Tasty Malware Analysis with T.A.C.O.
Bringing Cuckoo Metadata into IDA Pro

Jason Jones
Me

- Security Architect for Arbor Networks
  - Security Research Analyst with Arbor ASERT for 3.5 yrs prior
- Previously spoken at
  - BlackHat / Ruxcon / AusCERT / REcon
- Research Interests
  - Automating reverse engineering
  - Graph theory / database applications for RE / security
  - Botnet monitoring
Similar Work
Similar Work

- Nothing (that I know of) uses Cuckoo as it's mechanism for propagating data into an IDB
- Inspired by similar work from many authors
- UI takes inspiration from IDAScope by Daniel Plohmann (@push_pnx)
  - Excellent plugin, in my toolbox
funcap

- https://github.com/deresz/funcap
- IDA Pro script to add some useful runtime info to static analysis.
IDA Pro Pintracer

• Maintained by Hex-Rays
• Highlights executed instructions
• Can also track registers
Joe Sandbox

- Commercial product from Joe Security
- Can produce execution graphs
- Claims to have similar plugin
- Never used personally
- Seeing that they were using API traces gave inspiration to look into doing similar with Cuckoo
- Opted to not attempt to find code so my plugin would be "clean"
Background Material
Malware Analysis Challenges

- Packers / Crypters
- Self-Modifying Code
- Process / DLL Injection
- DLL Side-loading
Cuckoo Sandbox

- Popular open-source and free sandbox
- Injects monitor DLL into malicious process, logs API calls
- Cuckoo 2.0 currently in RC stage
  - HTTPS Decryption
  - Debug stacktrace available
Memory Dumping

- Using the debug stacktrace in Cuckoo 2.0 can
  - Build a list of executed addresses
  - Use procdump to get base executable dumped
  - Attempt to retrieve memory pages containing addresses from the ramdump
  - Also use malfind from Volatility to locate other pages possibly undetected
- Using extra memory regions can then append extra sections onto base executable dump
  - Appending segment in IDA is non-ideal, IDA auto-analysis falls down in a few places
TACO Overview

- Started out as dynamically generated IDAPython scripts
  - Clunky, prevented from doing "cool" things
  - Dynamically generating "clean" IDAPython is hard
- Some features incompatible with Cuckoo 1.2 due to lack of call metadata
- Cuckoo-Modified and current Cuckoo 2.0-dev branch supported
- Idea sprung out of Joe Security's posts about execution graphs and seeing they imported analysis info into IDA
- Prior usage of tools like funcap and IDA's pintracer
TACO

• Consists of Cuckoo-based tabs for showing:
  • Processes
  • API Calls
  • Signatures
  • Imports
• Also includes other IDAPython scripts I have developed
  • Byte / Stack String viewer
  • "Interesting" XOR locator
  • Switch Jump / Case statement viewer
Loader Tab

- Main location to show a process tree and allow for specific processes to be inspected
API Calls Tab

- Reproduction of Cuckoo's Output
- Filterable / Searchable / Clickable
- Detect Called vs Logged API

Filterable by Category

Filterable by Call / Argument value

Differentiate between logged and called API
## Imports Tab

- Tries to detect dynamic imports via direct / indirect calls
- Can rename addresses of detected imports

<table>
<thead>
<tr>
<th>Address</th>
<th>DLL</th>
<th>ProcName</th>
<th>ProcAddress</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00401E64</td>
<td>WS2_32.dll</td>
<td>WSAStartup</td>
<td>0x71ab6a35</td>
<td>Dynamic</td>
</tr>
<tr>
<td>0x00401EA6</td>
<td>WS2_32.dll</td>
<td>gethostname</td>
<td>0x71ab5449</td>
<td>Dynamic</td>
</tr>
<tr>
<td>0x00401EE6</td>
<td>WS2_32.dll</td>
<td>gethostbyname</td>
<td>0x71ab5355</td>
<td>Dynamic</td>
</tr>
<tr>
<td>0x00401F16</td>
<td>WS2_32.dll</td>
<td>inet_ntoa</td>
<td>0x71ab45c1</td>
<td>Dynamic</td>
</tr>
<tr>
<td>0x00401F5F</td>
<td>WS2_32.dll</td>
<td>WSACleanup</td>
<td>0x71ab3fed</td>
<td>Dynamic</td>
</tr>
<tr>
<td>0x00402010</td>
<td>mssock.dll</td>
<td>NSPStartup</td>
<td>0x71a5bd98</td>
<td>Indirect</td>
</tr>
<tr>
<td>0x00402010</td>
<td>mssock.dll</td>
<td>NSPStartup</td>
<td>0x71a5bd98</td>
<td>Indirect</td>
</tr>
<tr>
<td>0x00402010</td>
<td>winmm.dll</td>
<td>NSPStartup</td>
<td>0x76fb1688</td>
<td>Indirect</td>
</tr>
<tr>
<td>0x004045E2</td>
<td>wininet.dll</td>
<td>InternetOpenA</td>
<td>0x3d945828</td>
<td>Dynamic</td>
</tr>
<tr>
<td>0x00404674</td>
<td>wininet.dll</td>
<td>InternetSetOptionA</td>
<td>0x3d94c39a</td>
<td>Dynamic</td>
</tr>
<tr>
<td>0x004046C3</td>
<td>wininet.dll</td>
<td>InternetConnectA</td>
<td>0x3d956f4e</td>
<td>Dynamic</td>
</tr>
<tr>
<td>0x00404712</td>
<td>wininet.dll</td>
<td>HttpOpenRequestA</td>
<td>0x3d9565a8</td>
<td>Dynamic</td>
</tr>
<tr>
<td>0x00404761</td>
<td>wininet.dll</td>
<td>HttpSendRequestA</td>
<td>0x3d947021</td>
<td>Dynamic</td>
</tr>
<tr>
<td>0x00404770</td>
<td>wininet.dll</td>
<td>HttpQueryInfoA</td>
<td>0x3d95182d</td>
<td>Dynamic</td>
</tr>
<tr>
<td>0x0040483D</td>
<td>wininet.dll</td>
<td>HttpSendRequestExA</td>
<td>0x3d95baa6</td>
<td>Dynamic</td>
</tr>
</tbody>
</table>
Signatures Tab

- Simple Display of Cuckoo Triggered Signatures
Switch Jump Viewer

- Switch jumps in malware can indicate config or cmd parsing
Stack String Locator
"Interesting" XOR Tab

<table>
<thead>
<tr>
<th>Function</th>
<th>Address</th>
<th>Loop</th>
<th>Disassembly</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 sub_401130</td>
<td>0x401349L</td>
<td>True</td>
<td>xor al, 1Ah</td>
</tr>
<tr>
<td>2 sub_401130</td>
<td>0x40132dL</td>
<td>True</td>
<td>xor al, 0CDh</td>
</tr>
<tr>
<td>3 sub_4266C4</td>
<td>0x4268c1L</td>
<td>True</td>
<td>xor al, 0CDh</td>
</tr>
<tr>
<td>4 sub_4266C4</td>
<td>0x4268ddL</td>
<td>True</td>
<td>xor al, 1Ah</td>
</tr>
</tbody>
</table>
Fin

- https://github.com/arbor-jjones/idataco