Johnny AppCompatCache

The Ring of Malware

PRESENTED BY: Mary Singh & Brice Daniels, Senior Consultants

JULY 9, 2013
Introductions

- Mary is an Incident Responder / Forensic Analyst
- Brice is an Incident Responder / Proactive Assessor
- APT and Financial Cases

@marycheese
@theonehiding
Mandiant: Experts in Advanced Targeted Threats

- Expert Responders for Critical Security Incidents
  - Incident responders to the biggest breaches
  - We train the FBI & Secret Service
  - Our consultants wrote the book (literally) on incident response
  - Clients include more than 33% of Fortune 500

- Our Products Are Based on Our Experience
  - Built to find and stop advanced attackers
  - We use our own products in our investigations

- Nationwide Presence
  - 350+ employees
  - Offices in DC, New York, LA, San Francisco & Dublin
Mandiant’s Unique Approach

Security incident response management platform

Products

Services

High-end, white glove incident response & security consulting services

Threat Intel

Unmatched intelligence about tools & tactics of advanced attack groups directly from the front lines
Agenda

- Types of Attackers
- Application Compatibility Cache
  - Overview
  - Registry Key
  - Structure
- ShimCacheParser
- Case Study #1 – Stacking, a system in time saves 9
  Case Study #2 – What is seen, cannot be unseen
- Cash out
- Q&A
Types of Attackers

- Application Compatibility Cache
- ShimCacheParser
- Case Studies
- Cash out
# All Threat Actors Are Not Equal

<table>
<thead>
<tr>
<th>Objective</th>
<th>Nuisance Threats</th>
<th>Economic Espionage</th>
<th>Organized Crime</th>
</tr>
</thead>
<tbody>
<tr>
<td>Launch Points &amp; Nuisance</td>
<td>Botnets &amp; Spam</td>
<td>Advanced Persistent Threat</td>
<td>Credit Card Theft</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hacktivists</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defamation, Press &amp; Policy</td>
</tr>
<tr>
<td>Anonymous &amp; Lulzsec</td>
</tr>
</tbody>
</table>

Attacks which are targeted and persistent pose the greatest challenge and the greatest risk.
Types of Attackers

Application Compatibility Cache

ShimCacheParser
Case Studies
Cash out
Application Compatibility Cache

- Overview
  - Created by Microsoft to identify application compatibility issues, helps developers troubleshoot legacy functions
    - Windows looks at AppCompatCache to determine if modules require shimming for compatibility
  - The Cache data tracks file path, size, last modified time, and last execution time (depending on OS)
  - Most recent on top, written on shutdown

- Registry key

<table>
<thead>
<tr>
<th>XP</th>
<th>HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatibility\AppCompatCache</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-XP</td>
<td>HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatCache\AppCompatCache</td>
</tr>
</tbody>
</table>
That Shim is so Cache

- AppCompatCache is the key, but it’s a.k.a. “ShimCache”
- What is a “shim” anyway?
  - Small library that intercepts an API and fixes compatibility
  - Helps legacy apps that rely on incorrect / old functionality
  - For Linux types… “Wine” is an example of a shim which enables Windows apps to run on Unix-y OSes
- Caveats…
  - Unavailable on Windows 2000 or older
  - Only files with specific extensions are logged (e.g. “.exe”, ”.bat”, ”.dll”)
  - Registry updated only on system shutdown
  - Presence in Vista+ doesn’t prove execution (more later)
- Handy protip: wood shims fix a non-level cabinet or align a door
Note: some forensic tools do not read AppCompatCache
- Why? The information is stored as “REG_BINARY” data
- Many of these tools don’t parse “big data” values
- e.g. WRR, WRA, EnCase, Registry Viewer will not display the data in AppCompatCache correctly

Some registry tools can read this data:
- Regripper with updated plugin
- Lock and Code Registry Browser
- TZWorks Yaru, more…
AppCompat Structure

- AppCompat Structure (3 formats)
  - There are differences between Windows OS versions
  - Different metadata stored
  - Three unique binary data structure formats each with 32/64bit versions stored in a series of records

- Windows XP
  - Full path of file
  - Last Modified Date
  - File size (when executed)
  - Last time the file was executed
  - 96 entries
  - Header: “0xDEADBEEF”

```c
typedef struct AppCompatibilityEntry{
    WCHAR Path[MAX_PATH+4];
    FILETIME LastModTime;
    DWORD   dwFileSize;
    DWORD   dwFileSizeHigh;
    FILETIME LastExecTime;
};
```
AppCompat Structure

- **Windows Server 2003**
  - Last Modified Date
  - Full file path
  - File size (when executed)
  - 512 entries
  - Header: “0xBADC0FFE”

- **Windows Vista+**
  - Last Modified Date
  - Full file path
  - File size
  - Binary “execution” flag
  - Logs files executed and/or created
  - 1024 entries
  - Header: “0xBADC0FFE”

```
typedef struct AppCompat_Entry32_Type1 {
USHORT wLength;
USHORT wMaximumLength;
DWORD dwPathOffset;
FILETIME qwFileTime;
DWORD dwFileSize;
DWORD dwFileSizeHigh;
};
```

```
typedef struct AppCompat_Entry32_Type2 {
USHORT wLength;
USHORT wMaximumLength;
DWORD dwPathOffset;
FILETIME qwFileTime;
DWORD dwFileFlags;
DWORD dwFlags;
DWORD dwBlobSize;
DWORD dwBlobOffset;
};
```
Types of Attackers
Application Compatibility Cache

ShimCacheParser

Case Studies
Cash out
ShimCacheParser

- ShimCacheParser.py
  - Automatically locates AppCompatCache related keys, determines their structure type and exports the data
  - 6 types of input:
    - Binary file
    - Registry Hive
    - MIR XML
    - Local System
    - .REG file
    - Mass acquires

- Download at
  https://github.com/mandiant/ShimCacheParser
# ShimCacheParser

- **Output in CSV format**

```
> ShimCacheParser.py -i D:\case\SYSTEM -o D:\case\output.txt
```

<table>
<thead>
<tr>
<th>Last Modified</th>
<th>Last Update</th>
<th>Path</th>
<th>File Size</th>
<th>Process Exec Flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>08/27/12 19:53:26</td>
<td>N/A</td>
<td>C:\Windows\system32\sql.exe</td>
<td>N/A</td>
<td>No</td>
</tr>
<tr>
<td>08/27/12 19:52:34</td>
<td>N/A</td>
<td>C:\Users\joeuser\AppData\Local\Temp\tmp83e46c15\12345.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>07/14/09 01:14:41</td>
<td>N/A</td>
<td>C:\Windows\system32\svchost.exe</td>
<td>N/A</td>
<td>No</td>
</tr>
<tr>
<td>08/24/12 19:19:59</td>
<td>N/A</td>
<td>C:\Windows\system32\b.exe</td>
<td>N/A</td>
<td>No</td>
</tr>
<tr>
<td>07/14/09 01:14:12</td>
<td>N/A</td>
<td>C:\Windows\system32\at.exe</td>
<td>N/A</td>
<td>No</td>
</tr>
<tr>
<td>08/24/12 19:37:47</td>
<td>N/A</td>
<td>C:\Windows\system32\msabc.exe</td>
<td>N/A</td>
<td>No</td>
</tr>
<tr>
<td>07/14/09 01:14:27</td>
<td>N/A</td>
<td>C:\Windows\system32\net1.exe</td>
<td>N/A</td>
<td>No</td>
</tr>
<tr>
<td>07/14/09 01:14:45</td>
<td>N/A</td>
<td>C:\Windows\system32\whoami.exe</td>
<td>N/A</td>
<td>No</td>
</tr>
<tr>
<td>07/14/09 01:14:27</td>
<td>N/A</td>
<td>C:\Windows\system32\NETSTAT.EXE</td>
<td>N/A</td>
<td>No</td>
</tr>
<tr>
<td>08/24/12 19:16:36</td>
<td>N/A</td>
<td>C:\Users\joeuser\AppData\Local\Temp\tmp591d39cc\12345.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Case Studies

Types of Attackers
Application Compatibility Cache
ShimCacheParser

Cash Out
Case Study #1 – What is EVERYONE executing?!

- Use the AppCompatCache to find evil everywhere!

- Situation
  - 30,000 node Windows AD domain
  - Active attacker

- Benefits
  - Fast results
  - Develop investigative leads
Steps

1. Collect registry keys from your network
   1. Use IOC Finder to collect AppCompatCache keys as MIR XML – except IOC Finder is a ~14MB executable
   2. Use <2KB batch script to export keys to a .reg file
      ▪ https://github.com/theonehiding/ShimCacheCollector
2. Run ShimCacheParser.py across the set
3. Analyze

4. … Profit!

Gimme Da CACHE!
Gimme Da Cache!
Exporting AppCompat Keys

- Two commands
  - `reg export [key] [file]`
  - `regedit /e [file] [key]`

```bash
rem For Windows 7
reg export "HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatCache" %TEMP%\reg_0.reg /y >nul 2>1

rem For Windows XP
regedit /e %TEMP%\reg_0.reg
  "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatibility" >nul 2>1
```
Suspicious filenames
- Pwdump.exe

Utilities possibly used by the attacker
- at.exe
- psexec.exe / psexesvc.exe

Suspicious paths
- C:\Program Files\n- C:\Recycler\n- C:\Windows\addins\n- C:\ProgramData\n
File sizes of known malware
AppCompatCache only provides file modified times
  - Except for Windows XP
  - Could be modified by the attacker
- Could correspond to the creation time
  - Tools like psexesvc.exe
Hunting an older attacker

- Data in Windows 7, Windows 2008 remains longer
- Servers may contain older records
  - AppCompat data serialized only on shutdown
Stacking helps find needles in haystacks
Use to help find files masquerading as legitimate

Complicated by temporary files
Hotfix installers with Purposely unique directories

<table>
<thead>
<tr>
<th>Count</th>
<th>File Path</th>
<th>File Size</th>
<th>Executed</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>c:\Windows\System32\scvhost.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>763</td>
<td>c:\Windows\System32\svchost.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>System</th>
<th>Date Modified</th>
<th>Date Executed</th>
<th>File Path</th>
<th>File Size</th>
<th>Executed</th>
</tr>
</thead>
<tbody>
<tr>
<td>{Win-7}</td>
<td>2011-07-09 13:23:02</td>
<td>N/A</td>
<td>c:\01882cb67ce37b6f7137\Setup.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>{Win-7}</td>
<td>2011-01-11 12:41:10</td>
<td>N/A</td>
<td>c:\01e45c03736f85750ed2\install.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>{Win-XP}</td>
<td>2010-07-05 13:15:53</td>
<td>2013-06-10 15:23:39</td>
<td>c:\01f1236656ceca4125c\update\update.exe</td>
<td>755,576</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Case Study #2 – I see what you did there

- Attacker Type: Organized Crime
- Target: Corporation
  - *Filenames changed to protect the innocent (and our NDA)

- Discovered bad file “cdel.exe” (variant of Citadel)
  - C:\Users\mary\AppData\Roaming\Gappy\cdel.exe
  - The file was timestamped
  - Extracted filename creation date from $MFT
    - August 20, 2012 18:06:49

<table>
<thead>
<tr>
<th>Name</th>
<th>Std Info</th>
<th>SIA</th>
<th>SIA</th>
<th>SIA</th>
<th>Filename</th>
<th>Filename</th>
<th>Filename</th>
<th>Filename</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdel.exe</td>
<td>06/12/12</td>
<td>06/12/12</td>
<td>06/12/12</td>
<td>10/03/12</td>
<td>08/20/12</td>
<td>08/20/12</td>
<td>08/20/12</td>
<td>08/20/12</td>
</tr>
</tbody>
</table>
Imported SYSTEM hive, exported 1,022 rows

- Since we know “cdel.exe” is bad, search for that…
  - 3 entries for cdel.exe, next to 2 entries for “shoe1.exe”
  - Note: the Last Modified date matches the Standard Info Attribute

<table>
<thead>
<tr>
<th>Last Modified</th>
<th>Last Update</th>
<th>Path</th>
<th>Size</th>
<th>Exec Flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>06/12/12 17:10:41</td>
<td>N/A</td>
<td>C:\Users\mary\AppData\Roaming\Gappy\cdel.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>10/03/12 13:12:21</td>
<td>N/A</td>
<td>C:\Users\mary\AppData\Local\Temp\tmpebc090bd\shoe1.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>02/15/12 08:12:04</td>
<td>N/A</td>
<td>C:\Users\mary\AppData\Roaming\Gappy\cdel.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>09/24/12 13:09:49</td>
<td>N/A</td>
<td>C:\Users\mary\AppData\Local\Temp\tmp6e3a4f14\shoe1.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>08/23/10 18:01:54</td>
<td>N/A</td>
<td>C:\Users\mary\AppData\Roaming\Gappy\cdel.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Look around malicious filenames, lines before and after
- Remember: the most recent entries are on top

Good ol’ Timeline analysis
- Check 8/20/12 “cdel.exe” creation date, 8/21/12 was earliest :-(
- Extracted & parsed the backup copy of the SYSTEM registry hive :-)  
- 3 entries for “c123.exe”, 2 more entries for “cdel.exe”, “shoe1.exe” 
- Note: no other evidence of “shoe1.exe” or “c123.exe”!

<table>
<thead>
<tr>
<th>Last Modified</th>
<th>Last Update</th>
<th>Path</th>
<th>Size</th>
<th>Exec Flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>08/27/12</td>
<td>N/A</td>
<td>C:\Users\mary\AppData\Local\Temp\tmp83e46c15\c123.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>08/24/12</td>
<td>N/A</td>
<td>C:\Users\mary\AppData\Local\Temp\tmp591d39cc\c123.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>08/24/12</td>
<td>N/A</td>
<td>C:\Users\mary\AppData\Local\Temp\tmpc0803709\c123.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>08/21/12</td>
<td>N/A</td>
<td>C:\Users\mary\AppData\Local\Temp\tmp4313f0ee\shoe1.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>02/25/11</td>
<td>N/A</td>
<td>C:\Users\mary\AppData\Roaming\Gapiy\cdel.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>08/20/12</td>
<td>N/A</td>
<td>C:\Users\mary\AppData\Local\Temp\1jfm\lsif.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
</tbody>
</table>
### Break the case open!

<table>
<thead>
<tr>
<th>Last Modified</th>
<th>Last Update</th>
<th>Path</th>
<th>Size</th>
<th>Exec Flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>08/24/12 19:16:36</td>
<td>N/A</td>
<td>C:\Users\mary\AppData\Local\Temp\tmp591d39cc\c123.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>07/14/09 01:14:27</td>
<td>N/A</td>
<td>C:\Windows\SysWOW64\NETSTAT.EXE</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>07/14/09 01:14:45</td>
<td>N/A</td>
<td>C:\Windows\SysWOW64\whoami.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>07/14/09 01:14:27</td>
<td>N/A</td>
<td>C:\Windows\SysWOW64\net1.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>08/24/12 19:37:47</td>
<td>N/A</td>
<td>C:\Windows\SysWOW64\msbad.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>07/14/09 01:14:42</td>
<td>N/A</td>
<td>C:\Windows\SysWOW64\taskkill.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>08/24/12 20:49:00</td>
<td>N/A</td>
<td>C:\Windows\SysWOW64\msevil.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>07/14/09 01:14:20</td>
<td>N/A</td>
<td>C:\Windows\SysWOW64\find.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>12/27/10 15:01:12</td>
<td>N/A</td>
<td>C:\Windows\SysWOW64\schtasks.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>07/14/09 01:14:12</td>
<td>N/A</td>
<td>C:\Windows\SysWOW64\at.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>07/14/09 01:14:27</td>
<td>N/A</td>
<td>C:\Windows\SysWOW64\net.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>07/14/09 01:14:21</td>
<td>N/A</td>
<td>C:\Windows\SysWOW64\HOSTNAME.EXE</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>07/14/09 01:14:21</td>
<td>N/A</td>
<td>C:\Windows\SysWOW64\ipconfig.exe</td>
<td>N/A</td>
<td>Yes</td>
</tr>
</tbody>
</table>
What now?

- Knew about “cdel.exe”
- Discovered new things:
  - 3 new filenames: “1jfmlsif.exe”, “shoe1.exe”, and “c123.exe”
  - 2 new backdoors: “msbad.exe”, “msevil.exe”
    - Still on the system: hashes, file size, file ownership
- Potentially more…
  - Compromised account “mary”
    (3 files saved in this profile)
  - Network information from the backdoors
  - Scheduled Task log files
  - Timeframe of compromise
- Search logs for activity
  - DNS logs, EVT log process monitoring, lateral movement, etc.
- Use the Indicators of Compromise (IOCs) for other systems
What have we learned?

- The AppCompatCache tracks file metadata for investigators like Last Modified date, full path, and file size
- Most recent events are on top
- New entries are written on shutdown

Takeaways:

- Source of evidence for deleted files
- Use AppCompatCache along with your timelines to reconstruct and determine attacker activity
- Plug IOCs back into an investigation to find more
Q&A

- Email
  - mary.singh@mandiant.com | brice.daniels@mandiant.com

- ShimCacheParser Whitepaper

- Additional Resources
  - Mandiant Blog: blog.mandiant.com
  - Mandiant Reports:
    - M-Trends www.mandiant.com/m-trends
    - APT1 Report: www.mandiant.com/apt1

© 2013 Mandiant Corporation. All rights reserved.