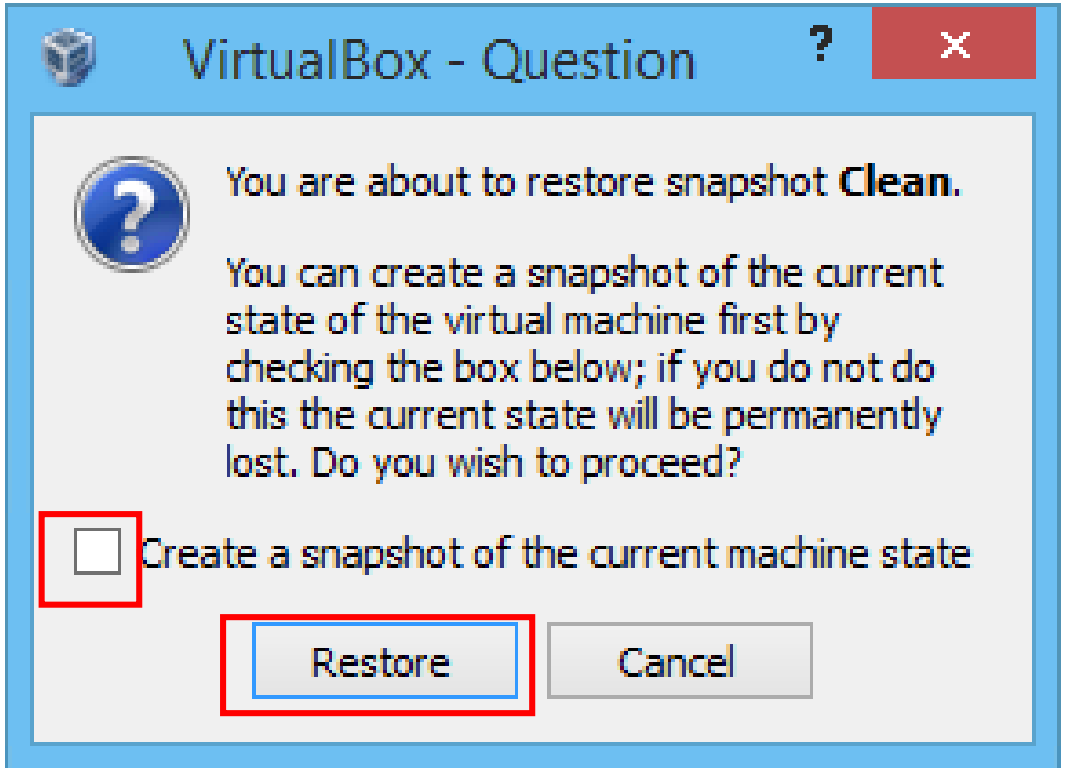


Reverting to a Clean Snapshot

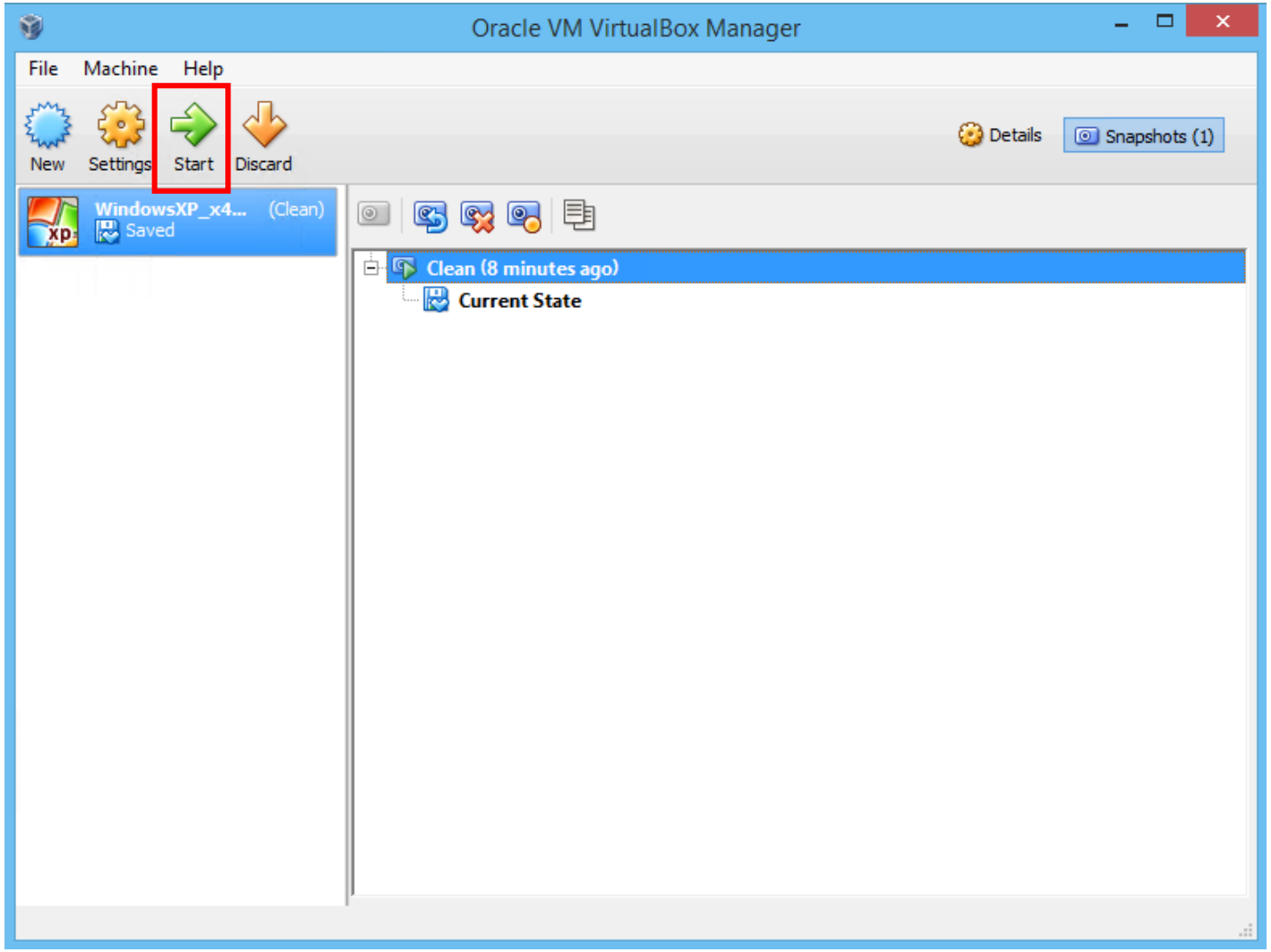


Reverting to a Clean Snapshot

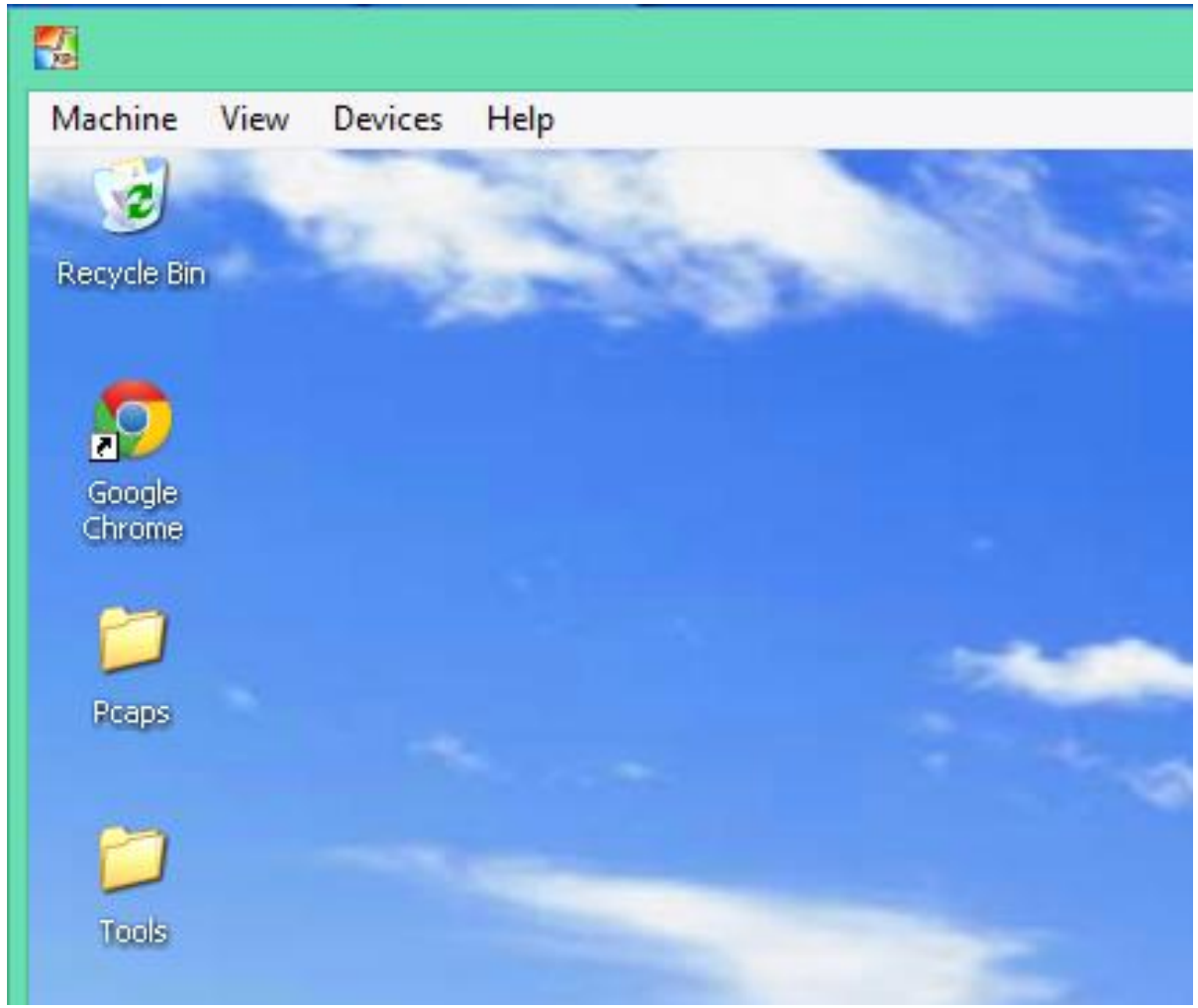
Uncheck Box



Reverting to a Clean Snapshot



Back to Clean Snapshot



Isolated Environment

Part II

Basic Static Analysis

Static Analysis

- Basic Static Analysis
  - Examine the malicious executable file
  - Not viewing actual code instructions inside
  - Not executing the file.
- Advanced Static Analysis
  - Load the malicious executable file into a disassembler (such as IDA)
  - Determine what the malware does by examining the code instructions of the executable file
  - Not executing the file.

(Sikorski and Honig, 2012, pp. 2-3)

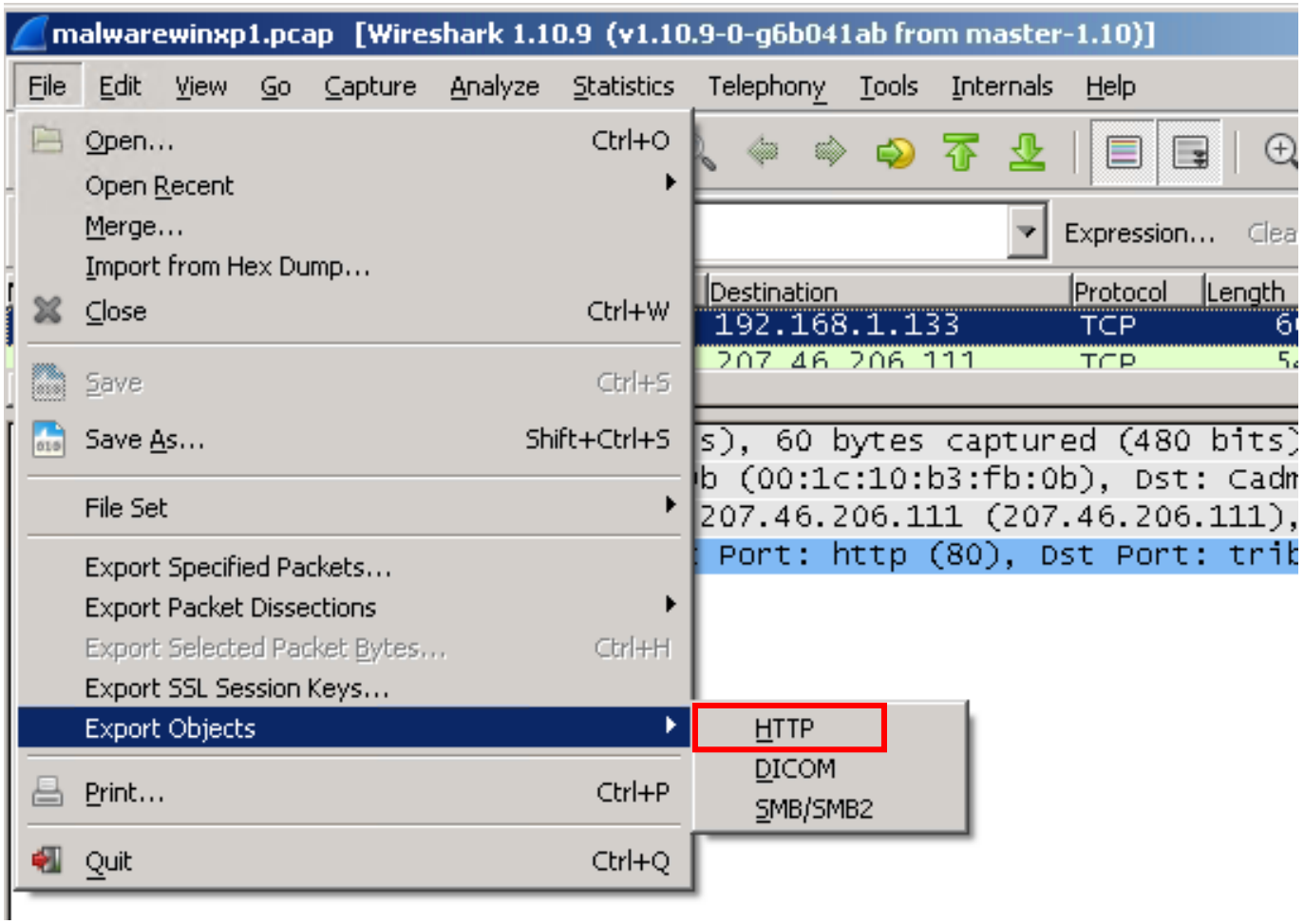
Laine.lora

- In last week's lab we noticed that the Crimeboss exploit kit downloader Trojan grabbed a large file called Laine.lora.

- Go ahead and open the Pcaps folder on the desktop.
- Now, double-click on the malwarewinxp1.pcap file to open it in Wireshark

Laine.lora

- File / Export Objects / HTTP



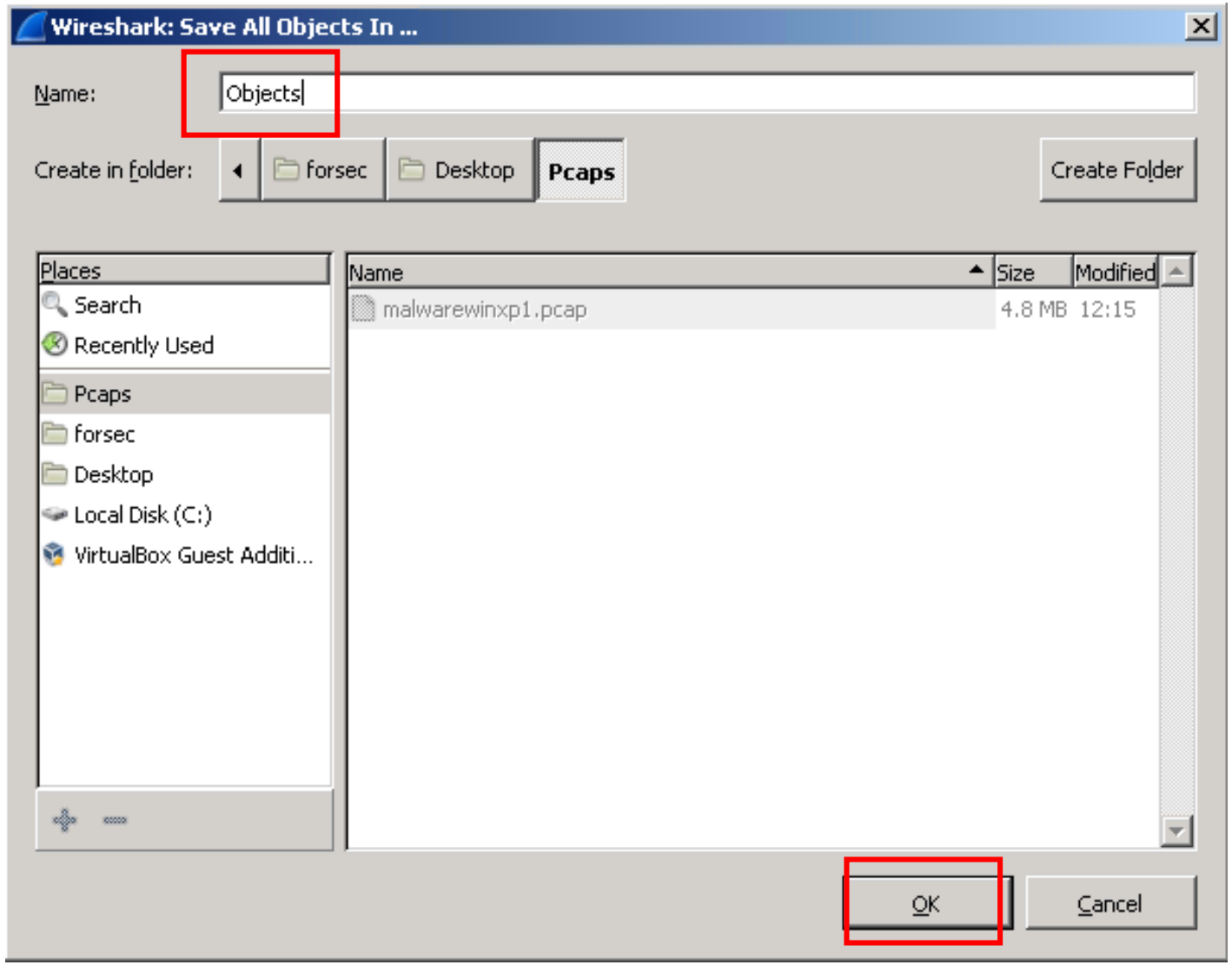
Laine.lora

Wireshark: HTTP object list

Packet num	Hostname	Content Type	Size	Filename
14	abrahamspath.org.uk	text/html	236 bytes	cb.php
19	ping1.unicast.com	image/gif	42 bytes	adstracking.gif?DV=3.90&PT=T&AD=597384&VD=0&TV=27&
21	udc.msn.com	image/gif	42 bytes	c.gif?parsergroup=hops&evt=unload&rid=4b3d23328c084d0
25	abrahamspath.org.uk	text/html	0 bytes	cb.php?action=jv&h=608299343
28	ble.com.gr	text/html	383 bytes	ads.js
37	criagift.com.br	text/html	243 bytes	index.php?setup=d
45	criagift.com.br	text/html	4602 bytes	index.php?setup=d&s=2&r=900013
54	abrahamspath.org.uk	image/vnd.microsoft.icon	0 bytes	favicon.ico
81	danieldelaney.com	application/java-archive	4679 bytes	amor1.jar
83	danieldelaney.com	application/java-archive	4822 bytes	java7.jar?r=463779
780	blogtcl.com.ar	application/x-msdownload	648 kB	rh.exe
5109	uploads.boxify.me		3730 kB	Laine.lora?1357412773
5145	tecnologiadesecundaria.com	application/x-www-form-urlencoded	86 bytes	index.php
5148	tecnologiadesecundaria.com	text/html	26 bytes	index.php

Help Save As Save All Cancel

Laine.lora


Wireshark: Save All Objects In ...
Name:
Objects
Create in folder:
forsec
Desktop
Pcaps
Create Folder
Places
Search
Recently Used
Pcaps
forsec
Desktop
Local Disk (C:)
VirtualBox Guest Additi...
Name
Size
Modified
malwarewinxp1.pcap
4.8 MB
12:15
OK
Cancel

Laine.lora

- Exit out of Wireshark and open the Pcaps\Objects Folder
- Do not execute any of these files!
  - Static Analysis does not involve execution.

What is the Laine.lora file???

- Open WinHex
- Drag Laine.lora file from Objects folder into Winhex
- Click OK on any warnings
- What does the header tell you about this file??

Laine.lora Header

Laine.lora%3f1357412773

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000000	50	4B	03	04	14	00	00	00	08	00	28	87	25	42	D4	C2	PK (‡%BÔÂ
00000010	9B	67	E6	D9	29	00	00	F2	3F	00	0C	00	00	00	4A	75	›gæÙ) ò? Ju
00000020	73	63	68	65	64	73	2E	65	78	65	DC	FD	0F	5C	54	55	scheds.exeÜý \TU
00000030	FA	07	8E	DF	B9	73	81	01	47	67	54	CC	3F	61	51	4D	ú ‡ß¹s‡ GgTÌ?aQM
00000040	A5	A9	05	61	A5	A2	85	C0	20	28	E8	08	03	E2	FF	48	¥© a¥¢…À (è âÿH
00000050	66	04	43	60	87	3B	FE	29	FF	C0	0E	7C	F2	7A	63	97	f C`‡;þ)ÿÀ |òzc—
00000060	DA	DA	F5	B3	B9	9F	D5	AD	FD	6C	DB	B6	E5	6E	6E	9A	ÚÚõ³¹ŸÕ-ýlÛ¶ånn›
00000070	B9	35	38	C6	1F	B5	04	75	13	FF	94	A4	54	17	C7	8A	¹58Æ µ u ÿ”¤T Ç‰
00000080	94	04	95	B8	BF	E7	39	E7	CC	30	20	66	BB	7D	7E	AF	” •¸¿ç9çÌ0 f»}~¯
00000090	EF	EF	FB	B3	86	F7	BD	CF	39	E7	39	CF	79	CE	39	CF	ïïû³†÷½Ï9ç9ÏyÎ9Ï
000000A0	79	CE	9F	7B	6F	DA	02	0B	C7	73	1C	27	70	06	4E	55	yÎŸ{oÚ Çs 'p NU
000000B0	39	6E	37	47	FF	C5	71	A3	B8	9B	FE	D3	70	DC	1E	23	9n7GÿÅq£¸›þÓpÜ #
000000C0	37	E8	F6	9D	A1	1F	DE	B1	5B	93	FA	E1	1D	55	55	D6	7èö¡ ¡ Þ±[“úá UUÖ
000000D0	BC	FC	92	C8	62	47	D1	72	47	CE	CA	C8	95	CE	12	31	¼ü’ÈbGÑrGÎÊÈ•Î 1
000000E0	F2	09	5B	A4	C3	59	18	E9	2C	CC	B5	39	22	E7	E5	17	ò [¤ÃY é,Ìµ9"çå
000000F0	C6	3C	38	30	CC	F4	08	F7	FF	23	FF	2C	66	8E	4B	D5	Æ<80Ìô ÷ÿ#ÿ,f‡KÕ
00000100	84	71	2F	55	28	16	1F	AD	99	FB	45	D9	00	0D	3F	92	„q/U(–‡ûEÙ ?’
00000110	FB	62	04	C7	E9	EE	25	B4	E2	EF	E0	DA	88	57	23	E1	ûb Çéî%´âïàÚˆW#á

Laine.lora Strings

- Close WinHex.
- Open BinText
  - Desktop / Tools / BinText / BinText.exe
- Drag Laine.lora into BinText Window
- Scroll to bottom of BinText Window

Laine.lora Strings (Cont.)

BinText 3.0.3

Search | Filter | Help

File to scan: C:\Users\Shawn\Desktop\Laine.lora Browse Go

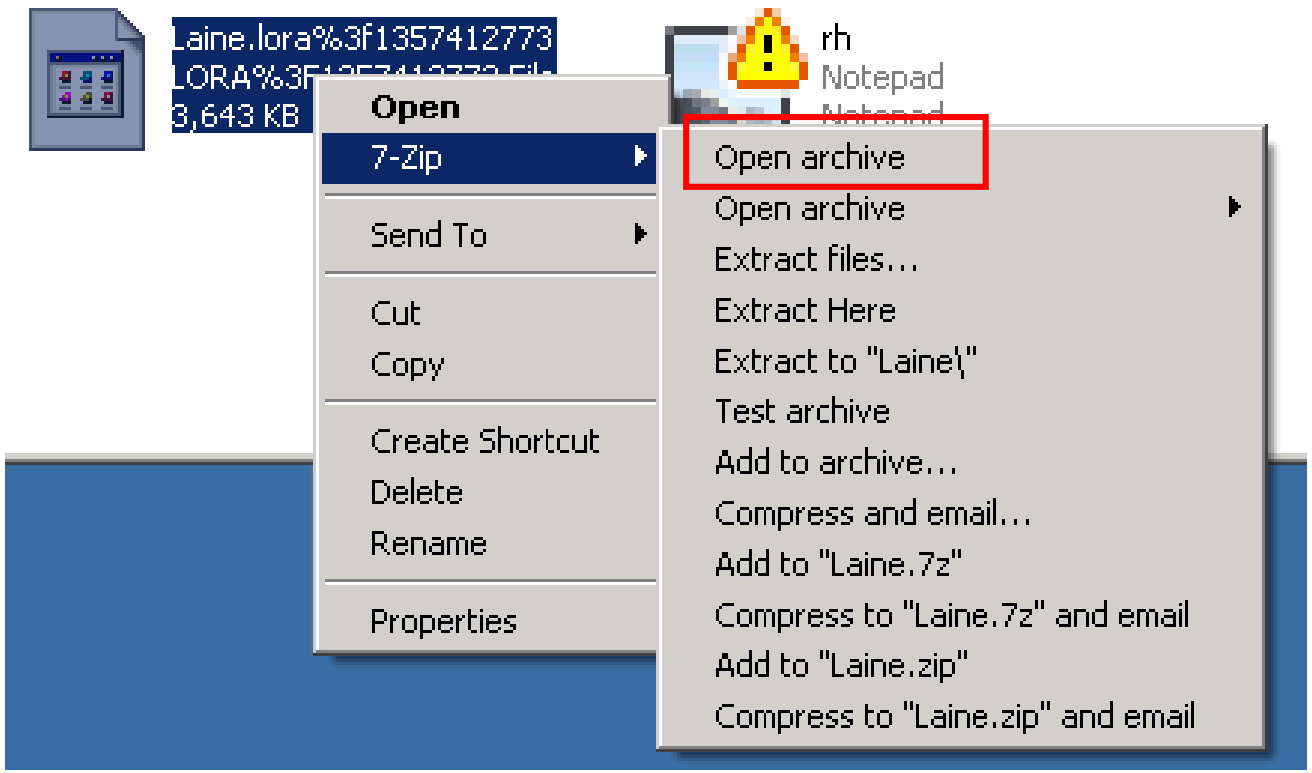
Advanced view Time taken : 1.326 secs Text size: 89249 bytes (87.16K)

File pos	Mem pos	ID	Text
A 00000038E656	00000038E656	0	7Q9S;
A 00000038E6B7	00000038E6B7	0	l:rSh?
A 00000038E786	00000038E786	0	#blJwMq!R
A 00000038E79E	00000038E79E	0	!09iF
A 00000038E7C2	00000038E7C2	0	$8vxJnFJ
A 00000038E8A4	00000038E8A4	0	eN$=E
A 00000038E984	00000038E984	0	*bBFd
A 00000038E98D	00000038E98D	0	V3LhRR
A 00000038E99C	00000038E99C	0	@F{jJf
A 00000038E9EA	00000038E9EA	0	LTT$=%
A 00000038E9F3	00000038E9F3	0	nb''#u_wZ
A 00000038EA50	00000038EA50	0	Juscheds.exe
A 00000038EAAE	00000038EAAE	0	WostWin.exe
A 00000038EB0B	00000038EB0B	0	javafx.exe

Ready | AN: 15882 | UN: 0 | RS: 0 | Find | Save

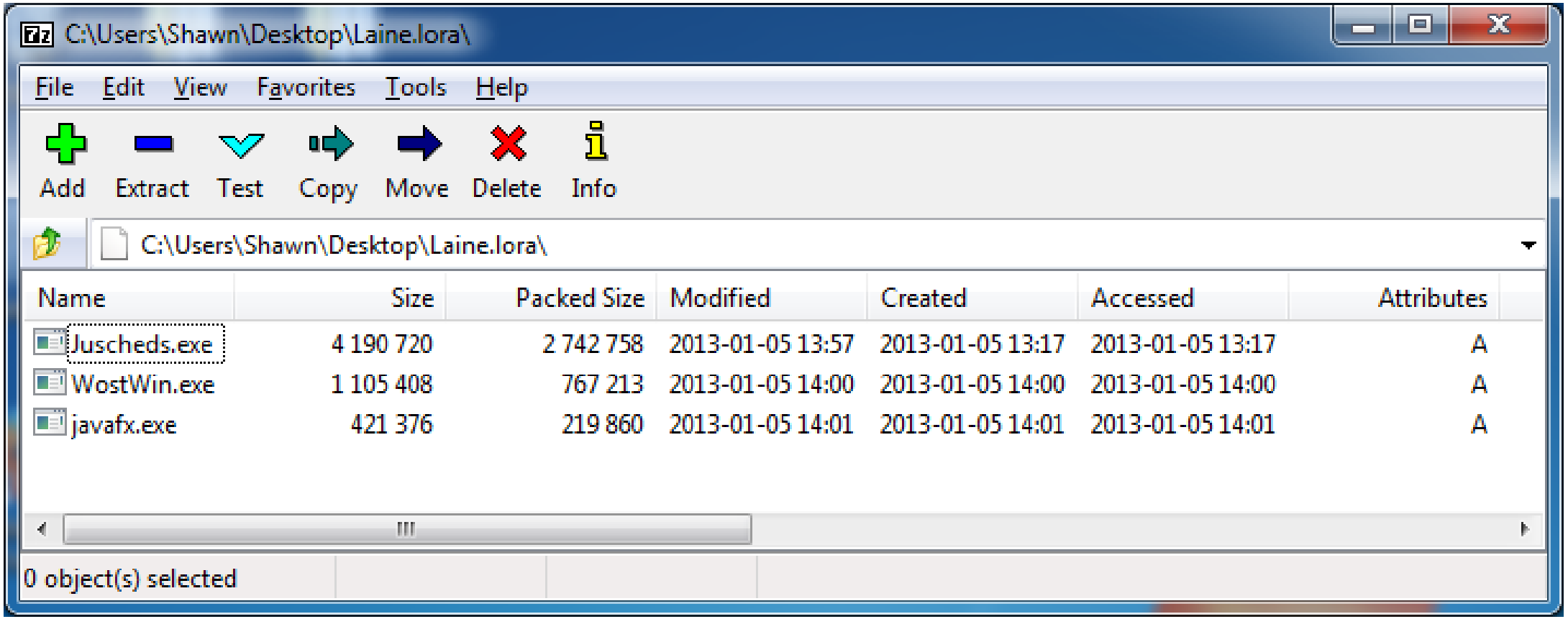
What is the Laine.lora file???

- Close BinText.
- In Objects folder, right click Laine.lora, select 7-Zip, Open archive



Laine.lora

- This file is really just a compressed archive.



Extract 3 Files from Laine.lora

- Select “Extract” in 7-Zip
- Keep default for Objects folder.
- Select “OK”
- Close out of 7-Zip
- Now in Objects folder we have:

Plan of Attack

- javafx.exe – We will walk through analyzing this file in the lecture now together.

Basic Static Analysis Tasks

1. Generate hash of executable
2. Search executable for interesting strings
3. View the executable in a hex editor to determine file type
4. Use PEiD to determine if executable is packed
5. Use PEview to determine compilation date
6. Use Resource Hacker to view resource section
7. Use Dependency Walker to determine functions that executable imports and exports

(Sikorski and Honig, 2012, pp. 10-26)

Hash with md5deep: javafx.exe

- Start / Run / cmd
- cd Desktop\tools\md5deep-4.3
- md5deep.exe ..\..\Pcaps\Objects\javafx.exe

```
C:\Documents and Settings\forsec\Desktop\Tools\md5deep-4.3>md5deep.exe ..\..\Pca
ps\Objects\javafx.exe
578b71ae5c129c9619b1614330115337  C:\Documents and Settings\forsec\Desktop\Pcaps
\Objects\javafx.exe
```

virustotal: javafx.exe

- Right click in terminal
- Select “Mark”
- Select hash and right click it to copy.
- Open Chrome on Win8.1 Physical Desktop
- Browse to www.virustotal.com
- Select “Search” tab
- Paste hash into text box.
- Select “Search it!”

virustotal: javafx.exe

File name: 578b71ae5c129c9619b1614330115337

Detection ratio: 26 / 46

Analysis date: 2013-02-18 00:45:14 UTC (1 year, 5 months ago)

Analysis | File detail | Additional information | Comments 0

Antivirus	Result
AVG	PSW.Banker6.ANMY
AntiVir	TR/Agent.421376.32
Avast	Win32:Delf-TFM [Trj]
BitDefender	Trojan.Generic.KDV.823444
CAT-QuickHeal	Trojan.Agent.gen
Comodo	UnclassifiedMalware
ESET-NOD32	Win32/Spy.Banker.XVL
F-Secure	Trojan.Generic.KDV.823444
Fortinet	W32/Banker.XVL!tr.spy

virustotal

- You just used the search feature to look for existing hashes and previous analysis
- You could also have just uploaded the malicious file for a new analysis

- Why wouldn't you want to do that???

- Alerts the attacker that you are on to them if they happen to be using a 0-day attack!

Interesting Strings: javafx.exe

- Drag javafx.exe into BinText
- Look through the strings.
- Which do you think are interesting???

Interesting Strings: javafx.exe

Text
C:\Program Files\
C:\Program Files (x86)
C:\Program Files (x86)\GbPlugin
C:\Windows\SysWOW64\drivers\
C:\Program Files (x86)\
C:\Program Files\GbPlugin
C:\Windows\System32\drivers\
C:\Arquivos de Programas\GbPlugin
C:\Arquivos de Programas\
cacls ''
km.sy
s'' /T /E /C /P SY
STEM:N

Text
shutdown
Service failed in custom message(%d): %s
Service installed successfully
Service ''%s'' failed to install with error: ''%s''
Service uninstalled successfully
Service ''%s'' failed to uninstall with error: ''%s''
Docked control must have a name
Error removing control from dock tree
- Dock zone not found
- Dock zone has no control
Unable to find a Table of Contents
Right
Shift+
Ctrl+
Clipboard does not support Icons
Cannot open clipboard
Menu '%s' is already being used by another form
Service failed on %s: %s
execute

GbPlugin

- Anyone know what this is???
- A little Google searching will tell you that this plugin is:
  - Used by Brazilian banks to protect customers during internet banking transactions.

Hex View: javafx.exe

javafx.exe

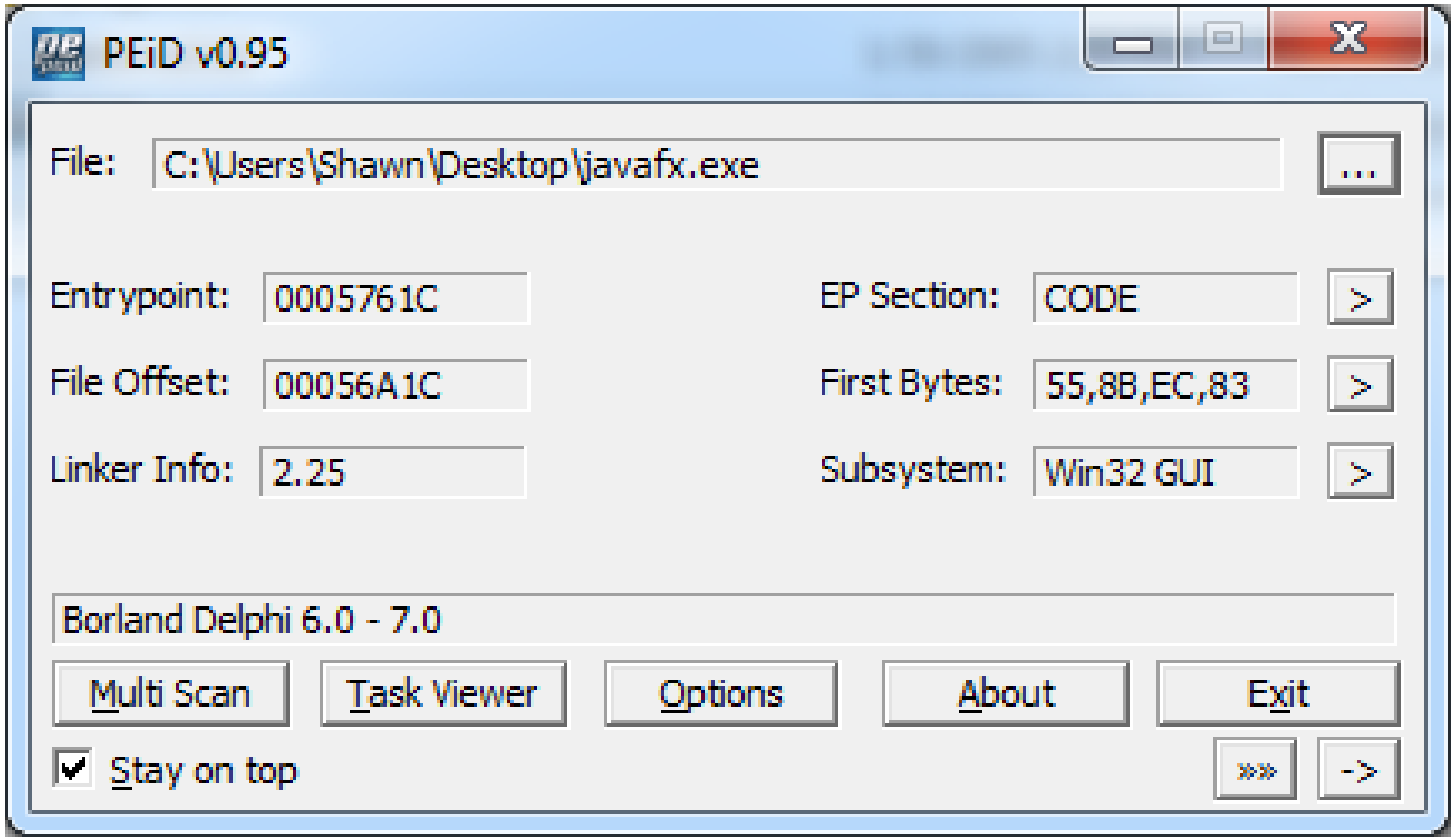
t	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00	4D	5A	50	00	02	00	00	00	04	00	0F	00	FF	FF	00	00	MZP ÿÿ
10	B8	00	00	00	00	00	00	00	40	00	1A	00	00	00	00	00	¸ @
20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
30	00	00	00	00	00	00	00	00	00	00	00	00	00	01	00	00	
40	BA	10	00	0E	1F	B4	09	CD	21	B8	01	4C	CD	21	90	90	º ´ Í!¸ LÍ!
50	54	68	69	73	20	70	72	6F	67	72	61	6D	20	6D	75	73	This program mus
60	74	20	62	65	20	72	75	6E	20	75	6E	64	65	72	20	57	t be run under W
70	69	6E	33	32	0D	0A	24	37	00	00	00	00	00	00	00	00	in32 $7
80	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
90	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00	50	45	00	00	4C	01	08	00	19	5E	42	2A	00	00	00	00	PE L ^B*
10	00	00	00	00	E0	00	8E	81	0B	01	02	19	00	68	05	00	à ▌ h
20	00	02	01	00	00	00	00	00	1C	76	05	00	00	10	00	00	v
30	00	80	05	00	00	00	22	01	00	10	00	00	00	02	00	00	▌ "

PEiD: javafx.exe

- Open PEiD tool
- Drag javafx.exe into it

PEiD: javafx.exe

- Valid PE (Portable Executable) file
- Borland Delphi and lots of strings = not packed



Packed Files

- Malicious executable is compressed with packer in order to obfuscate strings.
  - UPX, NSPack, Upack, FSG, etc.
- Executable must be unpacked to view strings.
- Generally, if you see many viewable strings, the executable is not packed.

PEview:

- Open Peview
- Close the Open dialog box
- Drag javafx into PEview GUI

- Used to view PE file format header information

Some Sections of a PE File

- DATA or .data
  - Contains the executable code
- BSS
  - Section for declaring variables
- .rdata
  - Read-only data accessed by program
- .rsrc
  - Resources needed by the executable

(Sikorski and Honig, 2012, p. 22)

PEview: javafx.exe – Compilation Date

PEview - C:\Users\Shawn\Desktop\javafx.exe

File View Go Help

- javafx.exe
  - IMAGE_DOS_HEADER
  - MS-DOS Stub Program
  - IMAGE_NT_HEADERS
    - Signature
    - IMAGE_FILE_HEADER
    - IMAGE_OPTIONAL_HEADER
  - IMAGE_SECTION_HEADER CODE
  - IMAGE_SECTION_HEADER DATA
  - IMAGE_SECTION_HEADER BSS
  - IMAGE_SECTION_HEADER .idata
  - IMAGE_SECTION_HEADER .tls
  - IMAGE_SECTION_HEADER .rdata
  - IMAGE_SECTION_HEADER .reloc
  - IMAGE_SECTION_HEADER .rsrc
  - SECTION CODE
  - SECTION DATA
  - SECTION BSS
  - SECTION .idata
  - SECTION .tls
  - SECTION .rdata
  - SECTION .reloc
  - SECTION .rsrc

pFile	Data	Description	Value
00000104	014C	Machine	IMAGE_FILE_MACHINE_I386
00000106	0008	Number of Sections	
00000108	2A425E19	Time Date Stamp	1992/06/19 Fri 22:22:17 UTC
0000010C	00000000	Pointer to Symbol Table	
00000110	00000000	Number of Symbols	
00000114	00E0	Size of Optional Header	
00000116	818E	Characteristics	
		0002	IMAGE_FILE_EXECUTABLE_IMAGE
		0004	IMAGE_FILE_LINE_NUMS_STRIPPED
		0008	IMAGE_FILE_LOCAL_SYMS_STRIPPED
		0080	IMAGE_FILE_BYTES_REVERSED_LO
		0100	IMAGE_FILE_32BIT_MACHINE
		8000	IMAGE_FILE_BYTES_REVERSED_HI

Viewing IMAGE_FILE_HEADER

PEview: javafx.exe – Console or GUI Program?

PEview - C:\Users\Shawn\Desktop\javafx.exe

File View Go Help

- javafx.exe
  - IMAGE_DOS_HEADER
  - MS-DOS Stub Program
  - IMAGE_NT_HEADERS
    - Signature
    - IMAGE_FILE_HEADER
    - IMAGE_OPTIONAL_HEADER
  - IMAGE_SECTION_HEADER CODE
  - IMAGE_SECTION_HEADER DATA
  - IMAGE_SECTION_HEADER BSS
  - IMAGE_SECTION_HEADER .idata
  - IMAGE_SECTION_HEADER .tls
  - IMAGE_SECTION_HEADER .rdata
  - IMAGE_SECTION_HEADER .reloc
  - IMAGE_SECTION_HEADER .rsrc
  - SECTION CODE
  - SECTION DATA
  - SECTION BSS
  - SECTION .idata
  - SECTION .tls
  - SECTION .rdata
  - SECTION .reloc
  - SECTION .rsrc

pFile	Data	Description	Value
00000158	00000000	Checksum	
0000015C	0002	Subsystem	IMAGE_SUBSYSTEM_WINDOWS_GUI
0000015E	0000	DLL Characteristics	
00000160	00100000	Size of Stack Reserve	
00000164	00004000	Size of Stack Commit	
00000168	00100000	Size of Heap Reserve	
0000016C	00001000	Size of Heap Commit	
00000170	00000000	Loader Flags	
00000174	00000010	Number of Data Directories	
00000178	00000000	RVA	EXPORT Table
0000017C	00000000	Size	
00000180	0005B000	RVA	IMPORT Table
00000184	0000237A	Size	
00000188	00067000	RVA	RESOURCE Table
0000018C	00006600	Size	
00000190	00000000	RVA	EXCEPTION Table
00000194	00000000	Size	
00000198	00000000	Offset	CERTIFICATE Table
0000019C	00000000	Size	
000001A0	00060000	RVA	BASE RELOCATION Table
000001A4	0000620C	Size	

Viewing IMAGE_OPTIONAL_HEADER

PEview: javafx.exe – Section Headers

PEview - C:\Users\Shawn\Desktop\javafx.exe

File View Go Help

- javafx.exe
  - IMAGE_DOS_HEADER
  - MS-DOS Stub Program
  - IMAGE_NT_HEADERS
    - Signature
    - IMAGE_FILE_HEADER
    - IMAGE_OPTIONAL_HEADER
  - IMAGE_SECTION_HEADER CODE
  - IMAGE_SECTION_HEADER DATA
  - IMAGE_SECTION_HEADER BSS
  - IMAGE_SECTION_HEADER .idata
  - IMAGE_SECTION_HEADER .tls
  - IMAGE_SECTION_HEADER .rdata
  - IMAGE_SECTION_HEADER .reloc
  - IMAGE_SECTION_HEADER .rsrc
  - SECTION CODE
  - SECTION DATA
  - SECTION BSS
  - SECTION .idata
  - SECTION .tls
  - SECTION .rdata
  - SECTION .reloc
  - SECTION .rsrc

pFile	Data	Description	Value
000001F8	43 4F 44 45	Name	CODE
000001FC	00 00 00 00		
00000200	00056688	Virtual Size	
00000204	00001000	RVA	
00000208	00056800	Size of Raw Data	
0000020C	00000400	Pointer to Raw Data	
00000210	00000000	Pointer to Relocations	
00000214	00000000	Pointer to Line Numbers	
00000218	0000	Number of Relocations	
0000021A	0000	Number of Line Numbers	
0000021C	60000020	Characteristics	
		00000020	IMAGE_SCN_CNT_CODE
		20000000	IMAGE_SCN_MEM_EXECUTE
		40000000	IMAGE_SCN_MEM_READ

Viewing IMAGE_SECTION_HEADER CODE

PEview: javafx.exe – Section Headers (Cont.)

- Describes each section of a PE file
- Virtual Size = How much space allocated during loading
- Size of Raw Data = How big the section is on disk

- If Virtual Size is much larger than Size of Raw Data:
  - Code could be packed

(Sikorski and Honig, 2012, p. 23)

PEview: javafx.exe – Section Headers (Cont.)

- Does this file seem to be packed based on Section Headers???

pFile	Data	Description	Value
000001F8	43 4F 44 45	Name	CODE
000001FC	00 00 00 00		
00000200	00056688	Virtual Size	
00000204	00001000	RVA	
00000208	00056800	Size of Raw Data	
0000020C	00000400	Pointer to Raw Data	
00000210	00000000	Pointer to Relocations	
00000214	00000000	Pointer to Line Numbers	
00000218	0000	Number of Relocations	
0000021A	0000	Number of Line Numbers	

.rsrc = Resource Section of PE file

- Icons used in executable
- GUI Menus
- Dialog Menus
- Strings
- Version/Company Name/Copyright

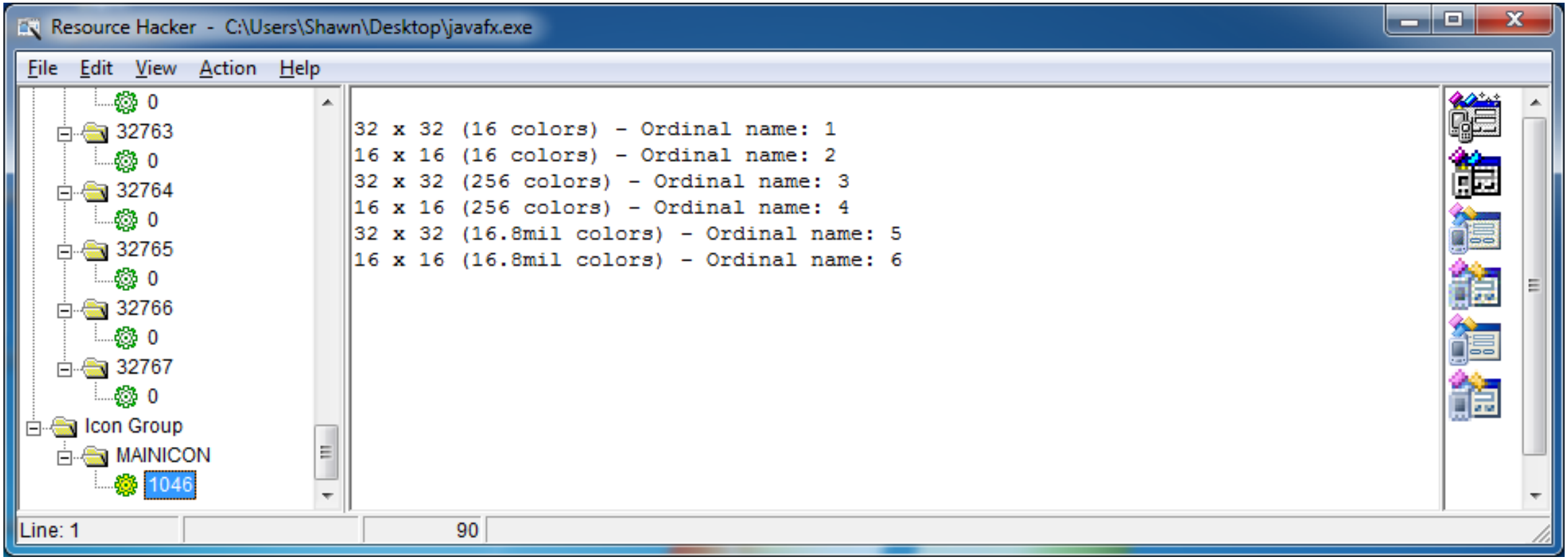
- Resource Hacker tool can view .rsrc section

(Sikorski and Honig, 2012, p. 25)

Resource Hacker: javafx.exe

- Open Resource Hacker tool
- Drag javafx.exe into tool
- Hit View \ Expand Tree

Resource Hacker: javafx.exe - Icons



Resource Hacker: javafx.exe - RCData

Resource Hacker - C:\Users\Shawn\Desktop\javafx.exe

File Edit View Action Help

- 4096
  - 0
- RCData
  - DVCLAL
    - 0
  - PACKAGEINFO
    - 0
  - TLISTTOPBW
    - 0
- Cursor Group
  - 32761
    - 0
  - 32762
    - 0
  - 32763
    - 0
  - 32764
    - 0
  - 32765
    - 0
  - 32766
    - 0
  - 32767
    - 0
- Icon Group

Compile Script

```
object ListTopBW: TListTopBW
  OldCreateOrder = False
  OnDestroy = ServiceDestroy
  DisplayName = 'Telos De Poira'
  LoadGroup = 'Boot File System'
  OnExecute = ServiceExecute
  OnShutdown = ServiceShutdown
  OnStart = ServiceStart
  Left = 494
  Top = 287
  Height = 170
  Width = 158
  object Timer1: TTimer
    Interval = 1
    OnTimer = Timer1Timer
    Left = 24
    Top = 24
  end
  object Timer2: TTimer
    Interval = 10000
    OnTimer = Timer2Timer
    Left = 80
    Top = 24
  end
end
```

Line: 1 358

Resource Hacker: javafx.exe – String Tables

Resource Hacker - C:\Users\Shawn\Desktop\javafx.exe

File Edit View Action Help

- 4093
  - 0
- 4094
  - 0
- 4095
  - 0
- 4096
  - 0
- RCData
  - DVCLAL
    - 0
  - PACKAGEINFO
    - 0
  - TLISTTOPBW
    - 0
- Cursor Group
  - 32761
    - 0
  - 32762
    - 0
  - 32763
    - 0
  - 32764
    - 0
  - 32765

Compile Script

```
STRINGTABLE
LANGUAGE LANG_NEUTRAL, SUBLANG_NEUTRAL
{
65520,      "'%s' is not a valid integer value"
65521,      "Out of memory"
65522,      "I/O error %d"
65523,      "File not found"
65524,      "Invalid filename"
65525,      "Too many open files"
65526,      "File access denied"
65527,      "Read beyond end of file"
65528,      "Disk full"
65529,      "Invalid numeric input"
65530,      "Division by zero"
65531,      "Range check error"
65532,      "Integer overflow"
65533,      "Invalid floating point operation"
65534,      "Floating point division by zero"
65535,      "Floating point overflow"
}
```

Line: 1 692

Imported Functions

- Malware often imports system DLLs which contain various functions that can be utilized.
- What is the difference between an EXE and a DLL???

EXE vs DLL

- EXE
  - Executable file that runs as its own process
  - Runs in its own address space
- DLL
  - Dynamic Link Library
  - Not directly executable, needs host EXE file to run it
  - Contains functions, classes, variables, resources, etc.
  - Does not run in its own address space

Why do Malware authors use DLLs?

- They can use existing Windows or 3rd party DLLs to keep their malware programs smaller
- To store malicious code

- You can analyze a DLL with Dependency Walker
  - In Dynamic analysis you could execute a DLL with the rundll32.exe command

Dependency Walker:

- Open the Dependency Walker tool
- Drag javafx.exe into it

Dependency Walker: javafx.exe

Dependency Walker - [javafx.exe]

File Edit View Options Profile Window Help

JAVAFX.EXE
- KERNEL32.DLL
- USER32.DLL
- ADVAPI32.DLL
- OLEAUT32.DLL
- KERNEL32.DLL
- ADVAPI32.DLL
- KERNEL32.DLL
- VERSION.DLL
- GDI32.DLL
- USER32.DLL
- KERNEL32.DLL
- OLEAUT32.DLL
- ADVAPI32.DLL
- COMCTL32.DLL

PI	Ordinal ^	Hint	Function
C	N/A	0 (0x0000)	RegQueryValueExA
C	N/A	0 (0x0000)	RegOpenKeyExA
C	N/A	0 (0x0000)	RegCloseKey

E	Ordinal ^	Hint	Function
O#	1000 (0x03E8)	N/A	N/A
C	1001 (0x03E9)	360 (0x0168)	I_ScGetCurrentGroupStateW
C	1002 (0x03EA)	0 (0x0000)	A_SHAFinal
C	1003 (0x03EB)	1 (0x0001)	A_SHAInit
C	1004 (0x03EC)	2 (0x0002)	A_SHAUpdate
C	1005 (0x03ED)	3 (0x0003)	AbortSystemShutdownA

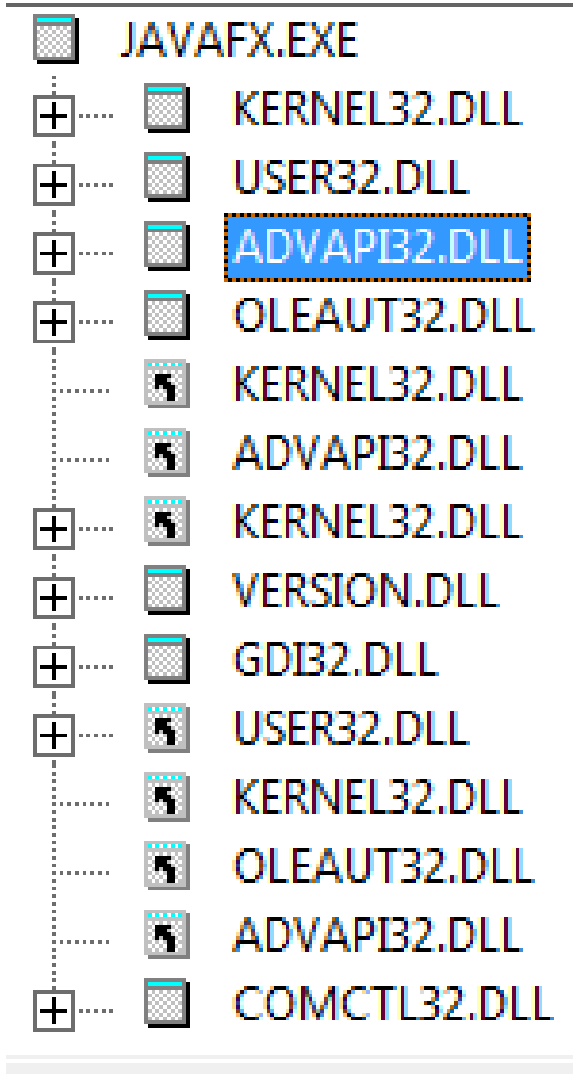
Module	File Time Stamp	Link Time Stamp	File Size	Attr.	Link C
GPSVC.DLL	Error opening file. The system cannot find the file specified (2).				
IESHIMS.DLL	Error opening file. The system cannot find the file specified (2).				
IEFRAME.DLL	11/20/2010 10:25p	11/20/2010 7:00a	10,990,080	A	0x00
SHLWAPI.DLL	11/20/2010 10:23p	11/20/2010 7:06a	350,208	A	0x00
ADVAPI32.DLL	11/20/2010 10:24p	11/20/2010 6:54a	640,512	A	0x00
API-MS-WIN-CORE-CONSOLE-L1-1-0.DLL	07/13/2009 8:03p	07/13/2009 8:04p	3,072	HA	0x00

Warning: At least one delay-load dependency module was not found.
Warning: At least one module has an unresolved import due to a missing export function in a delay-load dependent module.

DLLs being imported into malware exe

(Sikorski and Honig, 2012, p. 16)

Dependency Walker: javafx.exe



- KERNEL32.DLL - Common DLL, allows access of memory, files, and hardware
- USER32.DLL - Contains UI components (buttons, scroll bars, etc.)
- ADVAPI32.DLL - Allows access to Service Manager and Registry
- OLEAUT32.DLL - Contains functions to link to documents and other objects
- VERSION.DLL - Contains functions to retrieve file information from a file
- GDI32.DLL - Contains functions for displaying and manipulating graphics
- COMCTL32.DLL - Contains functions for common Windows controls

(Sikorski and Honig, 2012, p. 17)

Note about javafx.exe imported DLLs

- No networking DLLs were imported such as:
  - WSock32.dll
  - Ws2_32.dll
  - Wininet.dll

Dependency Walker: javafx.exe

Dependency Walker - [javafx.exe]

File Edit View Options Profile Window Help

JAVAFX.EXE
- KERNEL32.DLL
- USER32.DLL
- ADVAPI32.DLL
- OLEAUT32.DLL
- KERNEL32.DLL
- ADVAPI32.DLL
- KERNEL32.DLL
- VERSION.DLL
- GDI32.DLL
- USER32.DLL
- KERNEL32.DLL
- OLEAUT32.DLL
- ADVAPI32.DLL
- COMCTL32.DLL

PI	Ordinal ^	Hint	Function
C	N/A	0 (0x0000)	RegQueryValueExA
C	N/A	0 (0x0000)	RegOpenKeyExA
C	N/A	0 (0x0000)	RegCloseKey

E	Ordinal ^	Hint	Function
O#	1000 (0x03E8)	N/A	N/A
C	1001 (0x03E9)	360 (0x0168)	I_ScGetCurrentGroupStateW
C	1002 (0x03EA)	0 (0x0000)	A_SHAFinal
C	1003 (0x03EB)	1 (0x0001)	A_SHAInit
C	1004 (0x03EC)	2 (0x0002)	A_SHAUpdate
C	1005 (0x03ED)	3 (0x0003)	AbortSystemShutdownA

^	Module	File Time Stamp	Link Time Stamp	File Size	Attr.	Link
?	GPSVC.DLL	Error opening file. The system cannot find the file specified (2).				
?	IESHIMS.DLL	Error opening file. The system cannot find the file specified (2).				
	IEFRAME.DLL	11/20/2010 10:25p	11/20/2010 7:00a	10,990,080	A	0x00
	SHLWAPI.DLL	11/20/2010 10:23p	11/20/2010 7:06a	350,208	A	0x00
	ADVAPI32.DLL	11/20/2010 10:24p	11/20/2010 6:54a	640,512	A	0x00
	API-MS-WIN-CORE-CONSOLE-L1-1-0.DLL	07/13/2009 8:03p	07/13/2009 8:04p	3,072	HA	0x00

Warning: At least one delay-load dependency module was not found.
Warning: At least one module has an unresolved import due to a missing export function in a delay-load dependent module.

Functions being imported in malware EXE from selected DLL

(Can double click on function to query MSDN for more info)

(Sikorski and Honig, 2012, p. 16)

Interesting Imported Functions: javafx.exe

- KERNEL32.DLL
  - WriteFile
- USER32.DLL
  - GetKeyboardType, LoadSTringA, MessageBoxA, CharNextA
- ADVAPI32.DLL
  - ReqQueryValueExA, RegOpenKeyExA, RegCloseKey
- VERSION.DLL
  - VerQueryValueA, GetFileVersionInfoSizeA, Get FileVersionInfoA

Dependency Walker: javafx.exe

Dependency Walker - [javafx.exe]

File Edit View Options Profile Window Help

JAVAFX.EXE
- KERNEL32.DLL
- USER32.DLL
- ADVAPI32.DLL
- OLEAUT32.DLL
- KERNEL32.DLL
- ADVAPI32.DLL
- KERNEL32.DLL
- VERSION.DLL
- GDI32.DLL
- USER32.DLL
- KERNEL32.DLL
- OLEAUT32.DLL
- ADVAPI32.DLL
- COMCTL32.DLL

PI	Ordinal ^	Hint	Function
C	N/A	0 (0x0000)	RegQueryValueExA
C	N/A	0 (0x0000)	RegOpenKeyExA
C	N/A	0 (0x0000)	RegCloseKey

E	Ordinal ^	Hint	Function
O#	1000 (0x03E8)	N/A	N/A
C	1001 (0x03E9)	360 (0x0168)	I_ScGetCurrentGroupStateW
C	1002 (0x03EA)	0 (0x0000)	A_SHAFinal
C	1003 (0x03EB)	1 (0x0001)	A_SHAInit
C	1004 (0x03EC)	2 (0x0002)	A_SHAUpdate
C	1005 (0x03ED)	3 (0x0003)	AbortSystemShutdownA

^	Module	File Time Stamp	Link Time Stamp	File Size	Attr.	Link
?	GPSVC.DLL	Error opening file. The system cannot find the file specified (2).				
?	IESHIMS.DLL	Error opening file. The system cannot find the file specified (2).				
	IEFRAME.DLL	11/20/2010 10:25p	11/20/2010 7:00a	10,990,080	A	0x00
	SHLWAPI.DLL	11/20/2010 10:23p	11/20/2010 7:06a	350,208	A	0x00
	ADVAPI32.DLL	11/20/2010 10:24p	11/20/2010 6:54a	640,512	A	0x00
	API-MS-WIN-CORE-CONSOLE-L1-1-0.DLL	07/13/2009 8:03p	07/13/2009 8:04p	3,072	HA	0x00

Warning: At least one delay-load dependency module was not found.
Warning: At least one module has an unresolved import due to a missing export function in a delay-load dependent module.

All functions that could be exported from selected DLL

(Sikorski and Honig, 2012, p. 16)

Part III

Advanced Static Analysis

Advanced Static Analysis

- Using a disassembler like IDA Pro
- X86 disassembly is a specialized skill

Levels of Abstraction

- Malware authors write code in a high level language such as C, C++, Java, etc.
- Compiled into binary code which is at machine code level for the computer to run.

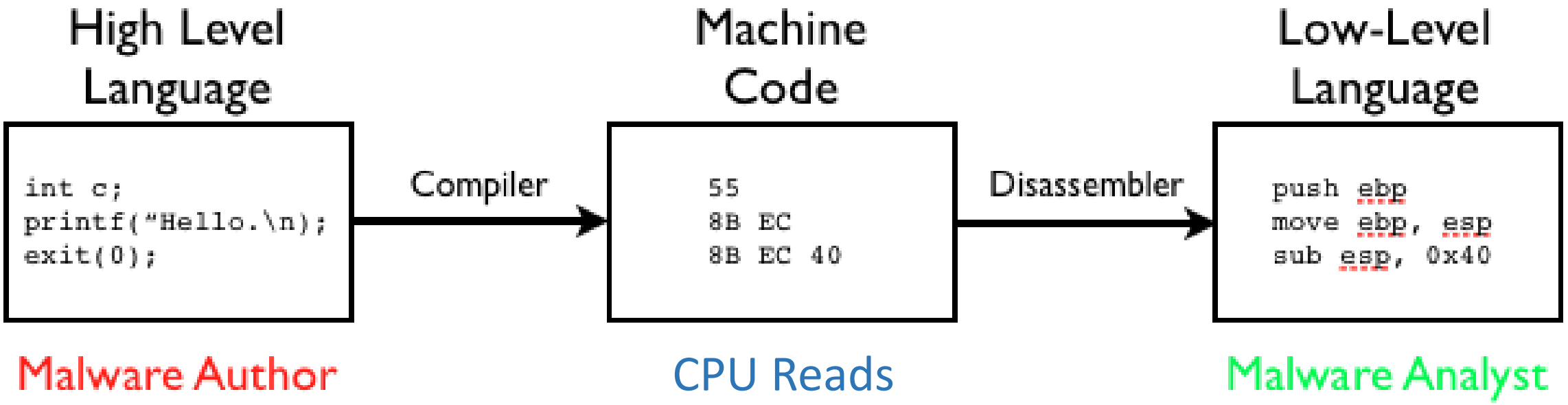
(Sikorski and Honig, 2012, p. 67)

Levels of Abstraction (Cont.)

- Machine code consists of opcodes:
  - Hexadecimal digits too difficult for human to comprehend
- Disassembler takes malware binary as input and generates low-level assembly code
  - Easier to read for analysts

(Sikorski and Honig, 2012, p. 67)

Levels of Abstraction (Cont.)



(Sikorski and Honig, 2012, p. 66)

Dialects of Assembly Language

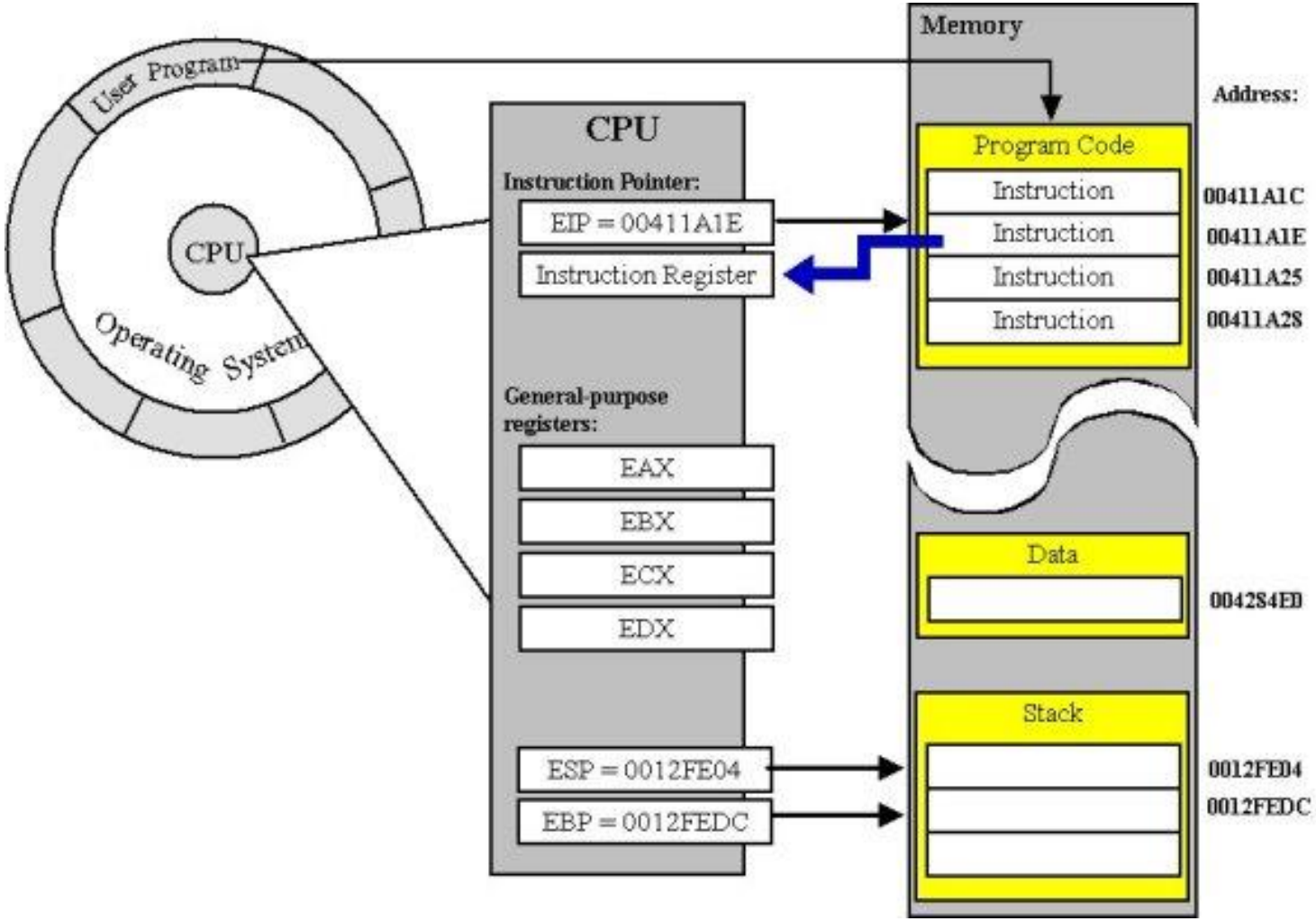
- Each family of microprocessors has a different assembly dialect
- x86, x64, SPARC, PowerPC, MIPS, ARM, etc.
- X86 most popular architecture for PCs

(Sikorski and Honig, 2012, p. 67)

Common Assembly Conditions

- JMP
  - Transfers control to new address
- CMP
  - Compares two operands
- MOV
  - Copies source operand to destination operand without changing the source
- PUSH
  - Write value to stack
- POP
  - Take whatever is on top of stack and put it into a register

CPU Registers & RAM Memory Stack



http://www.c-jump.com/CIS60/lecture01_2.htm

Uses of General-Purpose CPU Registers

Register	Size	Typical Uses
EAX	32-bit	Accumulator for operands and results
EBX	32-bit	Base pointer to data in the data segment
ECX	32-bit	Counter for loop operations
EDX	32-bit	Data pointer and I/O pointer
EBP	32-bit	Frame Pointer - useful for stack frames
ESP	32-bit	Stack Pointer - hardcoded into PUSH and POP operations
ESI	32-bit	Source Index - required for some array operations
EDI	32-bit	Destination Index - required for some array operations
EIP	32-bit	Instruction Pointer
EFLAGS	32-bit	Result Flags - hardcoded into conditional operations

http://www.c-jump.com/CIS77/ASM/Assembly/lecture.html

Assembly Code Example – Hello World

```
section .text
   global_start      ;must be declared for linker (ld)

_start:              ;tells linker entry point
   mov  edx,len      ;message length
   mov  ecx,msg      ;message to write
   mov  ebx,1        ;file descriptor (stdout)
   mov  eax,4        ;system call number (sys_write)
   int  0x80         ;call kernel

   mov  eax,1        ;system call number (sys_exit)
   int  0x80         ;call kernel

section .data
msg db 'Hello, world!', 0xa  ;our dear string
len equ $ - msg              ;length of our dear string
```

http://www.tutorialspoint.com/assembly_programming/assembly_basic_syntax.htm

IDA: javafx.exe

- We don’t have time to cover learning assembly language in this course but we will look at some IDA basics.

IDA: javafx.exe

IDA - C:\Documents and Settings\Forsec1\Desktop\Pcaps\Objects\javafx.exe

File Edit Jump Search View Debugger Options Windows Help

IDA View-A | Hex View-A | Exports | Imports | Names | Functions | Strings | Structures | Enums

```
CODE:0127761C ; ---------------------------------------------------------------
CODE:0127761C
CODE:0127761C                 public start
CODE:0127761C start:
CODE:0127761C                 push    ebp
CODE:0127761D                 mov     ebp, esp
CODE:0127761F                 add     esp, 0FFFFFFF0h
CODE:01277622                 push    ebx
CODE:01277623                 mov     eax, offset dword_127748
CODE:01277628                 call    sub_12263E4
CODE:0127762D                 jmp     short loc_127763F
CODE:0127762D ; ---------------------------------------------------------------
CODE:0127762F                 db 57h
```

Program control flow

00056A1C | 0127761C: CODE:start

Names win...

Name
GetStdHandle
RaiseException

Line 1 of 1174

Strings win...

Address	Length
CODE:0...	0000001C
CODE:0...	0000000D

Imports | Exports

```
File 'C:\Documents and Settings\Forsec1\Desktop\Pcaps\Objects\javafx.exe' is successfully loaded into the database.
Compiling file 'C:\Program Files\IDA Free\idc\ida.idc'...
Executing function 'main'...
Compiling file 'C:\Program Files\IDA Free\idc\onload.idc'...
Executing function 'OnLoad'...
IDA is analysing the input file...
You may start to explore the input file right now.
Propagating type information...
Function argument information is propagated
The initial autoanalysis has been finished.
```

AU: idle | Down | Disk: 6GB | Program control flow

IDA: javafx.exe

1. Open IDA Pro Free, Select “Go - Work on your own”
2. Drag javafx.exe into the IDA GUI
3. Leave the selection on PE and select “OK”
4. Wait until the file finishes loading (Numbers will stop in bottom left corner)
5. Expand the “Strings window”
6. Find the GbPlugin strings near the bottom.

IDA: javafx.exe (Cont.)

6. Double click on one of the GbPlugin lines

"..." Strings window

Address	Length	Type	String
"..." CODE:0...	00000020	C	C:\\Program Files (x86)\\GbPlugin
"..." CODE:0...	0000001D	C	C:\\Windows\\SysWOW64\\drivers\\
"..." CODE:0...	00000018	C	C:\\Program Files (x86)\\
"..." CODE:0...	0000001A	C	C:\\Program Files\\GbPlugin
"..." CODE:0...	0000001D	C	C:\\Windows\\System32\\drivers\\
"..." CODE:0...	00000022	C	C:\\Arquivos de Programas\\GbPlugin
"..." CODE:0...	0000001A	C	C:\\Arquivos de Programas\\
"..." CODE:0...	00000012	C	s\" /T /E /C /P SY
"..." CODE:0...	00000015	C	I\" /T /E /C /P SYSTE
"..." CODE:0...	00000011	C	*.*\" /T /E /C /
"..." CODE:0...	00000011	C	*\" /T /E /C /P
"..." CODE:0...	00000011	C	\" /T /E /C /P To
"..." CODE:0...	00000013	C	c\" /T /E /C /P Tod

IDA: javafx.exe (Cont.)

7. Expand the main IDA-View-A window
8. Double-click on one of the XREFs (cross-references under “GBPlugin”

IDA View-A

```
CODE:01276E90                 dd 0FFFFFFFFh, 19h
CODE:01276E98 aCProgramFilesG db 'C:\Program Files\GbPlugin',0 ; DATA XREF: sub_12767EC+7F↑o
CODE:01276EB2                 align 4
CODE:01276EB4                 dd 0FFFFFFFFh, 1Ch
CODE:01276EBC aCWindowsSystem db 'C:\Windows\System32\drivers\',0
CODE:01276EBC                                         ; DATA XREF: sub_12767EC+8C↑o
CODE:01276ED9                 align 4
CODE:01276EDC                 dd 0FFFFFFFFh, 21h
CODE:01276EE4 aCArquivosDePro db 'C:\Arquivos de Programas\GbPlugin',0
CODE:01276EE4                                         ; DATA XREF: sub_12767EC+A8↑o
CODE:01276F06                 align 4
CODE:01276F08                 dd 0FFFFFFFFh, 19h
CODE:01276F10 aCArquivosDeP_0 db 'C:\Arquivos de Programas\',0 ; DATA XREF: sub_12767EC+B5↑o
CODE:01276F2A                 align 4
CODE:01276F2C                 dd 0FFFFFFFFh, 7
CODE:01276F34 dword_1276F34   dd 6C636163h, 222073h, 0FFFFFFFFh, 1
CODE:01276F34                                         ; DATA XREF: sub_12767EC+D1↑o
CODE:01276F34                                         ; sub_12767EC+10F↑o ...
CODE:01276F44 dword_1276F44   dd 67h, 0FFFFFFFFh, 2   ; DATA XREF: sub_12767EC+D9↑o
CODE:01276F50 dword_1276F50   dd 7062h, 0FFFFFFFFh, 5 ; DATA XREF: sub_12767EC+DE↑o
CODE:01276F5C dword_1276F5C   dd 732E6D6Bh, 79h, 0FFFFFFFFh, 11h
CODE:01276F5C                                         ; DATA XREF: sub_12767EC+E3↑o
CODE:01276F6C aSTECPSy        db 's" /T /E /C /P SY',0 ; DATA XREF: sub_12767EC+E8↑o
CODE:01276F7E                 align 10h
CODE:01276F80                 dd 0FFFFFFFFh, 6
```

IDA: javafx.exe (Cont.)

```
mov     edx, offset aCProgramFilesG ; "C:\\Program Files\\GbPlugin"
call    sub_1224578
lea     eax, [ebp+var_8]
mov     edx, offset aCWindowsSystem ; "C:\\Windows\\System32\\drivers\\"
call    sub_1224578
lea     eax, [ebp+var_C]
mov     edx, offset aCProgramFiles ; "C:\\Program Files\\"
call    sub_1224578
jmp     short loc_12768AB

loc_12768AB:
mov     eax, [ebp+var_4]
call    sub_1228984
test    al, al
jz      loc_1276DAB

push    0
push    offset dword_1276F34
push    [ebp+var_8]
push    offset dword_1276F44
push    offset dword_1276F50
push    offset dword_1276F5C
push    offset aSTECPSy ; "s\" /T /E /C /P SY"
push    offset dword_1276F88 ; uCmdShow
lea     eax, [ebp+var_14]
mov     edx, 7
call    sub_1224860
```

9. This takes you to graph view of that XREF.
10. You can drag the graph around to see the conditional jumps.

Part IV

Basic Dynamic Analysis

Dynamic Analysis

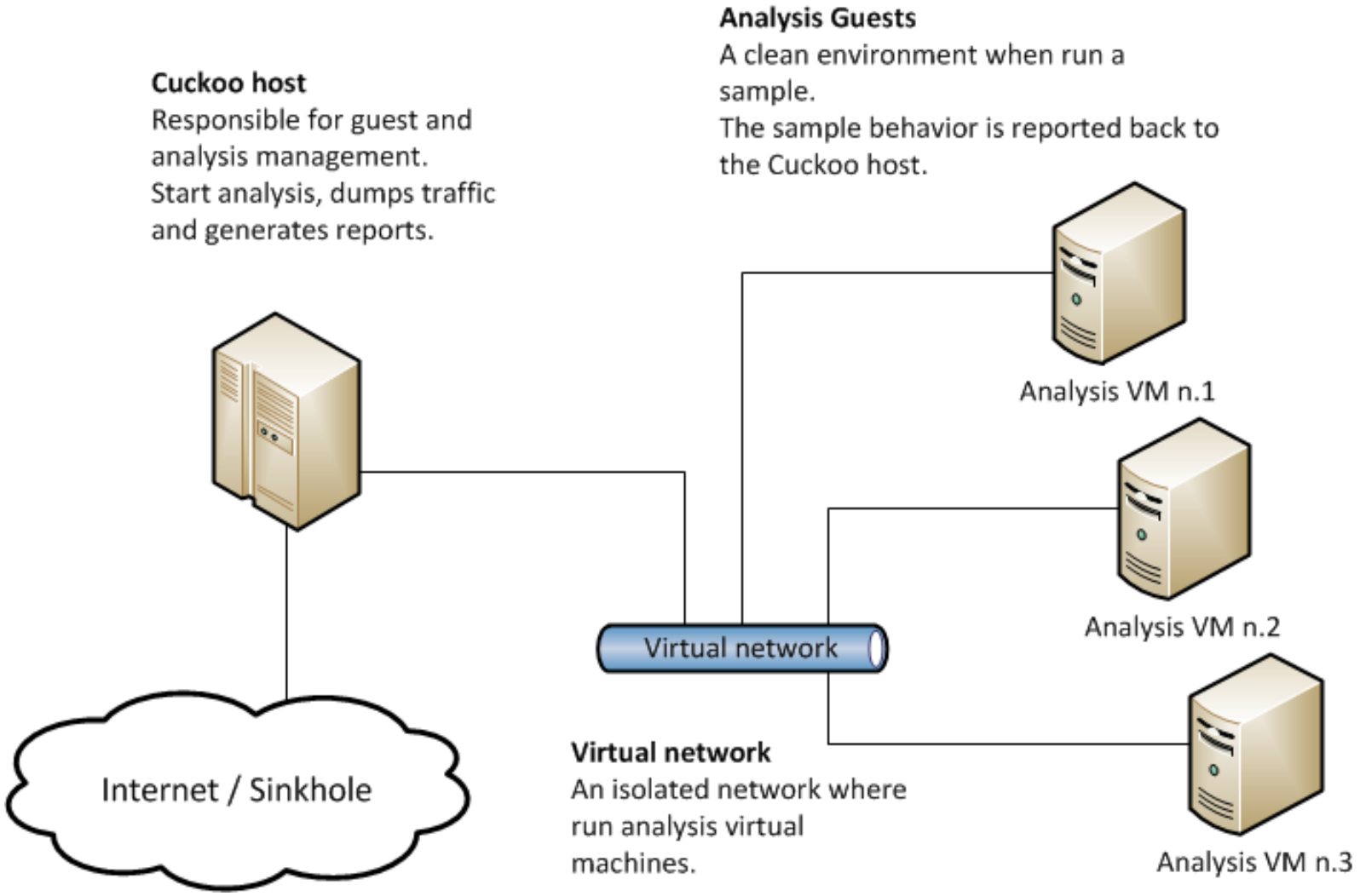
- Basic Dynamic Analysis
  - Executing the malware inside an isolated environment
  - Observing its behavior to determine behavior and produce effective signatures.
- Advanced Dynamic Analysis
  - Examine the internal state of running malware inside an isolated environment using a debugger (such as OllyDbg).

(Sikorski and Honig, 2012, pp. 2-3)

Malware Sandboxes

- Provides automated dynamic analysis
- Open source solution to run locally:
  - http://www.cuckoosandbox.org
- Online Sandboxes that accept sample uploads:
  - http://www.virustotal.com
  - http://www.malwr.com
    - Runs Cuckoobox
  - http://anubis.iseclab.org

Cuckoo Sandbox



http://cuckoo.readthedocs.org/en/latest/_images/architecture-main.png

Malware Sandboxes Pros vs. Cons

- Pros:
  - Great for large organizations/many daily samples
  - Helps analysts narrow down which samples need manual analysis
- Cons:
  - Runs executable without command-line options
  - Malware sleep function might evade quick running Sandbox
  - Doesn’t tell you what malware does.

Malware Sandboxes (Cont.)

- Sandboxes should not replace the need for manual analysis skills

Basic Dynamic Analysis Tasks

1. Monitor registry changes
2. Monitor processes
3. Monitor network activity

Setting up the Test – Follow along

1. Turn off networking (unless you are using a dirty pipe which we are not)
2. Ensure you have a clean snapshot of your VM
3. Run Fakenet
4. Run Process Monitor (procmon)
5. Run Process Explorer (procexp)
6. Run Regshot-x86-ANSI.exe

Setting up the Test (Cont.)

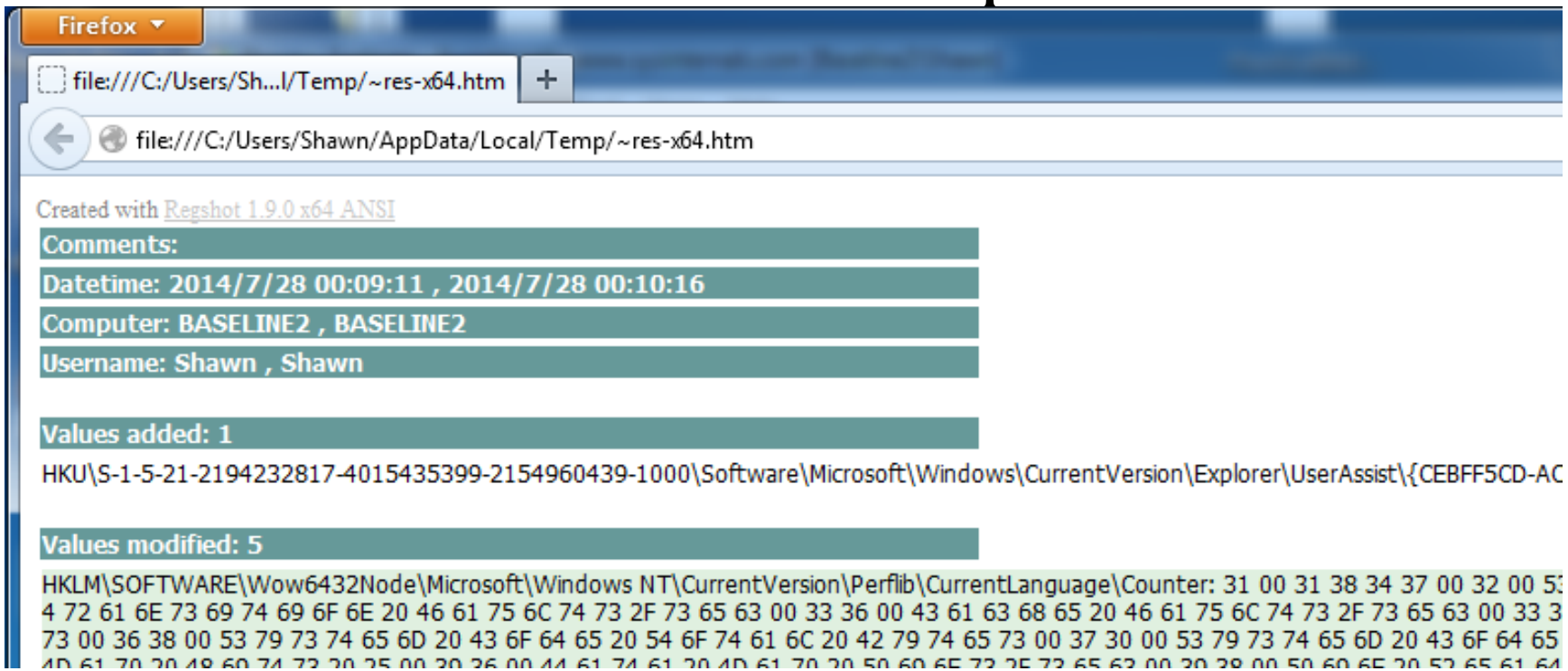
7. In Regshot, select "1st shot" and then "shot"
8. Wait until Regshot is done taking the first registry snapshot (2nd shot will light up)
9. Put Process Explorer where you can see it

Setting up the Test (Cont.)

10. Execute the malware (javafx.exe)
11. In Procmon, click File, and unselect "Capture Events"
12. Wait about a few seconds and click "2nd shot" and then "shot" in Regshot

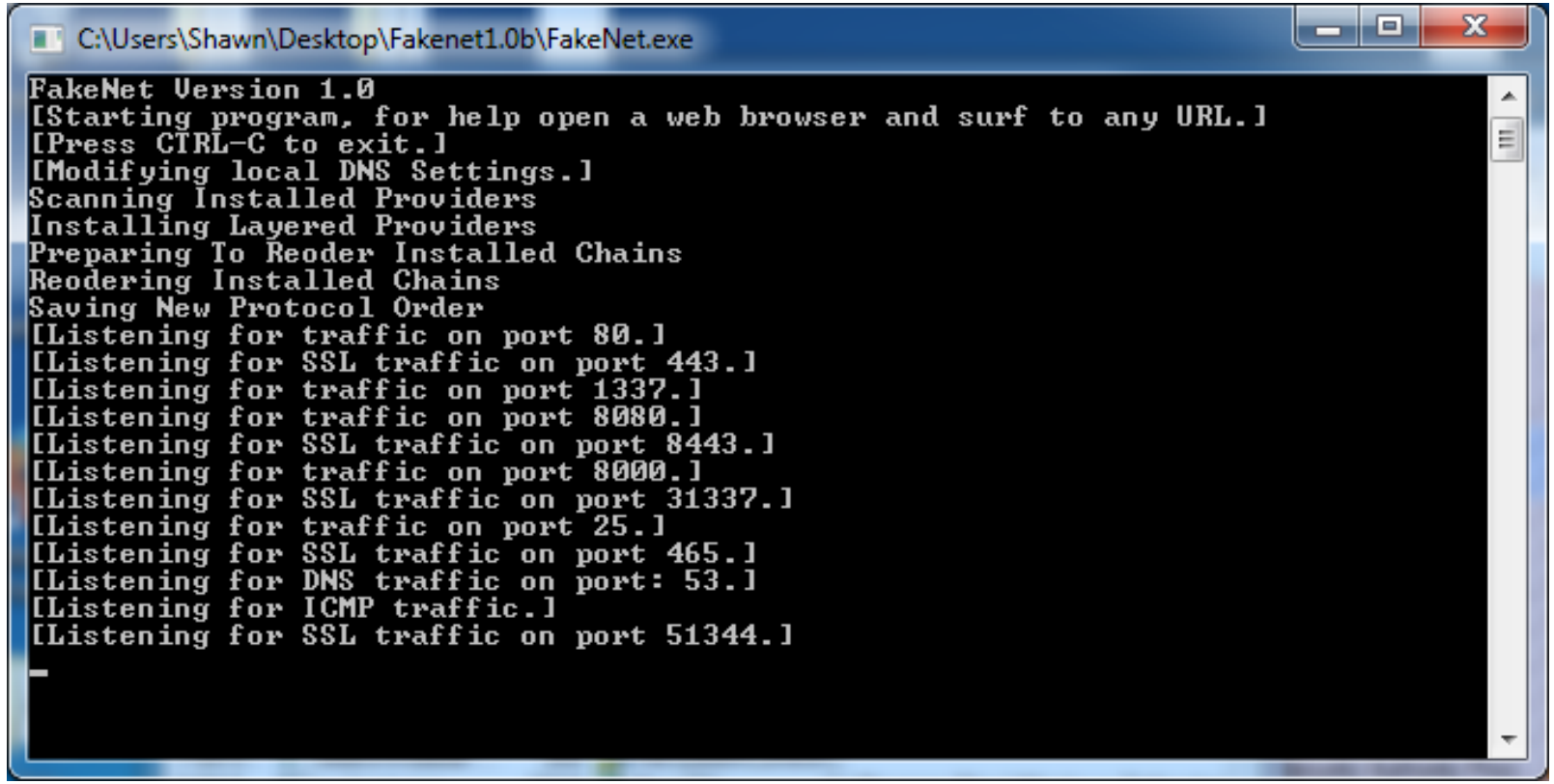
Analyzing the Test - Regshot

- Click “Compare” in Regshot
  - Shows Values added, modified, and deleted
  - Information doesn’t help us for this malware



Analyzing the Test - FakeNet

- Pull up the FakeNet Window
- No outbound traffic requests were made from the malware:



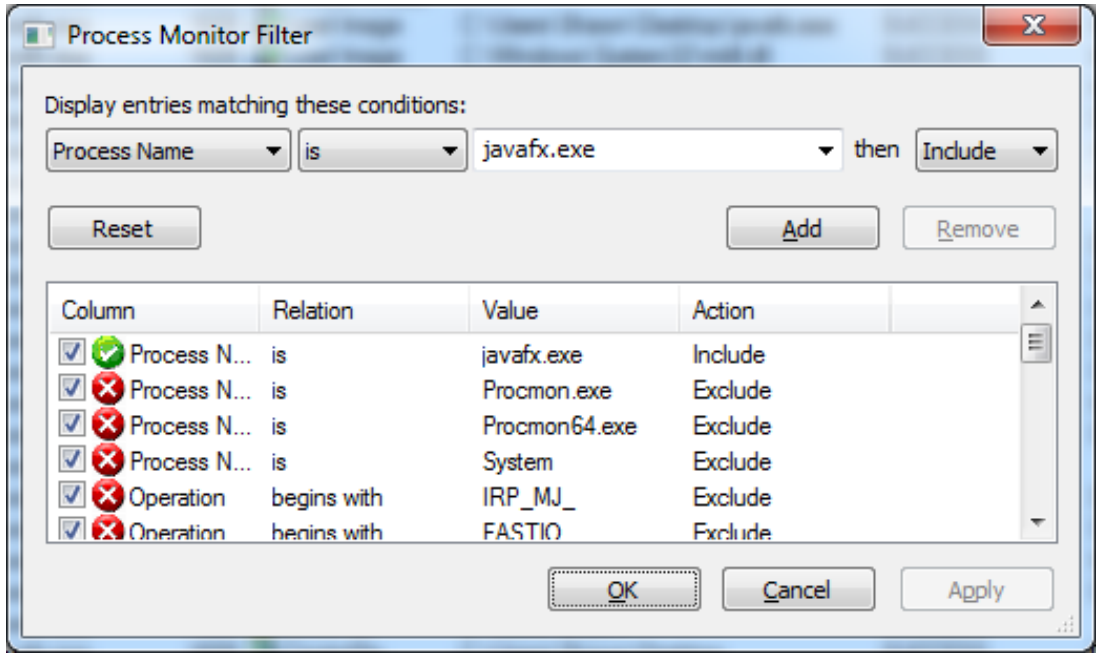
Analyzing the Test – Process Explorer

- javafx.exe process appeared and then disappeared
- Malware appeared to cleanly exit without starting another hidden process

**Process Explorer is more useful when malware continues to run or starts a new process.

Analyzing the Test - ProcMon

- Select “Filter” and “Filter” in ProcMon
- Change Filter so that Process Name is javafx.exe
- Click Add, Apply, and OK



Analyzing the Test - ProcMon

- Now all of the processes shown will be for javafx.exe only
- Expand “Detail” column to the right

Process Monitor - Sysinternals: www.sysinternals.com

File Edit Event Filter Tools Options Help

Time ...	Process Name	PID	Operation	Path	Result	Detail
7:09:5...	javafx.exe	1608	Process Start		SUCCESS	Parent PID: 1964, Command line: "C:\Users\Shawn\Desktop\javafx.exe" , Current directory: C:\Users\Shawn\Desktop\, Envi.
7:09:5...	javafx.exe	1608	Thread Create		SUCCESS	Thread ID: 1892
7:09:5...	javafx.exe	1608	Load Image	C:\Users\Shawn\Desktop\javafx.exe	SUCCESS	Image Base: 0x1220000, Image Size: 0x6e000
7:09:5...	javafx.exe	1608	Load Image	C:\Windows\System32\ntdll.dll	SUCCESS	Image Base: 0x76cd0000, Image Size: 0x1a9000
7:09:5...	javafx.exe	1608	Load Image	C:\Windows\SysWOW64\ntdll.dll	SUCCESS	Image Base: 0x76eb0000, Image Size: 0x180000
7:09:5...	javafx.exe	1608	CreateFile	C:\Windows\Prefetch\JAVAFX.EXE-2C743E64.pf	SUCCESS	Desired Access: Generic Read, Disposition: Open, Options: Synchronous IO Non-Alert, Attributes: n/a, ShareMode: None, Allo..
7:09:5...	javafx.exe	1608	QueryStandardI...	C:\Windows\Prefetch\JAVAFX.EXE-2C743E64.pf	SUCCESS	AllocationSize: 16,384, EndOfFile: 16,120, NumberOfLinks: 1, DeletePending: False, Directory: False
7:09:5...	javafx.exe	1608	ReadFile	C:\Windows\Prefetch\JAVAFX.EXE-2C743E64.pf	SUCCESS	Offset: 0, Length: 16,120, Priority: Normal
7:09:5...	javafx.exe	1608	CloseFile	C:\Windows\Prefetch\JAVAFX.EXE-2C743E64.pf	SUCCESS	
7:09:5...	javafx.exe	1608	CreateFile	C:	SUCCESS	Desired Access: Read Attributes, Write Attributes, Synchronize, Disposition: Open, Options: Synchronous IO Non-Alert, Comple.
7:09:5...	javafx.exe	1608	QueryInformatio...	C:	SUCCESS	VolumeCreationTime: 9/10/2013 2:19:00 PM, VolumeSerialNumber: E0AF-2 ... ects: True, VolumeLabel:
7:09:5...	javafx.exe	1608	FileSystemControl	C:	SUCCESS	Control: FSCTL_FILE_PREFETCH
7:09:5...	javafx.exe	1608	CreateFile	C:\Users	SUCCESS	Desired Access: Read Data/List Directory, Synchronize, Disposition: Open, ... Synchronous IO Non-Alert, Com.
7:09:5...	javafx.exe	1608	SetBasicInform...	C:\Users	SUCCESS	CreationTime: -1, LastAccessTime: -1, LastWriteTime: -1, ChangeTime: -1, FileAttributes: n/a
7:09:5...	javafx.exe	1608	QueryFileIntern...	C:\Users	SUCCESS	IndexNumber: 0x10000000001cc

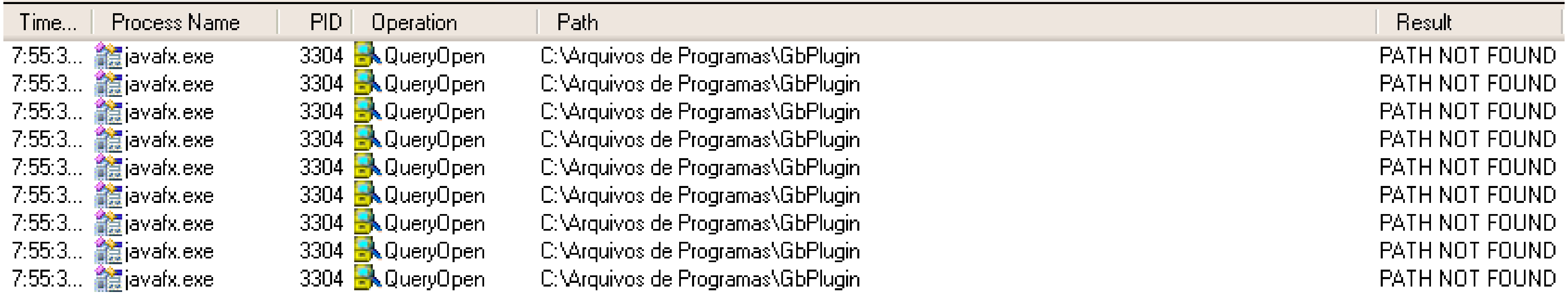
Offset: 0
Length: 16,120
Priority: Normal

Analyzing the Test - ProcMon

- Look through the Malware's process events.
- What pops out to you???

Analyzing the Test - ProcMon

- A lot of queries for C:\Arquivos de Programas\GbPlugin but “PATH NOT FOUND”



Time...	Process Name	PID	Operation	Path	Result
7:55:3...	javafx.exe	3304	QueryOpen	C:\Arquivos de Programas\GbPlugin	PATH NOT FOUND
7:55:3...	javafx.exe	3304	QueryOpen	C:\Arquivos de Programas\GbPlugin	PATH NOT FOUND
7:55:3...	javafx.exe	3304	QueryOpen	C:\Arquivos de Programas\GbPlugin	PATH NOT FOUND
7:55:3...	javafx.exe	3304	QueryOpen	C:\Arquivos de Programas\GbPlugin	PATH NOT FOUND
7:55:3...	javafx.exe	3304	QueryOpen	C:\Arquivos de Programas\GbPlugin	PATH NOT FOUND
7:55:3...	javafx.exe	3304	QueryOpen	C:\Arquivos de Programas\GbPlugin	PATH NOT FOUND
7:55:3...	javafx.exe	3304	QueryOpen	C:\Arquivos de Programas\GbPlugin	PATH NOT FOUND
7:55:3...	javafx.exe	3304	QueryOpen	C:\Arquivos de Programas\GbPlugin	PATH NOT FOUND
7:55:3...	javafx.exe	3304	QueryOpen	C:\Arquivos de Programas\GbPlugin	PATH NOT FOUND

- Showed the malware process exited on its own.
- No other processes spawned.

What now???

- Our analysis appears that the malware is looking for the GbPlugin
- We would like to know what the malware will do if it finds the GbPlugin on a system.
- Any ideas???

Next steps

- Let's create a fake GbPlugin directory at the path the malware was searching for:

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Forsec1>mkdir c:\"Arquivos de Programas"\GbPlugin

C:\Documents and Settings\Forsec1>cd "c:\Arquivos de Programas\GbPlugin"

C:\Arquivos de Programas\GbPlugin>start notepad test.dll

C:\Arquivos de Programas\GbPlugin>
```

Create new file in Notepad.

Type a few characters and save the file.

Next steps (cont.)

- Re-run the similar ProcMon test as before
- Select “Edit” and “Clear Display”
- Select “File” and check “Capture Events”
  - Your javafx.exe filter is still active
- Execute javafx.exe again
- Wait 10 seconds
- Select “File” and uncheck “Capture Events”

2nd Test Result

- What do you notice a lot of now that you didn't before???

2nd Test Result

- Add a filter of Detail contains GbPlugin to narrow down the cacls commands.

PID	Operation	Path	Result	Detail
1568	Process Create	C:\WINDOWS\system32\cacls.exe	SUCCESS	PID: 1676, Command line: cacls "C:\Arquivos de Programas\GbPlugin*.dll" /T /E /C /P SYSTEM:N
1568	Process Create	C:\WINDOWS\system32\cacls.exe	SUCCESS	PID: 404, Command line: cacls "C:\Arquivos de Programas\GbPlugin*.*" /T /E /C /P SYSTEM:N
1568	Process Create	C:\WINDOWS\system32\cacls.exe	SUCCESS	PID: 784, Command line: cacls "C:\Arquivos de Programas\GbPlugin*" /T /E /C /P SYSTEM:N
1568	Process Create	C:\WINDOWS\system32\cacls.exe	SUCCESS	PID: 1108, Command line: cacls "C:\Arquivos de Programas\GbPlugin*.gpc" /T /E /C /P SYSTEM:N
1568	Process Create	C:\WINDOWS\system32\cacls.exe	SUCCESS	PID: 1212, Command line: cacls "C:\Arquivos de Programas\GbPlugin*.dll" /T /E /C /P Todos:N
1568	Process Create	C:\WINDOWS\system32\cacls.exe	SUCCESS	PID: 1100, Command line: cacls "C:\Arquivos de Programas\GbPlugin*.exe" /T /E /C /P Todos:N
1568	Process Create	C:\WINDOWS\system32\cacls.exe	SUCCESS	PID: 1148, Command line: cacls "C:\Arquivos de Programas\GbPlugin*" /T /E /C /P Todos:N
1568	Process Create	C:\WINDOWS\system32\cacls.exe	SUCCESS	PID: 1068, Command line: cacls "C:\Arquivos de Programas\GbPlugin*.gpc" /T /E /C /P Todos:N

- What does the cacls command do???

calcs command – javafx.exe

- cacls "C:\Arquivos de Programas/GbPlugin*" /T /E /C /P SYSTEM:N
- cacls command modifies ACLs
- /T – Change ACL
- /E – Edits ACL
- /C – Continues
- /P – Replaces existing permissions
- SYSTEM:N – Specifies no access to SYSTEM user

What is the cacls command being used to do overall?

- cacls command and options would effectively disable the GbPlugin banking security driver if it were present on the system

Part V

Advanced Dynamic Analysis

Advanced Dynamic Analysis Tasks

1. Running the Malware in a Debugger
2. Watching values of memory addresses as they change during execution
3. Could alter execution by changing variables

(Sikorski and Honig, 2012, p. 167)

OllyDbg

- 32-bit assembly level debugger for Windows Windows binaries
- We won't demo this tool in class but you will use it to set a breakpoint in the homework lab

OllyDbg Windows

Memory Address

Opcodes

Assembly Language

Olly Comments

Dump

CPU Registers

Flags

Floating Point Registers

Stack

OllyDbg - javafx.exe

File View Debug Plugins Options Window Help

CPU - main thread, module javafx

```
0127761C  $ 55            PUSH EBP
0127761D  . 8BEC          MOV EBP,ESP
0127761F  . 83C4 F0       ADD ESP,-10
01277622  . 53            PUSH EBX
01277623  . B8 84742701   MOV EAX,javafx.01277484
01277628  . E8 B7EDFAFF   CALL javafx.012263E4
0127762D  .v EB 10        JMP SHORT javafx.0127763F
0127762F    57            DB 57                    CHAR 'W'
01277630    4C            DB 4C                    CHAR 'L'
01277631    20            DB 20                    CHAR ' '
01277632    20            DB 20                    CHAR ' '
01277633    05            DB 05
01277634    00            DB 00
01277635    00            DB 00
01277636    00            DB 00
01277637    00            DB 00
01277638    00            DB 00
01277639    00            DB 00
0127763A    00            DB 00
0127763B    57            DB 57                    CHAR 'W'
0127763C    4C            DB 4C                    CHAR 'L'
0127763D    20            DB 20                    CHAR ' '
0127763E    20            DB 20                    CHAR ' '
0127763F  > A1 30902701   MOV EAX,DWORD PTR DS:[1279030]
01277644  . 8B00          MOV EAX,DWORD PTR DS:[EAX]
01277646  . 8B10          MOV EDX,DWORD PTR DS:[EAX]
01277648  . FF52 34       CALL DWORD PTR DS:[EDX+34]
0127764B  . 8B0D C8902701 MOV ECX,DWORD PTR DS:[12790C8]   javafx.0127AB9C
01277651  . A1 30902701   MOV EAX,DWORD PTR DS:[1279030]
EBP=0012FFF0
```

```
Registers (FPU)
EAX 00000000
ECX 0012FFB0
EDX 7C90E4F4 ntdll.KiFastSystemCallRet
EBX 7FFDF000
ESP 0012FFC4
EBP 0012FFF0
ESI FFFFFFFF
EDI 7C910208 ntdll.7C910208
EIP 0127761C javafx.<ModuleEntryPoint>
C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 1  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDE000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_NO_IMPERSONATION_TOKEN (0000051D)
EFL 00000246 (NO,NB,E,BE,NS,PE,GE,LE)
ST0 empty -UNORM BDEC 01050104 005C0067
ST1 empty +UNORM 0069 006E0069 002E0067
ST2 empty 0.0
ST3 empty 0.0
ST4 empty 0.0
ST5 empty 0.0
ST6 empty 0.0
ST7 empty 0.0
               3 2 1 0      E S P U O Z D I
FST 0000  Cond 0 0 0 0  Err 0 0 0 0 0 0 0 0  (GT)
FCW 027F  Prec NEAR,53  Mask    1 1 1 1 1 1
```

```
Address   Hex dump                 ASCII
01278000  00 00 00 00 00 00 00 00  ........
01278008  02 8D 40 00 00 00 00 00  ☺ì@.....
01278010  00 00 00 00 00 00 00 00  ........
01278018  00 00 00 00 00 00 00 00  ........
01278020  32 13 8B C0 02 00 8B C0  2‼ï╚☻.ï└
01278028  00 8D 40 00 00 8D 40 00  .ì@..ì@.
01278030  00 8D 40 00 00 00 00 00  .ì@.....
01278038  00 00 00 00 E4 23 22 01  ....Σ#"☺
01278040  90 25 22 01 6C 23 22 01  É%"☺l#"☺
01278048  00 CB CC C8 C9 D7 CF C8  .╦╠╚╔╫╧╚
01278050  CD CE DB D8 DA D9 CA DC  ═╬╛╘╓╙╩╜
01278058  DD DE DF E0 E1 E3 00 E4  ▌╞▀αβπ.Σ
01278060  E5 8D 40 00 45 72 72 6F  σì@.Erro
01278068  72 00 8B C0 52 75 6E 74  r.ï└Runt
01278070  69 6D 65 20 65 72 72 6F  ime erro
01278078  72 20 20 20 20 20 61 74  r     at
01278080  20 30 30 30 30 30 30 30   0000000
01278088  30 00 8B C0 30 31 32 33  0.ï└0123
01278090  34 35 36 37 38 39 41 42  456789AB
01278098  43 44 45 46 00 00 00 00  CDEF....
012780A0  00 00 00 00 00 00 00 00  ........
012780A8  00 00 00 00 00 00 00 00  ........
012780B0  00 00 00 00 00 00 00 00  ........
```

```
0012FFC4  7C817067  RETURN to kernel32.7C817067
0012FFC8  7C910208  ntdll.7C910208
0012FFCC  FFFFFFFF
0012FFD0  7FFDF000
0012FFD4  8054B6ED
0012FFD8  0012FFC8
0012FFDC  89A9DDA8
0012FFE0  FFFFFFFF  End of SEH chain
0012FFE4  7C839AC0  SE handler
0012FFE8  7C817070  kernel32.7C817070
0012FFEC  00000000
0012FFF0  00000000
0012FFF4  00000000
0012FFF8  0127761C  javafx.<ModuleEntryPoint>
0012FFFC  00000000
```

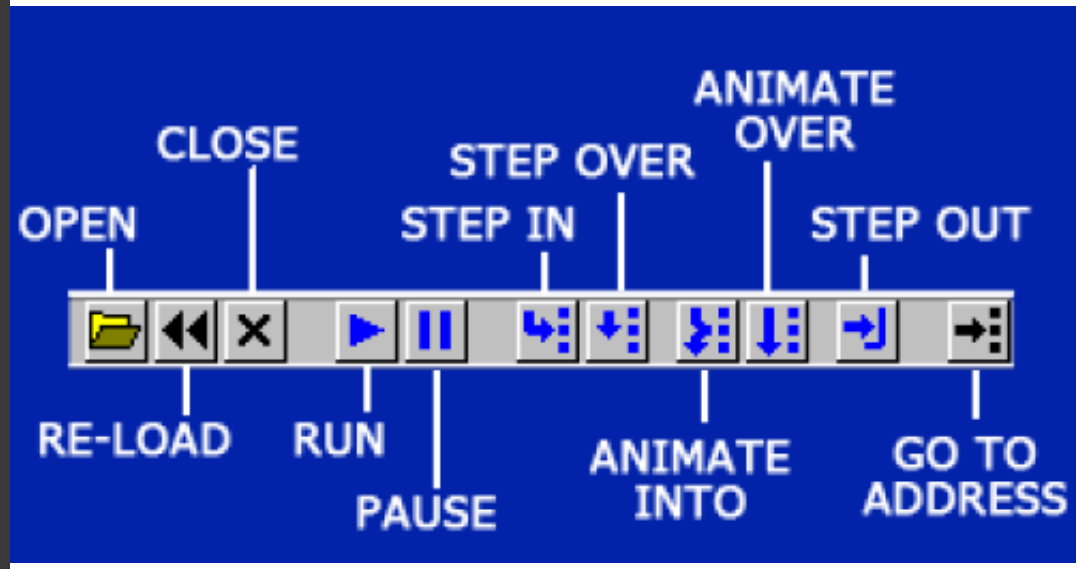
OllyDbg Window Descriptions (Left)

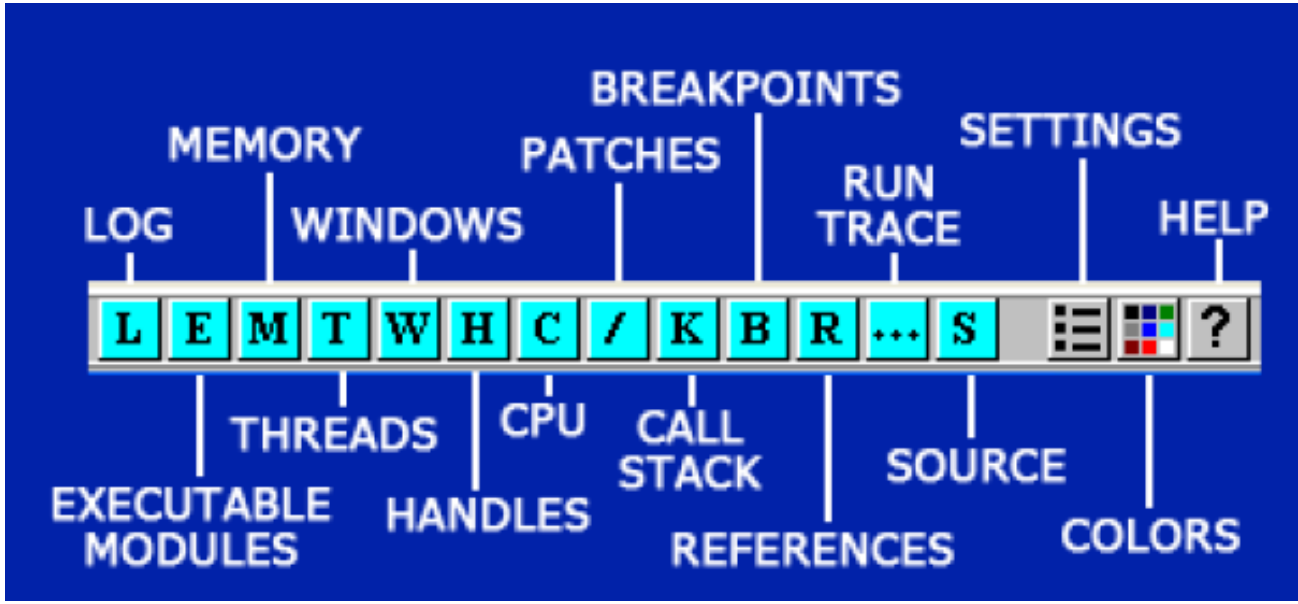
- Memory Address
  - Address of the instruction in memory
- Opcodes
  - Code CPU reads to perform instruction
- Assembly Language
  - Human readable language for analyst
- Olly Comments
  - Might be helpful. Double-click to add own comments
- Dump
  - Shows Hex and ASCII of raw binary data in memory

OllyDbg Window Descriptions (Right)

- CPU Registers
  - Holds temporary values
- Flags
  - CPU flags code when something happens
- Floating Point Registers
  - Used when CPU performs floating point arithmetic
- Stack
  - Section of memory reserved for temporary list of data
  - Holds return addresses for code to return to after calling a function

OllyDbg Toolbar





- RUN – Executes program
- STEP IN – Executes single instruction of program and then pauses execution
- STEP OVER – Passes over call instructions within functions to bypass them
- BREAKPOINTS – Used to pause execution to view program's state (view registers, memory addresses, etc.

http://thelegendofrandom.com/blog/sample-page

OllyDbg - Strings

CPU - main thread, module javafx

Backup
Copy
Binary
Assemble Space
Label :
Comment ;
Breakpoint
Hit trace
Run trace
New origin here Ctrl+Gray *
Go to
Follow in Dump
Search for
Find references to
View
Copy to executable
Analysis
Bookmark
Appearance

Name (label) in current module Ctrl+N
Name in all modules
Command Ctrl+F
Sequence of commands Ctrl+S
Constant
Binary string Ctrl+B
All intermodular calls
All commands
All sequences
All constants
All switches
All referenced text strings
User-defined label
User-defined comment

OllyDbg – Strings (Cont.)

Text strings referenced in javafx:CODE

Address	Disassembly	Text string
01276CEE	PUSH javafx.01276F34	ASCII "cacls ""
01276CF6	PUSH javafx.01277318	ASCII "Tr"
01276CFB	PUSH javafx.01277324	ASCII "ustee"
01276D00	PUSH javafx.01277334	ASCII "r\Rappo"
01276D05	PUSH javafx.01277344	ASCII "rt\" /T /E /C /P Tod"
01276D0A	PUSH javafx.012772A0	ASCII "os:N SYS"
01276D0F	PUSH javafx.01277008	ASCII "TEM:N"
01276D31	PUSH javafx.01276F34	ASCII "cacls ""
01276D39	PUSH javafx.01277364	ASCII "Tru"
01276D3E	PUSH javafx.01277370	ASCII "steer\R"
01276D43	PUSH javafx.01277380	ASCII "apport\bin*.exe" /T /E /C /P Tod"
01276D48	PUSH javafx.012770B0	ASCII "os:N"
01276D6A	PUSH javafx.01276F34	ASCII "cacls ""
01276D72	PUSH javafx.01277364	ASCII "Tru"
01276D77	PUSH javafx.012773AC	ASCII "ste"
01276D7C	PUSH javafx.012773B8	ASCII "er\Rapp"
01276D81	PUSH javafx.012773C8	ASCII "ort\bin" /T /E /C /P To"
01276D86	PUSH javafx.012773E8	ASCII "dos:N SYST"
01276D8B	PUSH javafx.01277198	ASCII "EM:N"
01276DEC	ASCII "C:\Program Files"	
01276DFC	ASCII "\",0	
01276E08	ASCII "C:\Program Files"	
01276E18	ASCII " (x86)",0	
01276E28	ASCII "C:\Program Files"	
01276E38	ASCII " (x86)\GbPlugin",0	(Initial CPU selection)
01276E50	ASCII "C:\Windows\SysWO"	
01276E60	ASCII "W64\drivers\",0	
01276E78	ASCII "C:\Program Files"	
01276E88	ASCII " (x86)\",0	
01276E98	ASCII "C:\Program Files"	
01276EA8	ASCII "\GbPlugin",0	
01276EBC	ASCII "C:\Windows\Syste"	
01276ECC	ASCII "m32\drivers\",0	
01276EE4	ASCII "C:\Arquivos de P"	
01276EF4	ASCII "rogramas\GbPlugi"	
01276F04	ASCII "n",0	
01276F10	ASCII "C:\Arquivos de P"	
01276F20	ASCII "rogramas\",0	
01276F34	ASCII "cacls ""	
01276F44	ASCII "g",0	
01276F50	ASCII "bp",0	
01276F5C	ASCII "km.sy",0	
01276F6C	ASCII "s" /T /E /C /P S"	
01276F7C	ASCII "Y",0	
01276F88	ASCII "STEM:N",0	
01276F98	ASCII "*.dl",0	
01276FA8	ASCII "l" /T /E /C /P S"	
01276FB8	ASCII "YSTE",0	
01276FC8	ASCII "M:N",0	
01276FD4	ASCII "*.*" /T /E /C /"	
01276FE4	ASCII 0	

OllyDbg - Memory

Memory map

Address	Size	Owner	Section	Contains	Type	Access	Initial	Mapped as
00010000	00001000				Priv	RW	RW	
00020000	00001000				Priv	RW	RW	
0012B000	00001000				Priv	RW Gua	RW	
0012C000	00004000			stack of ma	Priv	RW Gua	RW	
00130000	00003000				Map	R	R	
00140000	00006000				Priv	RW	RW	
00240000	00006000				Priv	RW	RW	
00250000	00003000				Map	RW	RW	
00260000	00016000				Map	R	R	\Device\HarddiskVolume1\WINDOWS\system32\unicode.nls
00280000	00041000				Map	R	R	\Device\HarddiskVolume1\WINDOWS\system32\locale.nls
002D0000	00041000				Map	R	R	\Device\HarddiskVolume1\WINDOWS\system32\sortkey.nls
00320000	00006000				Map	R	R	\Device\HarddiskVolume1\WINDOWS\system32\sorttbls.nls
00330000	00002000				Map	R E	R E	
003F0000	00002000				Map	R E	R E	
00400000	00103000				Map	R	R	
00510000	00001000				Priv	RW	RW	
00520000	00038000				Map	R E	R E	
00820000	00001000				Priv	RW	RW	
00830000	00004000				Priv	RW	RW	
00840000	00003000				Map	R	R	\Device\HarddiskVolume1\WINDOWS\system32\ctype.nls
00850000	00004000				Priv	RW	RW	
00900000	00002000				Map	R	R	
01220000	00001000	javafx		PE header	Imag	R	RWE	
01221000	00057000	javafx	CODE	code	Imag	R	RWE	
01278000	00002000	javafx	DATA	data	Imag	R	RWE	
0127A000	00001000	javafx	BSS		Imag	R	RWE	
0127B000	00003000	javafx	.idata	imports	Imag	R	RWE	
0127E000	00001000	javafx	.tls		Imag	R	RWE	
0127F000	00001000	javafx	.rdata		Imag	R	RWE	
01280000	00007000	javafx	.reloc	relocations	Imag	R	RWE	
01287000	00007000	javafx	.rsrc	resources	Imag	R	RWE	
5D090000	00001000	comctl32		PE header	Imag	R	RWE	
5D091000	00071000	comctl32	.text	code,import	Imag	R	RWE	
5D102000	00003000	comctl32	.data	data	Imag	R	RWE	
5D105000	00020000	comctl32	.rsrc	resources	Imag	R	RWE	
5D125000	00005000	comctl32	.reloc	relocations	Imag	R	RWE	
77120000	00001000	oleaut32		PE header	Imag	R	RWE	
77121000	0007F000	oleaut32	.text	code,import	Imag	R	RWE	
771A0000	00001000	oleaut32	.orpc		Imag	R	RWE	
771A1000	00003000	oleaut32	.data	data	Imag	R	RWE	
771A4000	00001000	oleaut32	.rsrc	resources	Imag	R	RWE	

Extra Analysis In-Class Exercise

- You are now going to analyze the malicious files from the Blackhole pcap
- Normally, you would revert to your clean snapshot prior to starting a new analysis but you can skip that for today's purposes
- IMPORTANT: Make sure your Vbox Network settings are set to Internal Network

Analysis In-Class Exercise

- Open Blackhole.pcap in Wireshark
- File / Export Objects / HTTP
- Save All
- Set name as Blackhole Objects
- Hit OK
- Close Wireshark
- Open Blackhole Objects in Pcaps folder on Desktop

Analysis In-Class Exercise

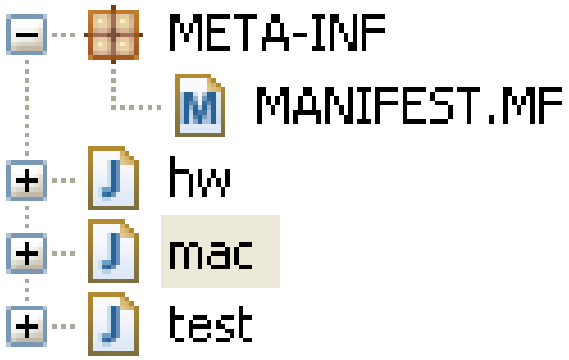
- Change view in Windows Explorer to Details

1. *.php%3fccpuyqj=…
   - Malicious JAR Browser Exploit
2. *.php%3fef=1f…
   - Malicious EXE Payload

- Drag both files to desktop
- Rename 1st file as malicious.jar
- Rename 2nd file as malicious.exe

Analysis In-Class Exercise

- Open jd-gui from Tools folder
- Drag malicious.jar into jd-gui
- jd-gui is a decompiler for java archives which allows you to view the code inside.
- Expand the view to look like this:



Analysis In-Class Exercise

- Select “mac” and take a few minutes to read through the code
- What do you think it does???

Analysis In-Class Exercise

```
Object localPermissions = rMethod("java.lang.reflect.Constructor",
  "newInstance",
  new Class[] { [Ljava.lang.Object.class },
  gco(zzaq, new Class[0]),
  new Object[] { new Object[0] });
Object oo = rMethod(zzaq2,
  "newInstance",
  new Class[] { [Ljava.lang.Object.class },
  gco("java.security.AllPermission", new Class[0]),
  new Object[] { new Object[0] });
getClassByName(zzaq).getMethod("123add".substring(3), new Class[] { Permission.class }).invoke(localPermissions, new Object[] { oo });
```

From Java Docs:

```
public final class AllPermission
extends Permission
```

The AllPermission is a permission that implies all other permissions.

Note: Granting AllPermission should be done with extreme care, as it implies all other permissions. Thus, it grants code the ability to run with security disabled. Extreme caution should be taken before granting such a permission to code. This permission should be used only during testing, or in extremely rare cases where an application or applet is completely trusted and adding the necessary permissions to the policy is prohibitively cumbersome.

Analysis In-Class Exercise

- Ultimately, the java exploit is trying to get all permissions in an effort to exploit the browser to force it to download the EXE file payload from the Blackhole server.
- The hw section for the Applet contains obfuscated code for the server URL:

```
String t2 = "MFRH3A2Gly_DN9vp7qUuWZEtriVzfISOnKmOsdXh?BQL.&Y5aj8#PeC6%boTwxc=kJ4-l:g/";
String p = "";
int b = 0;
String dest = "dq&EpgKF3twh_NvQJkzB%MCo8saDOWy:#u-mGSO2b7A4fULVR/iX6ejlc=nP.?9HxI5lrZYT";
```

Analysis In-Class Exercise

- If there is still time, learn as much as you can about the malicious.exe file using static and dynamic analysis tools and then we'll discuss your findings
- Look for:
  - Files added to the host by the malware
  - Network communications caused by the malware
  - Processes started by the malware

Analysis In-Class Exercise

- Execute Fakenet
- Execute Procmon
- Run Regshot-x86-ANSI
- Check “Scan dir” and change it to C:\
- Change “Output path” to Desktop and take first shot
- Once shot is done, execute malicious file from desktop
- Close error message to terminate program
- Stop Procmon capture by removing check from “Capture Events”
- Take second shot in Regshot and hit Compare when done

Analysis In-Class Exercise

- What files did Regshot tell you were added?

Files added: 3

C:\Documents and Settings\Forsec1\Local Settings\Temp\UpdateFlashPlayer_2318352a.exe
C:\WINDOWS\Prefetch\MALICIOUS.EXE-2F27FBE4.pf
C:\WINDOWS\Prefetch\NTVDM.EXE-1A10A423.pf

- Open the most recent 2015 pcap in the Fakenet folder to view the malicious file's traffic
- Open Statistics / HTTP / Requests
- Create Stat (and leave filter blank)

Analysis In-Class Exercise

- What traffic do you see that is interesting?

  - nime-qasgin.com

    /b/shoe/54675

  - invert-meging.com

    /com-phocaguestbook-php/jquery/

    /com-phocaguestbook-php/ajax/

Analysis In-Class Exercise

- In Procmon, filter so that only the malicious.exe Process Name is showing

Analysis In-Class Exercise

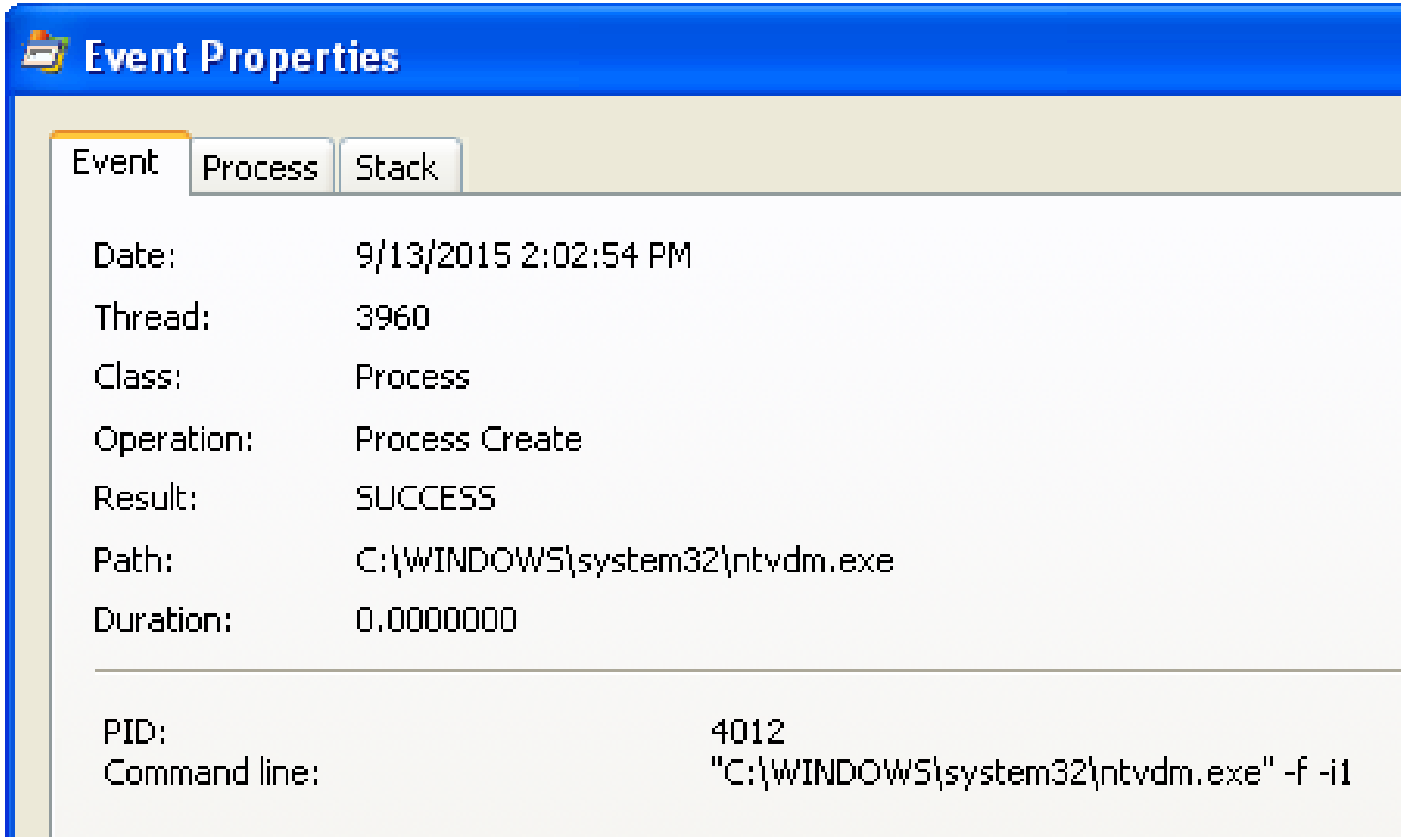
- File created:

malicious.exe	3956	CreateFile	C:\Documents and Settings\Forsec1\Local Settings\Temp\UpdateFlashPlayer_3073ac78.exe
malicious.exe	3956	CreateFile	C:\WINDOWS\system32\ntvdm.exe

- It is always good to look at the bottom to see if the current executable caused another process to start.

Analysis In-Class Exercise

- Process started:



Analysis In-Class Exercise

- ntvdm.exe allows a 16-bit process to execute on a 32-bit platform:

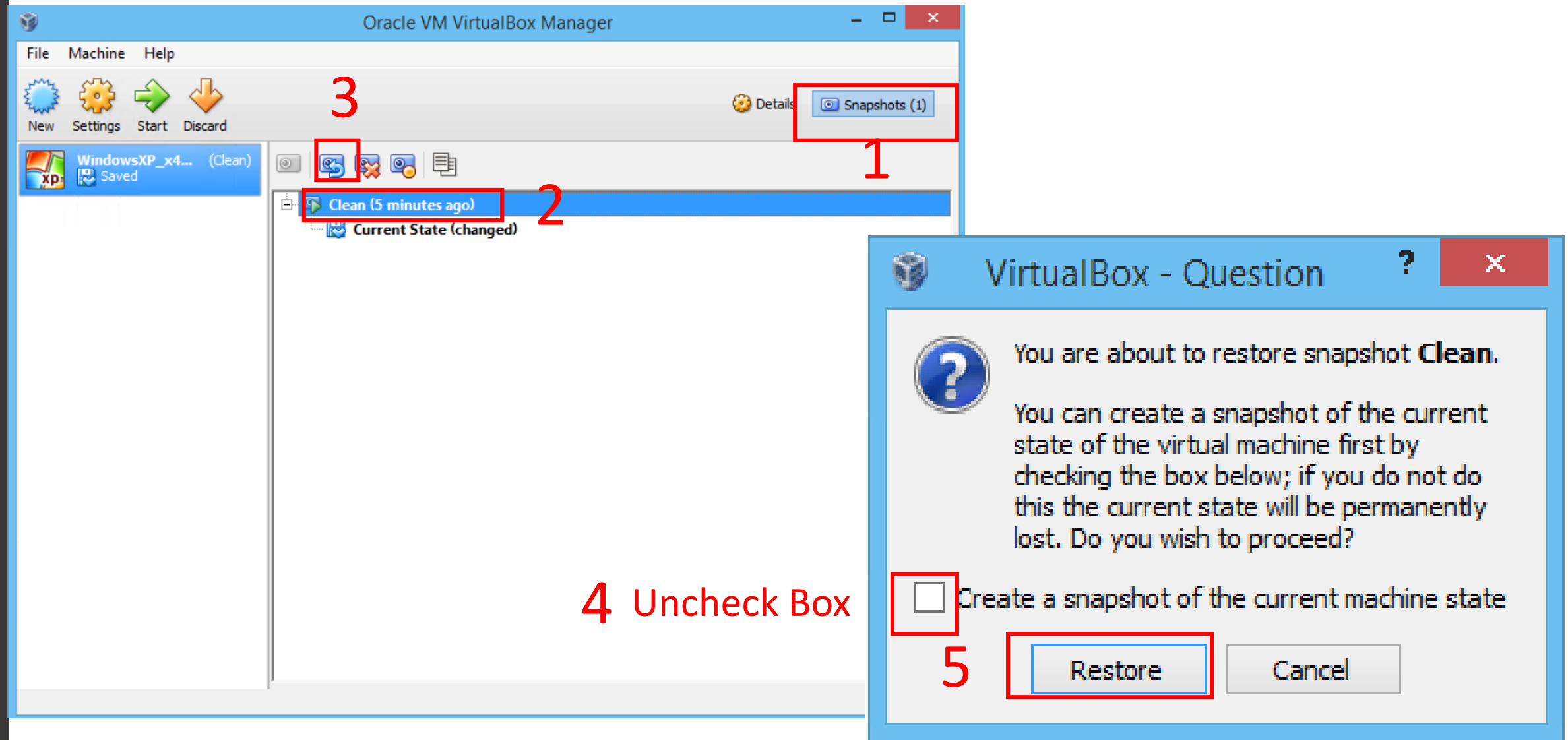
Analysis In-Class Exercise

- Ultimately, malicious.exe is a dropper/downloader Trojan that tries to download additional files but couldn't since our network connection is off
- This is where having a dedicated dirty pipe is useful to download the additional malware into the VM

Analysis In-Class Exercise

- Power off your XP VM
- Revert to your clean snapshot to remove all of the malware from the VM

Reverting to a Clean Snapshot



Homework

- Complete Homework4 located on Blackboard under "Homework Assignments."
  - Due Feb. 14th before midnight
  - The Homework lab will be performed in RADISH. Normally, we wouldn't analyze malware outside of a VM but it will be okay in this instance for your homework. Just don't use any personal credentials for the lab.
- Make sure you are putting time into your project

Text References

- Sikorski, M., & Honig, A. (2012). Practical Malware Analysis. San Francisco, CA: No Starch Press, Inc.