Official Malware Report

Malware Reverse Engineering part 1 of 2. Static analysis
Table of Contents

1. Scope..................................................................................................................................................... 4
2. Investigation goals ................................................................................................................................ 4
3. Malware samples analyzed ................................................................................................................... 4
4. Malware analysis methodology, software, and secure lab setup .......................................................... 4
5. General function and functionality of the malware............................................................................. 10
6. Behavioral patterns of the malware and local system interaction....................................................... 10
7. Files and registry keys created, modified and accessed ...................................................................... 10
8. Network behavior (including hosts, domains and ip’s accessed)........................................................ 13
9. Time and local system dependant features .......................................................................................... 21
10. Method and means of communication .............................................................................................. 22
11. Original infection vector and propogation methodology ................................................................. 22
12. Use of encryption for storage, delivery and or communication ...................................................... 22
13. Use of self modifying/replicating or encrypted code ........................................................................ 23
14. Any information concerning development of malware (compiler type, packer used, country of origin, author, names/handles, etc.) ......................................................................................... 23
15. Key questions and answers ............................................................................................................... 26
16. Conclusions and recommendations to prevent incident from recurring ....................................... 26
17. Followup actions and lessons learned ............................................................................................ 26
18. REFS..................................................................................................................................................... 26
1. **SCOPE**

This malware report is part 1 of 2. Part 2 will focus heavily on dynamic analysis, determining packers/encryption used and finding original entry point (OEP) of the malware sample, and will utilize IDA Pro, and Immunity de-bugger extensively. We will also bypass anti-debugging, and anti-reversing tactics employed by attackers, and malware authors in part 2. Stay tuned!

This report is an effort to track, categorize, contain, understand root cause and infection vector of said user account/s, networked equipment or computer/s. This report pertains to all incidents reported by TIER II help desk, TIER III engineers, customer complaints or random IT Security audit/finding/pen test.

2. **INVESTIGATION GOALS**

Determine extent of infection, network risk, determine risk of data exposure, figure out infection vector and propogation methods, etc.

3. **MALWARE SAMPLES ANALYZED**

3.1 Win32 Kryptik.YJA trojan variant **40dbdf4b-7db5306a.exe**

**MD5**: f0d0872763058e047922ead2474943ec

**SHA1**: 5629f91e72401440024ec170430e60f50d4f4590

**SHA256**: b811b4089b36660ae089db8a7c61f2d9dc1ebfeb367ac51e55535ec8eaf1d77a

3.2 Location C:\Documents and Settings\**anonymousvictim**\Local Settings\Temp\40dbdf4b-7db5306a.exe

3.3 Moving forward, and for brevity I will be referring to “40dbdf4b-7db5306a.exe” simply as the malware sample. When you read ‘malware sample’ in the remainder of this report, safely assume I am referring to **40dbdf4b-7db5306a.exe** which is the malicious sample used as the basis of this malware report.

3.4 Malware Sample properties. Note the Internet Explorer Developer Tools information recorded, and Original File Name: “iedvtool.dll”

"CompanyName", "Microsoft Corporation"
"FileDescription", "Internet Explorer Developer Tools"
"FileVersion", "8.00.6001.19044 (longhorn_ie8_gdr.110211-1700)"
"InternalName", "iedvtool.dll"
"LegalCopyright", "© Microsoft Corporation. All rights reserved."
"OriginalFilename", "iedvtool.dll"
"ProductName", "Windows® Internet Explorer"
Malware reverse engineering part 1. Static analysis

"ProductVersion", "8.00.6001.19044"

4. MALWARE ANALYSIS METHODOLOGY, SOFTWARE, AND SECURE LAB SETUP

Malware Methodology

4.1 This malware report focuses on malware static analysis but also lightly introduces dynamic analysis to determine if the malware sample is packed, armored, encrypted, and or obfuscated. There is also a very brief introduction to IDA Pro, and Immunity de-bugger.

4.2 Advanced modern malware applications are either protected, obfuscated, encrypted (armoring) and/or packed (the original code is compressed, encrypted or both). This technique is applied in an attempt to evade signature based malware detection, and to hinder the efforts of static analysis by malware analysts by employing anti-reversing, anti-debugging and self-modifying code tactics. This malware sample is no different. The unpacking or decrypting of the
Malware reverse engineering part 1. Static analysis

malware layers remains the most complicated & sophisticated task in the overall process of malware analysis and finding the original entry point (OEP). True analysis of packed malicious binary code can only be performed after the payload is unpacked. Dynamic analysis goes beyond the focus of this paper, and will be the focus of part 2 of this malware report. Stay tuned!

Software

4.3 Software used for the analysis of the malware sample.

1. Winalysis v3.1. Used to snapshot the OS and verify changes to the baseline after the malware sample has been executed.

![Winalysis screenshot](image1)

2. Mandiant Red Curtain v1.0. Look for entropy, packing indication, original entry point (OEP), compiler & packing signatures, digital signatures, and it generates a threat score.

![Mandiant Red Curtain screenshot](image2)

3. Mandiant Find Evil v0.1. Malware discovery tool which uses disassembly to detect packed executables.
Malware reverse engineering part 1. Static analysis

4. Resource Hacker v3.4.0.79. To view/modify Windows executable resources.

5. Sysinternals Suite. All sorts of goodness!

File Monitor.

Process Explorer.
TCP view.

6. Wireshark. Used to capture all network packets, DNS requests, HTTP get requests… etc generated by the malware sample.

7. Malicious domain research & staying anonymous during investigation.
I primarily use a mixture of the following. Tor/TorSOCKS, Privoxy, anonymous.org, hidemyass.com, and/or a VPN connection.

8. Researching malicious Domains, and IP’s.

Query whois records. www.networktools.nl/whois

How many malicious domains are hosted on an IP? www.networktools.nl/reverseip

Is IP listed in SPAM blacklists? www.networktools.nl/rblcheck

GeoIP location search/trace. www.ip-adress.com/ip_tracer/

9. IDA Pro v6.1

10. Immunity De-bugger v1.83

Secure Lab setup.

4.4 VMware workstation v8.0.1 build-528992. Under the guest VM I like to disable drag/drop, and copy/paste. I also set my host firewall to a default DROP/LOG ALL stance for the duration of the malware analysis, and you can also run snort on the host just for paranoia. I like to perform two different analysis. The same malware sample on a physical machine, and one on a virtual machine. I then compare the results and verify if the malware detected or changed its payload if under a VM (red pill) or tried to escape the VM sandbox (which is very possible). That is the reason you should have a dedicated malware machine for these purposes, and never be connected to the internet while analysis is underway. Your host machine can still be infected even if you run your guest machines under NAT/Bridged or host only networking modes. Being paranoid is the only way to survive!
5. **GENERAL FUNCTION AND FUNCTIONALITY OF THE MALWARE**

5.1 This malware sample installs fake antivirus software on the victim machine. It attempts to trick the user with several popups that resemble valid applications warning that the user is infected and that he/she needs to buy the full version of the software in order to be fully protected.

The malware sample’s main purpose is to steal credit card information from the victim. It has very extensive networking capabilities which are detailed in the Network Behavior section 7 of this report.

6. **BEHAVIORAL PATTERNS OF THE MALWARE AND LOCAL SYSTEM INTERACTION**

6.1 As soon as I executed the malware sample it immediately deleted itself.

Meaning that the malware sample disappeared right after I double clicked/executed it.

<table>
<thead>
<tr>
<th>Description</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deleted File</td>
<td>C:\Documents and Settings\InfoSec Student\Desktop\40dbf4b-7db5306a.exe</td>
</tr>
</tbody>
</table>

7. **FILES AND REGISTRY KEYS CREATED, MODIFIED AND ACCESSED**

7.1 The malware sample installed/dropped the following new malicious files, and executables on the victim machine.

<table>
<thead>
<tr>
<th>File Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>New File</td>
<td>C:|WINDOWS\Temp</td>
</tr>
<tr>
<td>New File</td>
<td>C:|WINDOWS\Prefetch|KGDDE3F4B-70B53064.EXE-E34D3914ME.pf</td>
</tr>
<tr>
<td>New File</td>
<td>C:|WINDOWS\Prefetch\MGK.EXE-2ECD97E020.pf</td>
</tr>
<tr>
<td>New File</td>
<td>C:|WINDOWS\Prefetch\TASHMR.EXE-220526C305.pf</td>
</tr>
<tr>
<td>New File</td>
<td>C:\Documents and Settings|InfoSec Student|Templates\jig231f466585k</td>
</tr>
<tr>
<td>New File</td>
<td>C:\Documents and Settings\InfoSec Student|Local Settings\Temp|jg231f466585k</td>
</tr>
<tr>
<td>New File</td>
<td>C:\Documents and Settings\InfoSec Student\Local Settings\Application Data\mgc.exe</td>
</tr>
<tr>
<td>New File</td>
<td>C:\Documents and Settings\InfoSec Student\Local Settings\Application Data\jig231f466585k</td>
</tr>
<tr>
<td>New File</td>
<td>C:\Documents and Settings|InfoSec Student|Local Settings\Application Data\jig231f466585k</td>
</tr>
<tr>
<td>New File</td>
<td>C:\Documents and Settings\All User\Application Data\jig231f466585k</td>
</tr>
</tbody>
</table>
7.2 The malware sample made 54 critical changes to the registry.

<table>
<thead>
<tr>
<th>Name</th>
<th>Critical</th>
<th>Warning</th>
<th>Info</th>
<th>Ignored</th>
<th>Tested</th>
<th>Snapshot</th>
</tr>
</thead>
<tbody>
<tr>
<td>HKLM\</td>
<td>54</td>
<td>2</td>
<td>14</td>
<td>12</td>
<td>1/6/2012 3:38:23 PM</td>
<td>1/6/2012 3:31:22 PM</td>
</tr>
</tbody>
</table>

7.3 It deleted the following registry keys from the registry.

<table>
<thead>
<tr>
<th>Description</th>
<th>Severity</th>
<th>Name</th>
<th>New Value</th>
<th>Old Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deleted Key</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Key</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\Error</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Key</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\Security</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Key</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\Parameters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Key</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\Path</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Key</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\ImagePath</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Key</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\RemoteMachine</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Key</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\Start</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\ErrorControl</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\SecurityControl</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\Parameters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\Path</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\ImagePath</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\RemoteMachine</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\ErrorControl</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\SecurityControl</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\Parameters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\Path</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\ImagePath</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\RemoteMachine</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\ErrorControl</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\SecurityControl</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\Parameters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\Path</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\ImagePath</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\RemoteMachine</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deleted Value</td>
<td></td>
<td>HKLM\SYSTEM\CurrentControlSet\Services\svchost.exe\Description</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

7.4 The malware sample created the following new registry keys, Subkeys, and values.
7.5 The malware sample modified the following services on the victim machine.

It started the BITS service with two new control parameters, and deleted the automatic updates service, and registry key values. This means that the malware sample has effectively disabled Windows update, and prevented the download and installation of critical Windows updates for the victim machine. This most likely means that it is covering its tracks because it takes advantage of an existing unpatched Windows vulnerability, and updating the OS will likely kill/disable the infection/communication/propagation vector of this malware variant.

<table>
<thead>
<tr>
<th>Description</th>
<th>Name</th>
<th>New Value</th>
<th>Old Value</th>
<th>Severity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service State</td>
<td>Background Intelligent Transfer Service</td>
<td>Running</td>
<td>Stopped</td>
<td>2</td>
</tr>
<tr>
<td>Controls Accepted</td>
<td>Background Intelligent Transfer Service</td>
<td>Stop,Shutdown</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Deleted Service</td>
<td>Automatic Updates</td>
<td></td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

7.6 Running processes before, and after the malware sample was executed. Note the “tyh.exe” that is now running.
7.7 Process explorer output. Note that it is not able to verify that it is from Microsoft. And each time I execute the malware sample the name of the executable changes. Before it was tyh.exe, and now it is ucm.exe as example.

8. NETWORK BEHAVIOR (INCLUDING HOSTS, DOMAINS AND IP’S ACCESSED)

8.1 This malware sample makes a function call to the native Windows API C:\WINDOWS\System32\winsock32.dll which is the Windows Sockets API used
by most Internet and Network applications to handle network connections, denoted below in highlighted blue.

8.2 The malware sample also makes DNS requests in an attempt to resolve numerous malicious sites including mimopywyn.com, dhojocitiz.com, qobirawif.com, QOBIRAWIF.com, gavywelugamoqe.com, sesusihyt.com, and xybobimaholos.com, etc. A total of 32 different DNS requests were made but not shown for brevity.

<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source Destination</th>
<th>Protocol</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3173</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>2</td>
<td>3244</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>3</td>
<td>3245</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>4</td>
<td>3246</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>5</td>
<td>3247</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>6</td>
<td>3248</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>7</td>
<td>3249</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>8</td>
<td>3250</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>9</td>
<td>3251</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>10</td>
<td>3252</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>11</td>
<td>3253</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>12</td>
<td>3254</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>13</td>
<td>3255</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>14</td>
<td>3256</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>15</td>
<td>3257</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>16</td>
<td>3258</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>17</td>
<td>3259</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>18</td>
<td>3260</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>19</td>
<td>3261</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
<tr>
<td>20</td>
<td>3262</td>
<td>10.2.127.3</td>
<td>DNS</td>
<td>Standard query A mimopywyn.com</td>
</tr>
</tbody>
</table>

8.3 Listening network sockets before and after execution of the malware sample on the victim machine. It is clear from the below snapshot that it opened TCP:139 NetBIOS Session, Windows File and Printer Sharing port. But also with any other system running Samba (SMB). The single most dangerous port on the internet.
8.4 It did not take long before a fake A/V scanner showed me false scan results that my machine was infected with a malware infection. Clearly this Trojan wanted to steal my credit card information. The malicious software was titled “XP Internet Security 2012”. The malicious site that I was redirected to is intended to steal/collect victims credit card information, and forward the results to the following server http://bekukokymyje.com/support.html with IP of 199.168.189.25 on TCP:80. The malicious server is located in Orlando Florida U.S.A.
Malware reverse engineering part 1. Static analysis

The above GUI/Application is running under process “hwi.exe” in the directory pictured below.
8.5 The above popup redirected me to the following website. The actual form was not even a website nor an actual .html file, it was a Windows Form/GUI. The GUI did not contain any .html/JavaScript. But it made a good attempt to fool the casual user with its Internet Explorer logo.
8.6 Whois, and geolocation trace of the two malicious IP's http://bekukokymye.com/support.html that the victim made the connection to.
IP Tracing and IP Tracking (199.168.189.25)

Want to trace or track an IP Address, host, or website easily? With our highly reliable IP Address Location Database, you can get detailed information on any IP Address anywhere in the world. Results include detailed IP address location, name of ISP, netspeed/speed of internet connection, and more.

199.168.189.25 [Track IP, host or website] Examples: 213.86.63.116 (IP address) or google.com (Website)

199.168.189.25 IP address location & more:
IP address: 199.168.189.25 [Whois] [Reverse IP]
IP country code: US
IP address country: United States
IP address state: Florida
IP address city: Orlando
IP postcode: 32801
IP address latitude: 28.5445
IP address longitude: -81.3706
ISP of this IP [?] Host of this IP [?] Organization: HostDime.com
Host of this IP [?] server.bootshop.ad[Whois] [Trace]
Local time in United States: 2012-01-09 03:34

Location of the IP address 199.168.189.25
Orlando in United States. Click for this IP address location image.
9. **TIME AND LOCAL SYSTEM DEPENDANT FEATURES**

9.1 This malware sample requires a valid internet connection, and execution to activate its payload. Once executed it makes numerous DNS requests to over 32 malicious sites to download the payload/instructions in a call home fashion.
10. **METHOD AND MEANS OF COMMUNICATION**

10.1 It communications, and receives the payload/instructions from the malicious server via port TCP 80.

10.2 Server details are: [http://bekukokymyje.com/support.html](http://bekukokymyje.com/support.html) with IP of 199.168.189.25 on TCP:80. The malicious server is located in Orlando Florida U.S.A.

11. **ORIGINAL INFECTION VECTOR AND PROPAGATION METHODOLOGY**

11.1 The victim could have visited a normal looking site or may have been the victim of a browser exploit running an unpatched version of Internet Explorer. Typical drive by download is another scenario.

12. **USE OF ENCRYPTION FOR STORAGE, DELIVERY AND OR COMMUNICATION**

12.1 Nowadays advanced malware applications are either protected, obfuscated, encrypted (armorining) and/or packed (the original code is compressed, encrypted or both). This technique is applied in an attempt to evade signature based malware detection, and to hinder the efforts of static analysis by malware analysts by employing anti-reversing, anti-debugging and self-modifying code tactics. This malware sample is no different. The unpacking or decrypting of the malware layers remains the most complicated & sophisticated task in the overall process of malware analysis and finding the original entry point (OEP). True analysis of packed malicious binary code can only be performed after the payload is unpacked.

12.2 Loading the malware sample in Immunity debugger I noticed the following loaded module. C:\WINDOWS\system32\CRYPT32.dll is the module that implements many of the Certificate and Cryptographic Messaging functions in the CryptoAPI, such as [CryptSignMessage](https://msdn.microsoft.com/en-us/library/windows/desktop/aa382931(v=vs.85).aspx). Crypt32.dll is a module that comes with the Windows and Windows Server operating systems, but different versions of this DLL provide different capabilities. There is no API to determine the version of CryptoAPI that is in use, but I can determine the version of crypt32.dll that is in use via the [GetFileVersionInfo](https://msdn.microsoft.com/en-us/library/windows/desktop/aa395758(v=vs.85).aspx) and [VerQueryValue](https://msdn.microsoft.com/en-us/library/windows/desktop/aa395789(v=vs.85).aspx) functions. The function is highlighted in blue below.
13. **USE OF SELF MODIFYING/REPLICATING OR ENCRYPTED CODE**

13.1 I noticed each time I executed the malware sample that the names of the dropped malicious files ".exe"s always changed to a random string/name. Different every single time. This might indicate the use of the rand function within the code. Other than the random naming convention on the malicious executables, the network traffic seemed to be always the same. The malware sample stuck to the same 32 malicious domains in it's C&C structure.

14. **ANY INFORMATION CONCERNING DEVELOPMENT OF MALWARE (COMPILER TYPE, PACKER USED, COUNTRY OF ORIGIN, AUTHOR, NAMES/HANDLES, ETC.)**

14.1 Reverse engineering using static analysis on the malware sample allows me to understand its functionality. Loading the malware sample indicated it might be packed/compressed for several reasons. The memory visualization bar within the
IDA GUI was not able to find any encoded/executable data. Usually normal unpacked executables have several blue sections with readable data. Below is a comparison of a packed executable vs a non packed executable application.

Note the memory visualization bar within the unpacked nc.exe application, and the graph overview.

14.2 Next is a high level overview of the malware sample which involves using the start function and the “display graph of xref’s from current identifier” button. This method allows us to generate a visualization graph. The graph allows us to zoom in and inspect various portions of the program and see how much of it is actually system API calls versus custom implemented code. We can also use the graph overview to see all the function calls the application is making.
14.3 I began by dumping the basic headers and imports/export entries in the malware sample using the dumpbin program. I extracted all data from all available sections of the malware sample. Sections that are available are .data, .idata, .rdata (hardcoded passwords/sometimes), .rsrc (resource), and .text (program code) as pictured below.

```
Directory of C:\Documents and Settings\Administrator\Desktop\Win-32 Kryptik.Yda trojan variant
11/09/2012 02:39 AM  (DIR) .
11/09/2012 02:39 AM  (DIR) ..
11/02/2012 06:17 PM  291.318 40d66b1f-7db5306a
11/02/2012 06:17 PM  291.318 40d66b1f-7db5306a - Copy
11/20/2011 09:06 PM  16,940 DUMPBIN.EXE
11/09/2012 02:40 AM  291.319 hv.exe
09/20/2011 07:25 PM  471.973 LINK.EXE
09/20/2011 07:25 PM  180.276 MFC90D0B.DLL
6 Files) 1,541,293 bytes
2 Dir(s) 32,586,883,822 bytes free

C:\Documents and Settings\Administrator\Desktop\Win-32 Kryptik.Yda trojan variant>DUMPBIN.EXE 40d66b1f-7db5306a
Microsoft (R) COFF Binary File Dumper Version 6.04.8164
Copyright (C) Microsoft Corp 1992-1998. All rights reserved.

Dump of file 40d66b1f-7db5306a
File Type: EXECUTABLE IMAGE
Summary
1C000 .data
2000 .idata
22000 .rdata
6000 .rsrc
18000 .text

14.4 I ran the following commands and dumped the above sections into .txt files for further analysis.

```
>DUMPBIN.EXE /RAWDATA:BYTES /SECTION:.idata 40d66b1f-7db5306a > idatasection.txt
>DUMPBIN.EXE /RAWDATA:BYTES /SECTION:.rdata 40d66b1f-7db5306a > rdatasection.txt
>DUMPBIN.EXE /RAWDATA:BYTES /SECTION:.rsrc 40d66b1f-7db5306a > rsrcregsection.txt
>DUMPBIN.EXE /SECTION:.text /DISASM 40d66b1f-7db5306a > code.txt
```

14.5 Next I performed a full binary disassembly with all libraries included.
15. **KEY QUESTIONS AND ANSWERS**

- How did the malware infection occur?
  [drive-by infection from site Yes]
- When did the malware infection occur?
  [On or before Jan. 04, 2012]
- What vulnerabilities allowed the infection to occur?
  [Unpatched Internet Explorer/drive-by infection/banner Ad]
- What is the risk of data loss?
  [High: Kryptik/Data Stealing Trojan on machine for several days]

16. **CONCLUSIONS AND RECOMMENDATIONS TO PREVENT INCIDENT FROM RECURRING**

On Jan. 04, 2012, while browsing the internet, ANONYMOUS triggered a drive-by infection probably coming from a banner ad. The drive-by infection triggered a series of exploit steps, eventually resulting in installation of a trojan downloader and the Win-32 Kryptik.YJA trojan variant. Because Kryptic is a data-stealing trojan, any sensitive information handled by the victim between date of infection and the date of the investigation (January 09, 2012) should be considered potentially compromised.

IT Security should implement a hardened browser standard operating procedure.

This SOP should include for example, disabling JavaScript, browser hardening standards (NSA), installing no-script, and removing admin access for affected users. Also take a look at official DoD, Sans papers on browser hardening or

www.us-cert.gov/reading_room/secure_browser/

17. **FOLLOWUP ACTIONS AND LESSONS LEARNED**

17.1 Blacklist the entire offending IP block/s.

17.2 Reset user password. Re-image victim machine. If the user used ANY personal passwords to login to ANY websites (banking, social media, news feeds, educational, work websites), he should reset said passwords, and notify companies he does business. Especially if he logged on to any banking website. Users Active Directory account password should be reset, and be monitored for any unusual/unauthorized activity.
REFS used in my .pdf report.

Generic Unpacking of Self-modifying, Aggressive, Packed Binary Programs

http://arxiv.org/abs/0905.4581

Practical malware analysis


What to Include in a Malware Analysis Report


Malware Analysis 101