So You Want To Analyze Malware?
Tools, Techniques, and Mindset
Introduction

Who, What, Why?
Introduction

• Me – Wes Brown
  – Software and Systems Hacker
    • Fond of Lisp-based and Functional Languages
    • Developed Lisp dialect with Scott Dunlop
      – Mosquito Lisp
      – Evolved into Wasp Lisp
  – Security Researcher and Malware Analyst
    • MOSREF – uses Mosquito Lisp for a remote command and execution framework
    • Malware Analyst – analyzed thousands of samples
  – Security Consultant
    • Penetration Testing
    • Code Review
    • SDL
  – IOActive
Agenda

- Motivations behind Malware Analysis
- Mindset behind Malware and Analysis
- Trends in Malware
- Building a Malware Lab
- Tools for Malware Analysis
- Analysis Walkthrough
Motivations behind Malware and Analysis

• Why Analyze Malware?
  – Better understanding of threats to protect network
    • Defender
  – To write software that detects malware
    • Tools for Defender
  – Aesthetic admiration
    • Admiration of Techniques
  – Writing a better mousetrap
    • Financial Gain

• Why Malware?
  – Financial gain
    • Follow the money
  – Political agenda
  – Used to be for the challenge and pranks
What Makes A Good Malware Analyst?

- **Mindset**
  - Meticulous data collection
  - Logical processes
  - Thinks outside the box
  - Tenacious

- **Technical**
  - Good systems understanding
  - Good understanding of programming
  - Some reverse engineering skills

- **Attitude**
  - Ties into motivations discussed earlier
Trends in Malware
Past, Present, and Future
Attack Vectors

• In the Ancient Past
  – Viruses via floppy disks
  – Downloaded via FTP or BBS’es

• Past
  – Systems level
  – Exploitation of remote services, worms
  – System protections an NAT/Firewalls made this difficult

• Now
  – System is only as strong as its weakest link
Human Factor

• In the past, attacks were mainly technical.
  – Attackers searched for network or systems level vulnerabilities.
  – Automatic exploitation and spread.
  – Humans not involved in the attack cycle.

• In the present, exploit the human.
  – Spam email
  – Compromise a legitimate site.
    • “Drive by” site
    • Human visits compromised site, is compromised.
  – Advertising attacks
    • Especially at shadier sites such as P2P trackers.
  – Goal is to get the initial injection vector in.
    • Once vector is in, payload can be sent, and network is compromised.
Attacking through Social Networks

- Social Networks
  - Flickr
  - Facebook
  - Twitter
  - Myspace
  - Etc

- File sharing
  - Torrents
  - Warez
  - P2P

- Highly connected network
- Massive information sharing
- Rich media content
Internationalization of Malware

• Formerly, English-targeted samples.
  – Easy to conduct a strings search on.

• Cultural assumptions of what Malware is.
  – Varies from region to region.
  – One man’s anti-cheating toolkit is another man’s rootkit.
    • Punkbuster
    • Korean and Chinese games

• What should it be flagged at?
  – Suspicious?
  – White list?
  – Malware?
Current Attack Lifecycle

• Initial payload is small
• Initial checks
  – Mutex, OS Version, Keyboard, location
  – Conficker A didn’t infect systems with Ukrainian Keyboard
• Payload is downloaded
• Backdoor/trojan/infect
• Contacts command and control server for tasks
• May fall back to secondary C&C
• Dynamically generate rendezvous point
• Conficker quietly spreads internally and waits before phoning home
Current Obfuscation Techniques

Staying on the System
Obfuscation

- Obfuscation used to confuse analysis
  - Antivirus signatures
  - Static analysis – decompilers
  - Dynamic analysis – tracing, debugging, inspection
- Obfuscation used legitimately for DRM systems
  - Hide important logic to slow reverse engineering
- Race to Zero Competition
  - Highlighted ineffectiveness of AV
Basic Techniques

- Polymorphism and Packers
  - UPX, Armadillo or custom packer
- Simple Debugger checks
  - IsDebuggerPresent()
- Jumping into data/ middle of instructions
- Encoding strings/values
- Manipulating imports
- Corrupting PE Header
  - Bad LoaderFlags
  - Bad NumberOfRvaAndSizes
- Section Header Stuff
  - Enormous bogus sections
  - Overlapping sections
Basic Techniques (cont.)

- Junk code
  - Spaghetti assembly
- SEH
  - Exception handler patches memory
  - Access to application context structure -> Erase Hardware debug Registers
Advanced Techniques

- Metamorphic malware
- Custom virtual machines
  - Polymorphic instruction sets
- Encryption
  - Corrupting PE Header, use corrupt data as key
- Instruction Timing
  - Model Specific Register (MSR), counts clock cycles
  - RDTSC instruction, moves timestamp to EDX and EAX
Advanced Techniques (cont.)

- Debugging register tricks
  - Trampolines pass shared stack via debug registers

- Breakpoint detection
  - Before calling API, check first few instructions breakpoints

- VMWare detection
  - VMWare Tools, Network card, hidden APIs

- Random note: Malicious JavaScript can only be fetched once
Custom Virtual Machines

- Purpose is to complicate static analysis by adding additional layer of translation
- P-Code machine (Pseudo-code)
- Create a software CPU
- Soft registers and pseudo language
- Mapping between pseudo language and real instructions
  - Mapping happens at runtime
- Makes static analysis very difficult
- Must run the system and step through things
- Make your Vmcode self modifying
- Really evil = Instruction set mapping changes after each instruction
Building a Malware Lab
Tools for Analysis
Malware Lab

- **Virtualization Platform**
  - Multi-core CPUs are cheap
  - Windows images can be reverted in seconds.
  - Can run dozens of Windows images.
  - Easy to audit
    - Use Copy on Write disk images

- **Must not be on any network but its own.**
  - Airgapped.
  - Prevents inadvertent contamination and information leakage.

- **Dynamic Internet Connection**
  - Preferrably a consumer-level connection.
  - Reissue new IP addresses via DHCP lease.
  - Prevents blacklists against
Virtualization Platform

- VMware
  - Why Vmware?
    - Stable.
    - Well-known.
    - Tools to analyze Vmware suspend images
    - Vmware ESXi is free, bare metal virtualization.
  - Fatal Flaw
    - Lowest common denominator.
    - Malware actively detects Vmware.
      - Virtualization drivers detectable.
      - Easy to detect.
        » Put value 10 (0x0a) in the ECX register, and put 0x564D5868 in the EAX register. Read a dword from 0x5658.
      - Exploits to break out of Vmware sandbox now.
  - Recommend strongly against using Vmware for a Malware Lab
Virtualization Platform (cont’d)

- Xensource
  - Payware
    - Now has a free product to compete with VMware ESXi
    - Yay competition!
  - Nicely packaged bare-metal virtualizer.
  - Good performance.
  - Excellent Copy-on-Write support

- Qemu
  - Roll your own virtualization platform
  - OpenSource
  - Slower than the others.
Neat Virtualization Tricks

• Serial Debugging
  – Debugger and Debuggee VMs with virtual serial connection.
  – Very handy for kernel debugging with tools such as WinDBG.

• Copy on Write
  – Original VM disk image is unmodified.
  – All changes are made to a separate file.
  – Can mount delta images and examine differences to see what malware changed.

• Memory Image
  – State of memory can be snapshotted while malware is run, and then disassembled and debugged.

• Fast reversion of images
  – Useful for analyzing thousands of samples in a day.
Database (aka, store everything!)

- **Database**
  - Needed to store data from automatic and manual analysis.
  - Malware analysis is far more useful with a corpus to compare against.
  - The more data we have on characteristics, the more we are able to do a determination of whether it is malware.
  - Reverse engineering is expensive in terms of man-power to do.
  - Identify characteristics and understand malware to allocate reverse engineering where it is worthwhile to.

- **Corpus**
  - Store actual malware sample.
  - Store all known characteristics.
  - Store network traces.
  - Store static forensics.
Obtaining Malware to Analyze

• Be an anti-virus or anti-malware software vendor.
  – Set up your software agent to automatically send back unknown samples.
  – Thousands of samples a day!

• Join an existing antimalware intelligence group.
  – Honeynet Project
  – Sandnet

• Build your own honeynet.
  – Collect malware samples from exploits.

• Beg, borrow, steal.
  – Obtain a feed from someone.
  – Offer a feed in return.
Additional Tools

- **Debuggers**
  - WinDBG
  - IDA
  - Ollydbg

- **Tracers**
  - Process Monitor (regmon, filemon)
  - Detours
  - Third party: apimonitor, strace

- **Unpackers**
  - PeID
  - Import rebuilders
Analysis Walkthrough

Dynamic and Static
Analysis Walkthrough

• Version of Sality family
• From the network logs we know some behavior
  – Slowly spreads internally
  – Outbound connections on high number ports
  – HTTP requests
  – Not detected by antivirus
• Initial samples
  – Four executables
  – Random filenames starting with “win”
  – Same size, different checksums
Process Monitor

- External behavior highlights what to look for during static analysis
  - Ex: strings of URLs, registry keys, file names
- A lot of what you’ll see is general noise as application loads libraries, reads registry keys, starts threads, accesses files
- Focus on RegSetValue for fast info
Process Monitor Video
<table>
<thead>
<tr>
<th>Path</th>
<th>Detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Setting...</td>
<td>Type: REG_DWORD, Length: 4, Data: 0</td>
</tr>
<tr>
<td>HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\policies\s...</td>
<td>Type: REG_DWORD, Length: 4, Data: 0</td>
</tr>
<tr>
<td>HKLM\System\CurrentControlSet\Services\SharedAccess\Param...</td>
<td>Type: REG_SZ, Length: 156, Data: C:\Documents and</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914-993627007\1768776769</td>
<td>Type: REG_DWORD, Length: 4, Data: 5</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914-993627007-757413758</td>
<td>Type: REG_DWORD, Length: 4, Data: 0</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914-993627007\1011363011</td>
<td>Type: REG_DWORD, Length: 4, Data: 0</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914-993627007\1514827516</td>
<td>Type: REG_DWORD, Length: 4, Data: 30</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914-993627007\253949253</td>
<td>Type: REG_DWORD, Length: 4, Data: 182</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914-993627007\2022726022</td>
<td>Type: REG_SZ, Length: 726, Data: 0500687474703A2F</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914-993627007\503464505</td>
<td>Type: REG_SZ, Length: 514, Data: BE0CE72B58D4AE</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914\A1_0</td>
<td>Type: REG_DWORD, Length: 4, Data: 3432392762</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914\A2_0</td>
<td>Type: REG_DWORD, Length: 4, Data: 5517</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914\A3_0</td>
<td>Type: REG_DWORD, Length: 4, Data: 17000001</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914\A4_0</td>
<td>Type: REG_DWORD, Length: 4, Data: 0</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914\A1_1</td>
<td>Type: REG_DWORD, Length: 4, Data: 659249704</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914\A2_1</td>
<td>Type: REG_DWORD, Length: 4, Data: 1768780236</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914\A3_1</td>
<td>Type: REG_DWORD, Length: 4, Data: 1752039936</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914\A4_1</td>
<td>Type: REG_DWORD, Length: 4, Data: 1768776769</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914\A1_2</td>
<td>Type: REG_DWORD, Length: 4, Data: 2523696295</td>
</tr>
<tr>
<td>HKCU\Software\Administrator914\A2_2</td>
<td>Type: REG_DWORD, Length: 4, Data: 3537558799</td>
</tr>
</tbody>
</table>
RegSetValue Standard Stuff

- Adds self to Firewall Policy Authorized Applications List
- GlobalUserOffline -> 0
  - Switches to online if was “Work Offline” mode
- EnableLUA -> 0
  - Turn off User Access Control for Administrator
RegSetValue Interesting Stuff

- HKCU\Software\Administrator914\-993627007\2022726022
- Size 726
- Value:
  0500687474703A2F2F61736A6469776575723837777364636
  E622E696E666F2F74616E67612E67696600687474703A2F2
  F7065646D656F3232326E622E696E666F2F74616E67612E6
  7696600687474703A2F2F676F6E646F6C697A6F313834383
  32E696E666F2F74616E67612E67696600687474703A2F2F7
  46563686E6963616E2E772E696E74657269612E706C2F746
  16E67612E67696600687474703A2F2F707A726B2E72752F6
  96D672F6C6F676F342E676966
RegSetValue Interesting Stuff

- Decodes to:

  http://asjdiweur87wsdcnb.info/tanga.gif
  http://pedmeo222nb.info/tanga.gif
  http://gondolizo18483.info/tanga.gif
  http://technican.w.interia.pl/tanga.gif
  http://pzrk.ru/img/logo4.gif
<table>
<thead>
<tr>
<th>File</th>
<th>Value</th>
<th>Key</th>
<th>Type</th>
<th>Length</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>winqlxip.exe</td>
<td>3756 ReqSetValue</td>
<td>HKCU\Software\Administrator914A1_0</td>
<td>REG_DWORD</td>
<td>4</td>
<td>3432392762</td>
</tr>
<tr>
<td>winqlxip.exe</td>
<td>3756 ReqSetValue</td>
<td>HKCU\Software\Administrator914A2_0</td>
<td>REG_DWORD</td>
<td>4</td>
<td>5517</td>
</tr>
<tr>
<td>winqlxip.exe</td>
<td>3756 ReqSetValue</td>
<td>HKCU\Software\Administrator914A3_0</td>
<td>REG_DWORD</td>
<td>4</td>
<td>17000001</td>
</tr>
<tr>
<td>winqlxip.exe</td>
<td>3756 ReqSetValue</td>
<td>HKCU\Software\Administrator914A4_0</td>
<td>REG_DWORD</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>winqlxip.exe</td>
<td>3756 ReqSetValue</td>
<td>HKCU\Software\Administrator914A1_1</td>
<td>REG_DWORD</td>
<td>4</td>
<td>659249704</td>
</tr>
<tr>
<td>winqlxip.exe</td>
<td>3756 ReqSetValue</td>
<td>HKCU\Software\Administrator914A2_1</td>
<td>REG_DWORD</td>
<td>4</td>
<td>1758780236</td>
</tr>
<tr>
<td>winqlxip.exe</td>
<td>3756 ReqSetValue</td>
<td>HKCU\Software\Administrator914A3_1</td>
<td>REG_DWORD</td>
<td>4</td>
<td>1752039936</td>
</tr>
<tr>
<td>winqlxip.exe</td>
<td>3756 ReqSetValue</td>
<td>HKCU\Software\Administrator914A4_1</td>
<td>REG_DWORD</td>
<td>4</td>
<td>1758776769</td>
</tr>
<tr>
<td>winqlxip.exe</td>
<td>3756 ReqSetValue</td>
<td>HKCU\Software\Administrator914A1_2</td>
<td>REG_DWORD</td>
<td>4</td>
<td>2523696295</td>
</tr>
<tr>
<td>winqlxip.exe</td>
<td>3756 ReqSetValue</td>
<td>HKCU\Software\Administrator914A2_2</td>
<td>REG_DWORD</td>
<td>4</td>
<td>3537558799</td>
</tr>
<tr>
<td>winqlxip.exe</td>
<td>3756 ReqSetValue</td>
<td>HKCU\Software\Administrator914A3_2</td>
<td>REG_DWORD</td>
<td>4</td>
<td>3554258627</td>
</tr>
<tr>
<td>winqlxip.exe</td>
<td>3756 ReqSetValue</td>
<td>HKCU\Software\Administrator914A4_2</td>
<td>REG_DWORD</td>
<td>4</td>
<td>3537553538</td>
</tr>
<tr>
<td>winqlxip.exe</td>
<td>3756 ReqSetValue</td>
<td>HKCU\Software\Administrator914A1_3</td>
<td>REG_DWORD</td>
<td>4</td>
<td>721497331</td>
</tr>
<tr>
<td>winqlxip.exe</td>
<td>3756 ReqSetValue</td>
<td>HKCU\Software\Administrator914A2_3</td>
<td>REG_DWORD</td>
<td>4</td>
<td>1011366222</td>
</tr>
</tbody>
</table>

Kill off the malware process and a little while later....
Thread Injection

- You can actually see the thread injection

```
winqilxhp.exe 3756 ReqCloseKey <INVA... 3744
winqilxhp.exe 3756 ReqCloseKey <INVA... 3744
Explorer.EXE 1996 Thread Create Thread ID: 3784 3744
Explorer.EXE 1996 Thread Create Thread ID: 3532 3744
jusched.exe 300 Thread Create Thread ID: 3252 3744
jusched.exe 300 Thread Create Thread ID: 3248 3744
wsctfny.exe 460 Thread Create Thread ID: 3080 3744
wsctfny.exe 460 Thread Create Thread ID: 3084 3744
GoogleToolb... 468 Thread Create Thread ID: 1912 3744
GoogleToolb... 468 Thread Create Thread ID: 3796 3744
ctfmon.exe 496 Thread Create Thread ID: 1900 3744
ctfmon.exe 496 Thread Create Thread ID: 2368 3744
TPAutoConn... 504 Thread Create Thread ID: 1848 3744
TPAutoConn... 504 Thread Create Thread ID: 1816 3744
winqilxhp.exe 3756 Thread Create Thread ID: 1976 3744
winqilxhp.exe 3756 Thread Create Thread ID: 1960 3744
winqilxhp.exe 3756 ReqCloseKey <INVA... 3744
```
No more safeboot!

| Explorer.EXE  | 1996 | ReqDeleteKey | HKLM\System\CurrentControlSet\Control\SafeBoot\Minimal\AppMqmt |
| Explor.. | 1996 | ReqDeleteKey | HKLM\System\CurrentControlSet\Control\SafeBoot\Minimal Base |
| Explor.. | 1996 | ReqDeleteKey | HKLM\System\CurrentControlSet\Control\SafeBoot\Minimal\Base Boot Ext |
| Explor.. | 1996 | ReqDeleteKey | HKLM\System\CurrentControlSet\Control\SafeBoot\Minimal\Boot file syst |
| Explor.. | 1996 | ReqDeleteKey | HKLM\System\CurrentControlSet\Control\SafeBoot\Minimal\CryptSvc |
| Explor.. | 1996 | ReqDeleteKey | HKLM\System\CurrentControlSet\Control\SafeBoot\Minimal\DcomLaunch |
| Explor.. | 1996 | ReqDeleteKey | HKLM\System\CurrentControlSet\Control\SafeBoot\Minimal\dmadmin |
| Explor.. | 1996 | ReqDeleteKey | HKLM\System\CurrentControlSet\Control\SafeBoot\Minimal\dmboot.sys |
| Explor.. | 1996 | ReqDeleteKey | HKLM\System\CurrentControlSet\Control\SafeBoot\Minimal\dmio.sys |
| Explor.. | 1996 | ReqDeleteKey | HKLM\System\CurrentControlSet\Control\SafeBoot\Minimal\dmload.sys |
| Explor.. | 1996 | ReqDeleteKey | HKLM\System\CurrentControlSet\Control\SafeBoot\Minimal\dmserver |
| Explor.. | 1996 | ReqDeleteKey | HKLM\System\CurrentControlSet\Control\SafeBoot\Minimal\EventLog |

IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES
Some other Things

• See the Libraries it's loading

  - winqilxhp.exe  3756  Load Image  C:\WINDOWS\system32\shell32.dll
  - winqilxhp.exe  3756  Load Image  C:\WINDOWS\system32\comctl32.dll
  - winqilxhp.exe  3756  Load Image  C:\WINDOWS\system32\wsock32.dll
  - winqilxhp.exe  3756  Load Image  C:\WINDOWS\system32\rasapi32.dll
  - winqilxhp.exe  3756  Load Image  C:\WINDOWS\system32\rasman.dll
  - winqilxhp.exe  3756  Load Image  C:\WINDOWS\system32\netapi32.dll
  - winqilxhp.exe  3756  Load Image  C:\WINDOWS\system32\tapi32.dll
  - winqilxhp.exe  3756  Load Image  C:\WINDOWS\system32\rtutils.dll
  - winqilxhp.exe  3756  Load Image  C:\WINDOWS\system32\winmm.dll
  - winqilxhp.exe  3756  Load Image  C:\WINDOWS\system32\msv1_0.dll
  - winqilxhp.exe  3756  Load Image  C:\WINDOWS\system32\sensapi.dll
  - winqilxhp.exe  3756  Load Image  C:\WINDOWS\system32\userenv.dll
  - winqilxhp.exe  3756  Load Image  C:\WINDOWS\system32\urlmon.dll

• Writes System.ini

• Thread heavy >100 threads in 1 minute
Static Analysis and Debugging

- More difficult than simple runtime trace analysis
- Malware is usually packed
- Uses anti-debugging techniques
  - Debugger checks
  - Import table stuff
  - SEH
  - Timing
- Unpack
  - Automated tools, PeID
  - Manually with memdumper
- Fix Imports
- Use Debugger with anti-anti-debugging features
Unpacking

- PEiD Fails
- At least we know it’s UPX (probably)
Manual unpacking

- Entry point at 0x425F30:

<table>
<thead>
<tr>
<th>00425F30</th>
<th>60</th>
<th>PUSHAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>00425F31</td>
<td>E8 000</td>
<td>CALL winqilxh.00425F36</td>
</tr>
<tr>
<td>00425F36</td>
<td>50</td>
<td>PUSH EAX</td>
</tr>
<tr>
<td>00425F37</td>
<td>FECA</td>
<td>DEC DL</td>
</tr>
<tr>
<td>00425F39</td>
<td>EB 01</td>
<td>JMP SHORT winqilxh.00425F3C</td>
</tr>
<tr>
<td>00425F3B</td>
<td>9C</td>
<td>PUSHFD</td>
</tr>
<tr>
<td>00425F3C</td>
<td>8BF5</td>
<td>MOV ESI,EBP</td>
</tr>
</tbody>
</table>

- PUSHAD pushes all registers onto stack
- PUSHAD & POPAD usually surround the packer logic
Manual Unpacking Cont.

- Step the PUSHAD
- Set a hardware access breakpoint on the location of the stack pointer
- Pray
- Continue

Normally you note where its jumping two and dump the process
But its jumping back to the same entry point!
Manual Unpacking Cont.

• Follow the jump

| 00425F30 | 60 | PUSHAD |
| 00425F31 | BE 00004200 | MOV ESI,winqilxh.00420000 |
| 00425F36 | 8DBE 0010FEFF | LEA EDI,DWORD PTR DS:[ESI+FF] |
| 00425F3C | 57 | PUSH EDI |
| 00425F3D | 83CD FF | OR EBP,FFFFFFFF |

• Same 425F30
• Same PUSHAD
• Different Code
• Packed twice!
Manual Unpacking Cont.

- At the second POPAD

| 004260B6 | 61 8D4424 80          | POPAD                  |
| 004260B7 | POPAD                  |
| 004260BB | 6A 00                  |
| 004260BD | 39C4                   |
| 004260BF | ^75 FA                 |
| 004260C1 | 83EC 80                |
| 004260C4 | -E9 E728FEFF           |
| 004260C4 | LEA EAX,DWORD PTR SS:[ESP-80] |
| 004260C4 | PUSH 0                 |
| 004260C4 | CMP ESP,EAX             |
| 004260C4 | JNZ SHORT winqilxh.004260BB |
| 004260C4 | SUB ESP,-80             |
| 004260C4 | JMP winqilxh.004089B0   |

- Looks much better
- Short loop to zero out stack (?)
- Jump to 4089B0
- Dump to new PE file
Dumping

- Used OllyDump to rebuild an unpacked version of the PE file
Fixing imports
Assembly Stuff

- **Mutex**
  ```
push  offset Name ; "S_SERV_v0122ALPHAA275s1"
push  1 ; bInitialOwner
push  0 ; lpMutexAttributes
call  CreateMutexA
  ```

- **Threads**
  ```
push  offset sub_4070FD ; lpStartAddress
push  0 ; dwStackSize
push  0 ; lpThreadAttributes
call  CreateThread
push  eax ; hObject
call  CloseHandle
push  400h ; dwMilliseconds
call  Sleep
  ```

- **Sockets**
  ```
push  35h ; htons(0)
call  htons(0)
mov  word ptr [ebp+name.sa_data], ax
mov  dword ptr [ebp+name.sa_data+2], 0
push  0 ; protocol
push  2 ; type, udp
push  2 ; af, ipv4
call  socket
  ```
Strings

db 'mailc.microsoft.com',0 ; DATA XREF: UPX0:off_40E0C4fo
db 'maila.microsoft.com',0 ; DATA XREF: UPX0:off_40E0C8fo
db 'mailb.microsoft.com',0 ; DATA XREF: UPX0:off_40E0CCfo
db 'smtp.mail.ru',0 ; DATA XREF: UPX0:off_40E0D0fo

db 'Proxy-Connection: close',0Dh,0Ah
db 'Content-type: text/html; unsigned charset=us-ascii',0Dh,0Ah
db 0Dh,0Ah
db '<html><head><title>502 Bad Gateway</title></head>',0Dh,0Ah
db '<body><h2>502 Bad Gateway</h2><h3>Host Not Found or connection fa'
db 'iled</h3></body></html>',0Dh,0Ah,0
align 10h
Analysis Conclusion

• A lot can be learned from simple tracing
• Anti-debugging tricks can slow down reverser significantly
  – Small effort for malware writer
  – Large effort for reverser

• Network analysis
  – Sniff traffic with protocol analyzer
  – Spoof servers to feed same payload
  – Now trace the virus

• Create wrappers to call functions in the malcode
  – Encrypt/decrypt
  – Rendezvous point generation function
Overall Conclusion

• Not as bad as it could be
• Simple tracing/monitoring can give lots of information
• Static analysis of Malware can also yield many clues.
• Storing all bits of data and characteristics in a database can yield large dividends.
• Trend is toward decentralized botnets (P2P)
• New coordination efforts in botnet takedowns
Thank You!

Wes Brown
wbrown@ioactive.com