ARBOR NETWORKS

Tasty Malware Analysis with T.A.C.O.

Bringing Cuckoo Reports into IDA Pro

Ruxcon 2015

Jason Jones

Who Am I?

- Sr. Security Research Analyst for Arbor Networks' ASERT
- Attend AHA! in Austin semi-frequently
  - Welcome to the AHA track!
- Speaker at
  - BlackHatUSA / Botconf / AusCERT / REcon
- Research interests
  - RE automation
  - Malware clustering
  - Graph database applications to Reverse Engineering / Threat Intel

Agenda

- Similar Work
- Malware Behaviors
- Cuckoo Sandbox
- TACO
  - Features
  - UI
  - Demo
  - Future Work

ARBOR NETWORKS

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

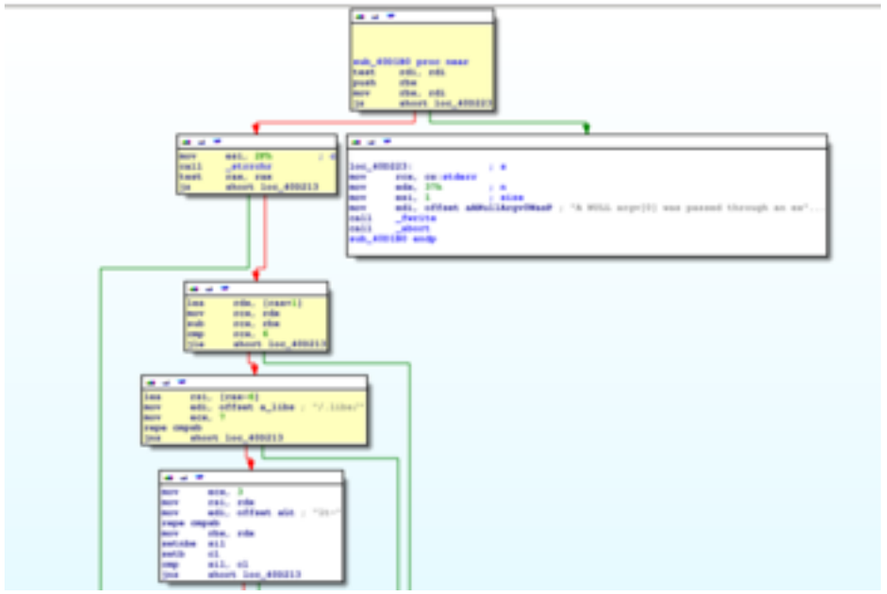
funcap

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

```
lea     eax, [ebp+NewFileName]
push    1               ; dwFlags
push    eax             ; lpNewFileName
push    esi             ; lpExistingFileName
   arg_00: 0x00404314 --> 'C:\Documents and Settings\Administrator\Local Settings\RDSessMgr'
   arg_04: 0x0012fd8c --> 'C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\~da29.tmp'
   arg_08: 0x00000001 --> 'N/A'
call    ds:MoveFileExA  ; kernel32_MoveFileExA()
   EAX: 0x00000001 --> 'N/A'
   s_arg_00: 0x00404314 --> 'C:\Documents and Settings\Administrator\Local Settings\RDSessMgr'
   s_arg_04: 0x0012fd8c --> 'C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\~da29.tmp'
   s_arg_08: 0x00000001 --> 'N/A'
push    1               ; bFailIfExists
```

IDA Pro pintracer

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



IDA View-EIP | Trace window

Thread	Address	Instruction	Result
00000000	00000000	Memory layout changed: 146 segments	Memory layout c
00000C98	.text:WinMain(x,x,x,x)+87	jz loc_10019E5	ST0=FFFFFFFF
00000C98	.text:WinMain(x,x,x,x)+8D	cmp [ebp+Msg.message], 50h	
00000C98	.text:WinMain(x,x,x,x)+91	jz loc_1004D35	PF=1
00000C98	.text:WinMain(x,x,x,x):loc_10014	mov eax, _hDlgFind	
00000C98	.text:WinMain(x,x,x,x)+9C	cmp eax, esi	EAX=00000000
00000C98	.text:WinMain(x,x,x,x)+9E	jnz loc_1004D4D	ZF=1
00000C98	.text:WinMain(x,x,x,x):loc_10014A9	lea eax, [ebp+Msg]	
00000C98	.text:WinMain(x,x,x,x)+A7	push eax ; lpMsg	EAX=000CFEDC
00000C98	.text:WinMain(x,x,x,x)+A8	push _hAccel ; hAccTable	ESP=000CFECC
00000C98	.text:WinMain(x,x,x,x)+AE	push _hwndNP ; hWnd	ESP=000CFEC8
00000C98	.text:WinMain(x,x,x,x)+B4	call ds:__imp__TranslateAcceleratorW@12; TranslateAc...	ESP=000CFEC4
00000C98	.text:WinMain(x,x,x,x)+BA	test eax, eax	EAX=00000000
00000C98	.text:WinMain(x,x,x,x)+BC	jnz short loc_1001481	
00000C98	.text:WinMain(x,x,x,x)+BE	lea eax, [ebp+Msg]	
00000C98	.text:WinMain(x,x,x,x)+C1	push eax ; lpMsg	EAX=000CFEDC
00000C98	.text:WinMain(x,x,x,x)+C2	call ds:__imp__TranslateMessage@4 ; TranslateMess...	ESP=000CFECC
00000C98	.text:WinMain(x,x,x,x)+C8	lea eax, [ebp+Msg]	EAX=00000000
00000C98	.text:WinMain(x,x,x,x)+CB	push eax ; lpMsg	EAX=000CFEDC
00000C98	.text:WinMain(x,x,x,x)+CC	call ds:__imp__DispatchMessageW@4 ; DispatchMess...	ESP=000CFECC
00000C98	.text:WinMain(x,x,x,x)+D2	jmp short loc_1001481	EAX=00000000
00000C98	.text:WinMain(x,x,x,x):loc_1001481	push esi ; wMsgFilterMax	
00000C98	.text:WinMain(x,x,x,x)+7D	push esi ; wMsgFilterMin	ESP=000CFECC
00000C98	.text:WinMain(x,x,x,x)+7E	push esi ; hWnd	ESP=000CFEC8
00000C98	.text:WinMain(x,x,x,x)+7F	lea eax, [ebp+Msg]	ESP=000CFEC4
00000C98	.text:WinMain(x,x,x,x)+82	push eax ; lpMsg	EAX=000CFEDC
00000C98	.text:WinMain(x,x,x,x)+83	call edi ; GetMessageW(x,x,x,x) ; GetMessageW(x,...	ESP=000CFEC0
00000C98	.text:NPWndProc(x,x,x,x)	mov edi, edi	EAX=010014DE

Line 1 of 39277

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"

ARBOR NETWORKS

Malware Analysis Challenges

Packers / Crypters

- Compress or encrypt code, designed to make malware less detectable
- UPX most popular packer (also watch out for things that look like, but are not UPX)
- Lots of packers with various trial licenses
- TitaniumCore by ReversingLabs can help automate
- No known (to me) auto un-crypters
- PIN, Dynamo Rio have tools to facilitate
- IDA Pro as a "universal unpacker" that has been useful at times

Self Modifying Code

- Exhibited by numerous malware families
  - Shylock
  - Andromeda / Gamarue
- Modify code that already exists instead of allocating new memory to unpack
  - Usually will be stomped during execution
  - More problematic to do automated dumps

Process / DLL Injection

- Can be done via
  - CreateRemoteThread (Suspended)
  - QueueUserAPC
  - Process Hollowing
- Cuckoo uses injection to get monitor DLL into malicious processes

DLL Side Loading

- Popular technique with targeted malware
  - PlugX
  - HTTP Browser RAT
- Load malicious DLL into legit (signed) executable
  - Bypass (some) AV
  - Bypass requirements of running code in signed exe

ARBOR NETWORKS

Cuckoo Sandbox

Cuckoo Sandbox

- Likely most popular open-source / free sandbox available
- 2.0 Supports Android (via emulator), Linux, and x64 analysis
  - Switch to new monitor code
- Third-party kernel introspection support - "zer0m0n"
- Popular fork "cuckoo-modified" by @spender of Optiv, Inc. (Accuvant)
- https://github.com/brad-accuvant/cuckoo-modified
- Contains bugfixes + additions to old cuckoomon not available in -trunk
- Cuckoo 2.0 solves many of the issues we relied on -modified fork for and adds new things

Cuckoo Sandbox

- Multiple analysis methods
- Cuckoo Monitor DLL injected into spawned process
  - Injects into any other spawned / injected processes
  - Hooks many common API calls
  - Nothing is immune to un-hooking, including Monitor
- Logs
  - Win32 API calls
  - Registry
  - Created / Modified Files
- Postprocessing Signatures

Cuckoo Behavior Report

cuckoo

Compare this analysis to...

Quick Overview | Static Analysis | Behavioral Analysis | Network Analysis | Dropped Files (2) | Reports | Comments | Statistics | Admin

Process Tree

- 45446231addbaf0ea542b64c462d7c2c.exe 456
  - vercIsid.exe 3832
    - dwwin.exe 1684
    - drwtsn32.exe 1908

Search | 45446231addbaf0ea542b64c462d7c2c.exe | vercIsid.exe | dwwin.exe | drwtsn32.exe

45446231addbaf0ea542b64c462d7c2c.exe, PID: 456, Parent PID: 520

default | registry | filesystem | network | process | threading | services | device | synchronization | crypto | browser | all

Comma-separated API filter

1

Time	TID	Caller	API	Arguments	Status	Return	Repeated
2015-10-08 12:15:02,186	2764	0x004023bf 0x004010e4	GetSystemTimeAsFileTime		success	0x00000000	

ARBOR NETWORKS

Cuckoo Behavior - Calls

Caller / Parent Caller Addresses

2015-10-08 12:15:07,015	2764	0x02135130 0x02135936	NtProtectV	ProcessHandle: 0xffffffff BaseAddress: 0x71ab1000	success	0x00000000
2015-10-08 12:15:07,015	2764	0x02135130 0x02135936	**NtProtectVirtualMemory**	OldAccessProtection: PAGE_READWRITE NumberOfBytesProtected: 0x00001000 NewAccessProtection: PAGE_EXECUTE_READ ProcessHandle: 0xffffffff BaseAddress: 0x71ab1000	success	0x00000000
2015-10-08 12:15:07,015	2764	0x02135130 0x02135936	**NtOpenSection**	DesiredAccess: 0x0000000e ObjectAttributes: WS2HELP.dll SectionHandle: 0x00000000	failed	OBJECT_NAME_NOT_FOUND
2015-10-08 12:15:07,015	2764	0x02135130 0x02135936	**NtQueryAttributesFile**	FileName: C:\Documents and Settings\Administrator\Local Settings\Temp\WS2HELP.dll	failed	OBJECT_NAME_NOT_FOUND
2015-10-08 12:15:07,015	2764	0x02135130 0x02135936	**NtQueryAttributesFile**	FileName: C:\WINDOWS\system32\ws2help.dll	success	0x00000000
2015-10-08 12:15:07,015	2764	0x02135130 0x02135936	**NtOpenFile**	ShareAccess: FILE_SHARE_READ|FILE_SHARE_DELETE FileName: C:\WINDOWS\system32\ws2help.dll DesiredAccess: FILE_EXECUTE|SYNCHRONIZE FileHandle: 0x00000094	success	0x00000000
2015-10-08 12:15:07,015	2764	0x02135130 0x02135936	**NtCreateSection**	ObjectAttributes: DesiredAccess: SECTION_QUERY|SECTION_MAP_READ|SECTION_MAP_WRITE|SECTION_MAP_EXECUTE SectionHandle: 0x00000098	success	0x00000000

Cuckoo Behavior JSON -Modified


```
    {
        "category": "network",
        "parentcaller": "0x02d9bd76",
        "return": "0x00cc0008",
        "timestamp": "2015-09-29 00:18:04,734",
        "caller": "0x02d9f5fd",
        "thread_id": "1532",
        "repeated": 0,
        "api": "InternetConnectA",
        "status": true,
        "arguments": [
            {
                "name": "Username",
                "value": ""
            },
            {
                "name": "Service",
                "value": "3"
            },
            {
                "name": "InternetHandle",
                "value": "0x00cc0004"
            },
            {
                "name": "ServerName",
                "value": "macsystem.jp.net"
            }
        ]
    }
```

Cuckoo Behavior JSON -2.0


```
},
{
    "category": "network",
    "status": 1,
    "stacktrace": [
        "InternetSetOptionW+0x68 InternetCreateUrlA-0x68a6 wininet+0xbca9 @ 0x771bbca9",
        "8982410d05e1839148299ca96af9e4c8+0xb7f2 @ 0x100b7f2"
    ],
    "api": "InternetSetOptionA",
    "return_value": 1,
    "arguments": {
        "option": 31,
        "internet_handle": "0x00cc000c"
    },
    "time": 1444464500.5,
    "tid": 1420,
    "flags": {
        "option": "INTERNET_OPTION_SECURITY_FLAGS"
    }
},
{
    "category": "system",
    "status": 1,
    "stacktrace": [
        "GetProcAddress+0x3e IsProcessorFeaturePresent-0x4c kernel32+0xae6e @ 0x7c80ae6e",
        "8982410d05e1839148299ca96af9e4c8+0x100b7 @ 0x10100b7"
    ],
    "api": "LdrGetProcedureAddress",
    "return_value": 0,
    "arguments": {
        "ordinal": 0,
        "module_address": "0x771b0000",
        "function_address": "0x77212ebc",
        "function_name": "HttpSendRequestW"
    },
    "time": 1444464500.5,
    "tid": 1420,
    "flags": {}
},
```


ARBOR NETWORKS

ASERT's Sandbox Usage

- Treat Cuckoo (and other sandboxes) as a black-box
  - Malware in, report / memory dumps / files out
  - Tasks deleted upon completion
- Centralized malware processing system
  - Normalize + insert results
  - Post-processing of memory, network traffic, behavior
  - Custom post-processing of specific families to extract various sample properties

Cuckoo API Additions needed

- Cuckoo can produce a process dump
  - This is not loadable by IDA Pro (AFAIK)
  - Can be extremely large, especially in case of {explorer,svchost,iexplore,etc.}.exe
- Can also produce full RAM dump
- Volatility has plugins to dump processes, DLLs, VADs
  - Dumping process as a PE not supported natively by Cuckoo
  - Due to time needed to use volatility, decided that was not the right place
  - Don't always want dumps, sometimes we need to do "extra"
- Added new API call to allow for arbitrary volatility plugins to run "on-demand"

API Additions needed (cont)

- Run volatility against ramdump to get process dumps for all PIDs known
- Injection detected = run malfind and dump pages
  - Stitch dumped memory pages into process dumps for "complete" view
- Supports family specific behavior
  - DLL dump
  - Specific process / memdumps

Dumping Memory

- That said... malfind doesn't always find everything
  - Will not dump DLL injected with CreateRemoteThread by design
  - Permissions stomp = undetected
  - Walk the Cuckoo API Calls per process
    - Get list of memory ranges that contain executed code
    - Run vadwalk for the PID
    - Parse the output and find all the required VAD's to cover what got executed
    - Request those VADs and then order with malfind VAD's and stitch an executable together
- Using that dump, can now follow execution much better

Creating the Memory Dump

- Attempted to add as sections using http://git.n0p.cc/?p=SectionDoubleP.git
  - Works great for any case where section is above ImageBase
  - BUT many malwares like to inject below the ImageBase
    - Modify ImageBase
    - Modify each existing section's VirtualAddress
    - Modify AddressOfEntryPoint
    - Add Sections...
    - Fail.
    - Fallback to using IDA Pro segment create / put_many_bytes
    - Non-ideal, but IDA plugin requires IDA Pro...
- Non-trivial method of creating dumps, but worth it

Memory Dump Process Output

- python create_voldump.py --task 294832 --pid 3816
- [+] Base memory range: 01000000 -> 01005600
- [+] Interesting page: 0x000C0000
- [+] Interesting page: 0x00B40000
- [+] Interesting page: 0x00B50000
- [+] Interesting page: 0x00B60000
- [+] Interesting page 0x000C0000 is in VAD 0x000C0000 - 0x000DCFFF
- [+] Interesting page 0x00B40000 is in VAD 0x00B40000 - 0x00B70FFF
- [+] Interesting page 0x00B50000 is in VAD 0x00B40000 - 0x00B70FFF
- [+] Interesting page 0x00B60000 is in VAD 0x00B40000 - 0x00B70FFF
- [+] Retrieving VAD 0x000C0000
- [+] Retrieving VAD 0x00B40000
- [+] Generating IDB with new memory regions
- [+] IDB available at explorer.exe-3816.idb

ARBOR®
NETWORKS

TACO

Overview

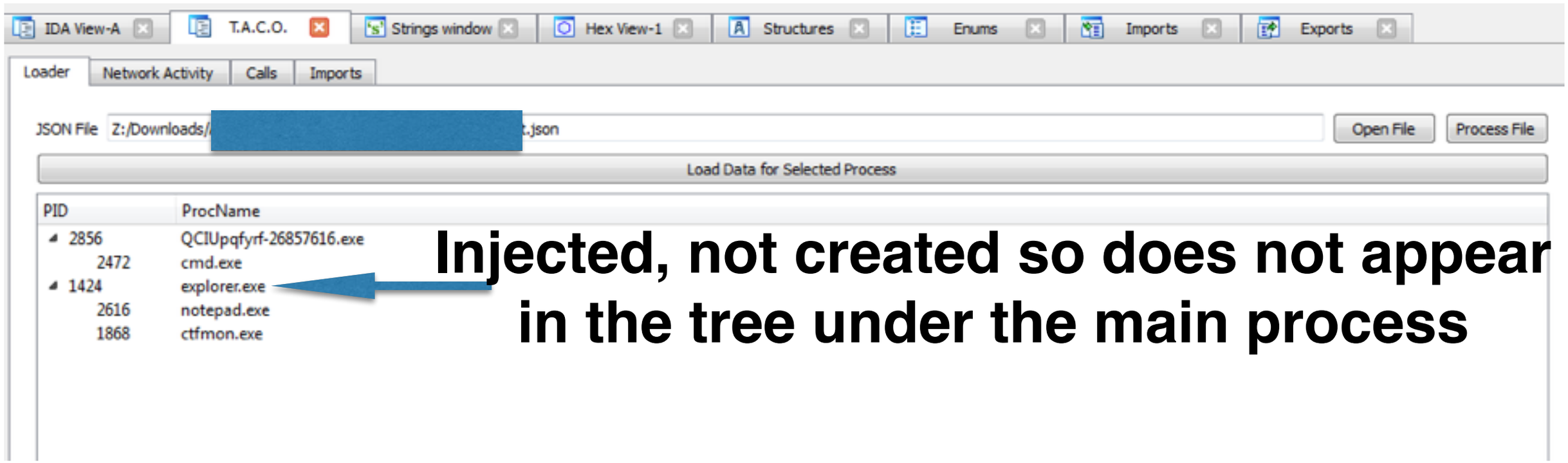
- Started out as dynamically generated Python scripts
  - Clunky, prevented from doing "cool" things
  - Dynamically generating "clean" IDAPython is hard
  - Some features incompatible with Cuckoo 1.2 due to lack of call metadata
  - Cuckoo-Modified and current Cuckoo 2.0-dev branch supported supported for markup
    - Cuckoo 2.0-dev is still a WIP as some oddities are encountered
- 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 Overview

- What does TACO stand for?
  - It's fluid..
  - Considered naming TACOZ - Tasty Analysis using Cuckoo Output and Zoidberg
    - Because why not Zoidberg?
- 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

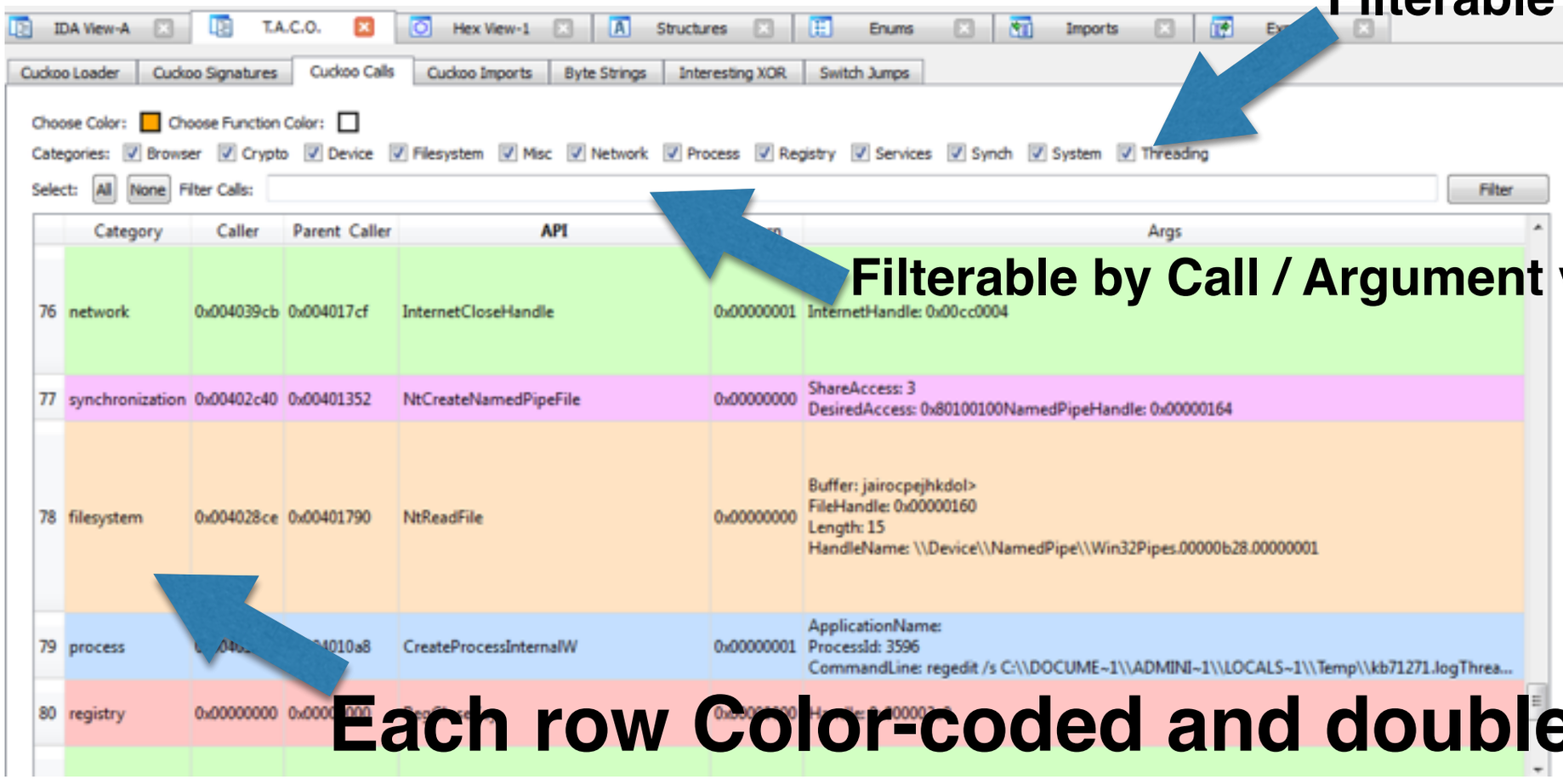
Loader Tab

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



API Call Tab

- Reproduction of Cuckoo's Output
- Filterable / Searchable / Clickable



API Call Tab (cont.)

- Add / Remove Markup to IDB
  - All
  - Category
- Context menu
  - Markup per Instruction
  - Copy value

Imports Tab

- Tries to detect dynamic imports via direct / indirect calls

Cuckoo Loader | Cuckoo Network Activity | Cuckoo Calls | **Cuckoo Imports** | Byte Strings | Interesting XOR | Switch Jumps

	Address	DLL	ProcName	ProcAddress	Type
1	0x00401E64	WS2_32.dll	WSAStartup	0x71ab6a55	Dynamic
2	0x00401EA6	WS2_32.dll	gethostname	0x71ab5449	Dynamic
3	0x00401EE6	WS2_32.dll	gethostbyname	0x71ab5355	Dynamic
4	0x00401F16	WS2_32.dll	inet_ntoa	0x71ab45c1	Dynamic
5	0x00401F5F	WS2_32.dll	WSACleanup	0x71ab3fed	Dynamic
6	0x00402010	mswsock.dll	NSPStartup	0x71a5bd98	Indirect
7	0x00402010	mswsock.dll	NSPStartup	0x71a5bd98	Indirect
8	0x00402010	winrnr.dll	NSPStartup	0x76fb1688	Indirect
9	0x004045E2	wininet.dll	InternetOpenA	0x3d945828	Dynamic
10	0x00404674	wininet.dll	InternetSetOptionA	0x3d94c39a	Dynamic
11	0x004046C3	wininet.dll	InternetConnectA	0x3d956f4e	Dynamic
12	0x00404712	wininet.dll	HttpOpenRequestA	0x3d9565a8	Dynamic
13	0x00404761	wininet.dll	HttpSendRequestA	0x3d947021	Dynamic
14	0x004047AA	wininet.dll	HttpQueryInfoA	0x3d95182d	Dynamic
15	0x0040483D	wininet.dll	HttpSendRequestExA	0x3d9baba6	Dynamic

```
.text:004047B8   mov   [ebp+var_B4], 'H'
.text:004047C2   mov   [ebp+var_B3], 't'
.text:004047C9   mov   [ebp+var_B2], 't'
.text:004047D0   mov   [ebp+var_B1], 'p'
.text:004047D7   mov   [ebp+var_B0], 'S'
.text:004047DE   mov   [ebp+var_AF], bl
.text:004047E4   mov   [ebp+var_AE], 'n'
.text:004047EB   mov   [ebp+var_AD], 'd'
.text:004047F2   mov   [ebp+var_AC], 'R'
.text:004047F9   mov   [ebp+var_AB], bl
.text:004047FF   mov   [ebp+var_AA], 'q'
.text:00404806   mov   [ebp+var_A9], 'u'
.text:0040480D   mov   [ebp+var_A8], bl
.text:00404813   mov   [ebp+var_A7], 's'
.text:0040481A   mov   [ebp+var_A6], 't'
.text:00404821   mov   [ebp+var_A5], 'E'
.text:00404828   mov   [ebp+var_A4], 'x'
.text:0040482F   mov   [ebp+var_A3], 'A'
.text:00404836   mov   [ebp+var_A2], 0
.text:0040483D   call  ds:GetProcAddress
```

Cuckoo Signatures Tab

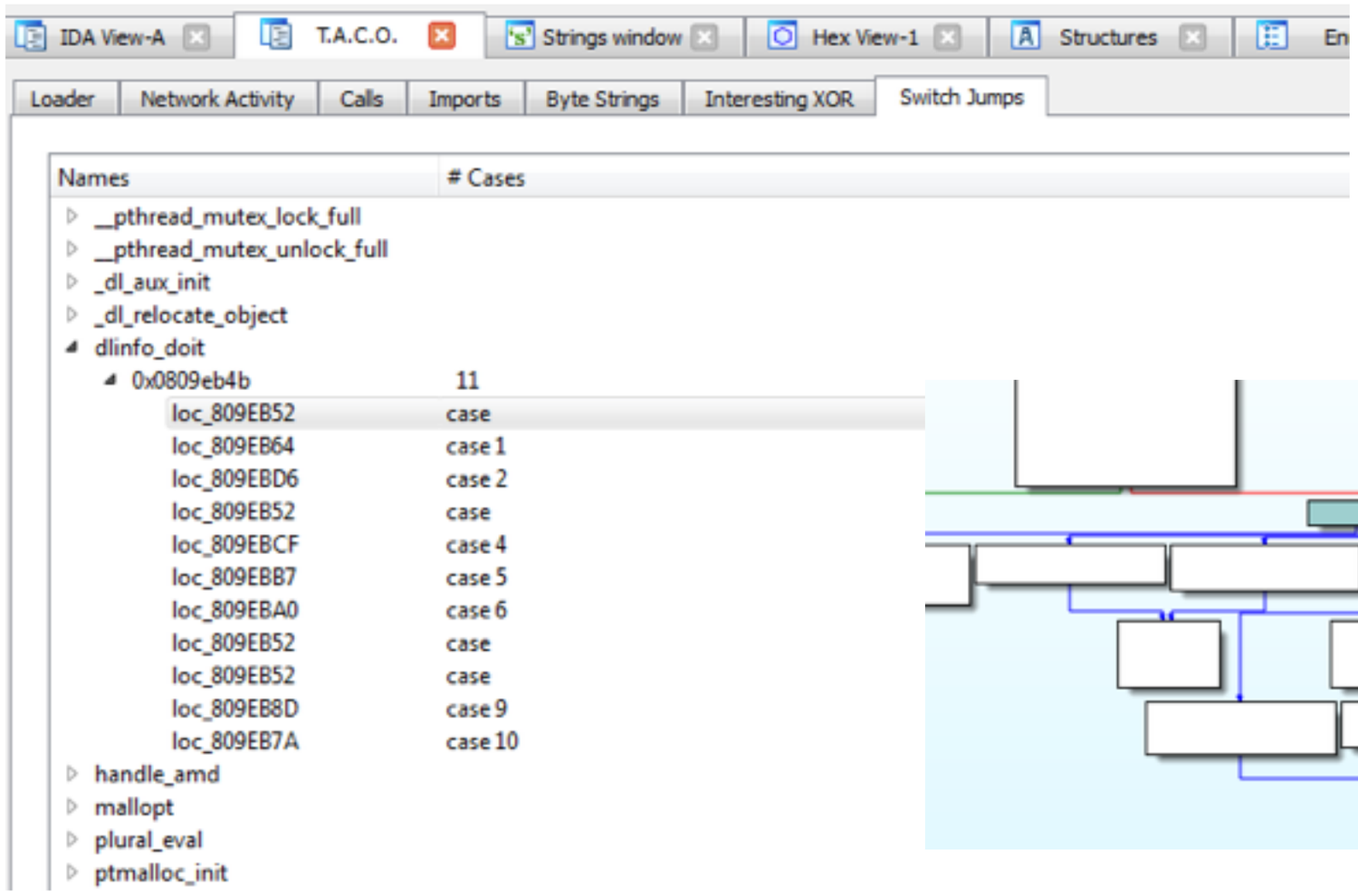
- Simple Display of Cuckoo Triggered Signatures

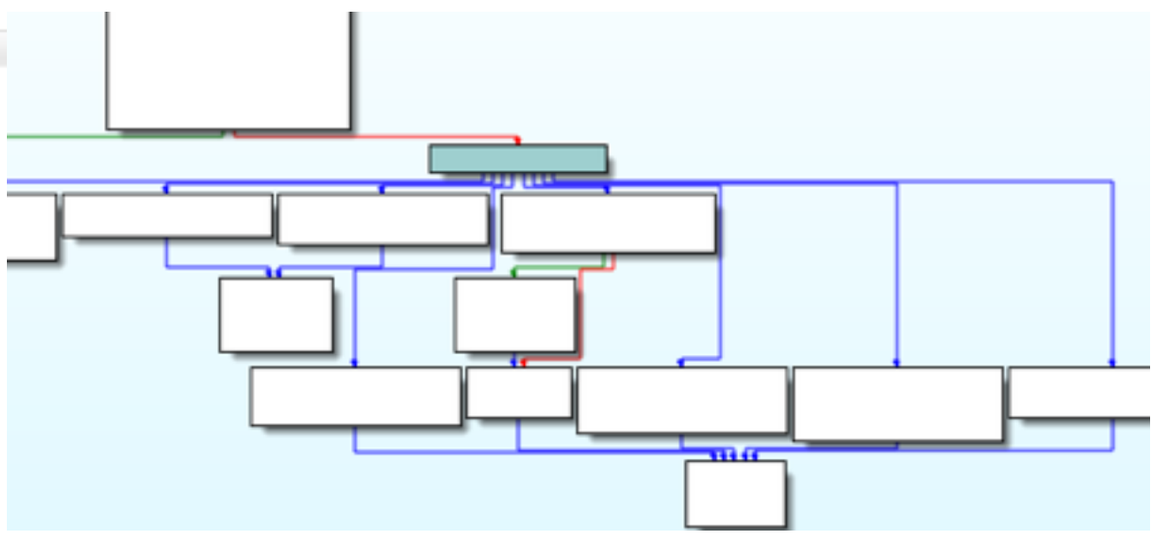
Cuckoo Loader | Cuckoo Signatures | Cuckoo Calls | Cuckoo Imports | Byte Strings | Interesting XOR | Switch Jumps

	Signature	Information	Severity
1	Reads data out of its own binary image	self_read: process: Gadget.exe, pid: 2388, offset: 0x00000000, length: 0x00006600	Severity: 2 Confidence: 30 Weight: 1
2	Creates a hidden or system file	file: C:\Documents and Settings\All Users\AVck\Gadget.exe	Severity: 3 Confidence: 100 Weight: 1
3	Deletes its original binary from disk		Severity: 3 Confidence: 100 Weight: 1
4	The binary likely contains encrypted or compressed data.	section: name: .data, entropy: 7.22, characteristics: IMAGE_SCN_CNT_INITIALIZED_DATA|IMAGE_SCN_MEM_READ| IMAGE_SCN_MEM_WRITE, raw_size: 0x00030000, virtual_size: 0x00032abc	Severity: 2 Confidence: 100 Weight: 1
5	Drops a binary and executes it	binary: C:\Documents and Settings\All Users\AVck\Gadget.exe	Severity: 2 Confidence: 50 Weight: 1
6	Creates RWX memory		Severity: 2 Confidence: 50 Weight: 1
7	Generates some ICMP traffic		Severity: 3 Confidence: 100 Weight: 1

Switch Viewer

- Switch jumps in malware can indicate config or cmd parsing





Byte String / Stack String Finder

IDA View-A | T.A.C.O. | Hex View-1 | Structures | Enums

Cuckoo Loader | Cuckoo Signatures | Cuckoo Calls | Cuckoo Imports | **Byte Strings** | Interesting XOR | Switch Jumps

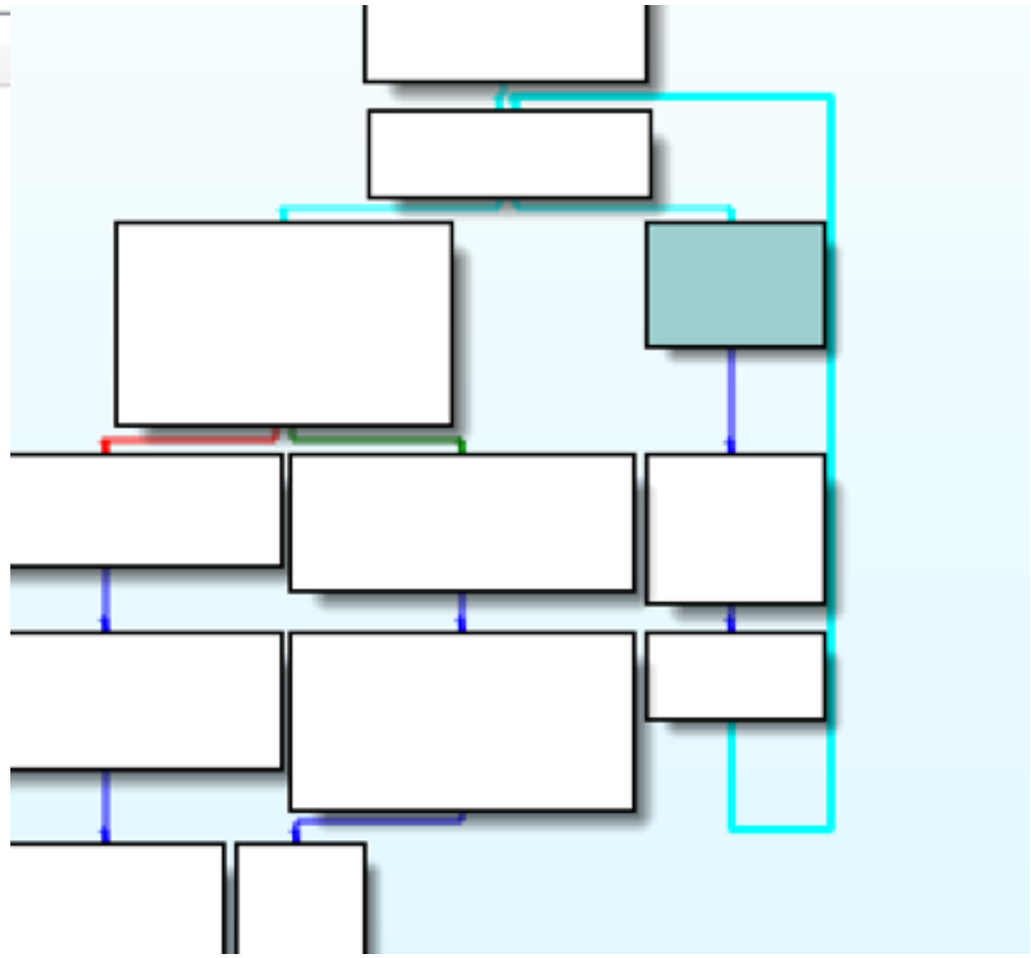
	Address	Function	String
31	0x404C46	sub_404C30	[HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run] "%s"="%s"
32	0x4025DC	sub_402460	[%s] Excuted Success [#%d]
33	0x4045F2	sub_404510	InternetSetOptionA
34	0x40360C	sub_4034C0	/AWS%d.jsp?%s
35	0x4010C2	_WinMain@16	[LM1213]
36	0x405010	sub_404C30	regedit /s %s
37	0x401B98	_WinMain@16	'%s' isnt command,press ? for help
38	0x404EEA	sub_404C30	kb71271.log
39	0x401EB0	sub_401D80	gethostbyname
40	0x402A82	sub_402950	ERROOTHER ERROR

```
:004045F2                       mov     [ebp+var_C8], 'I'
:004045F9                       mov     [ebp+var_C7], 'n'
:00404600                       mov     [ebp+var_C6], 't'
:00404607                       mov     [ebp+var_C5], bl
:0040460D                       mov     [ebp+var_C4], 'r'
:00404614                       mov     [ebp+var_C3], 'n'
:0040461B                       mov     [ebp+var_C2], bl
:00404621                       mov     [ebp+var_C1], 't'
:00404628                       mov     [ebp+var_C0], 'S'
:0040462F                       mov     [ebp+var_BF], bl
:00404635                       mov     [ebp+var_BE], 't'
:0040463C                       mov     [ebp+var_BD], 'O'
:00404643                       mov     [ebp+var_BC], 'p'
:0040464A                       mov     [ebp+var_BB], 't'
:00404651                       mov     [ebp+var_BA], 'i'
:00404658                       mov     [ebp+var_B9], 'o'
:0040465F                       mov     [ebp+var_B8], 'n'
:00404666                       mov     [ebp+var_B7], 'A'
:0040466D                       mov     [ebp+var_B6], 0
:00404674                       call    ds:GetProcAddress
:00404674 api: LdrGetProcedureAddress
:00404674 Ordinal: 0
:00404674 ModuleName: wininet.dll
:00404674 FunctionName: InternetSetOptionA
:00404674 ModuleHandle: 0x3d930000
:00404674 FunctionAddress: 0x3d94c39a
```

XOR Locator

Loader | Network Activity | Calls | Imports | Byte Strings | Interesting XOR

	Function	Address	Loop	Disassembly
1	sub_401130	0x401349L	True	xor al, 1Ah
2	sub_401130	0x40132dL	True	xor al, 0CDh
3	sub_4266C4	0x4268c1L	True	xor al, 0CDh
4	sub_4266C4	0x4268ddL	True	xor al, 1Ah



DEMO

- TACO Time!
  - Shifu (banker)
  - Andromeda (loader / stealer)
  - PlugX (targeted)
  - Etumbot (targeted)
  - Fobber (banker, Cuckoo 2.0-dev)
  - HttpBrowserRAT (targeted, Cuckoo 1.2)

ARBOR NETWORKS

Wrap-Up

Wrap-Up

- Hopefully you agree that a TACO is both a tasty treat and is a useful tool to bring run-time info into IDA Pro
- All code is / will be freely available on GitHub
  - https://github.com/arbor-jjones/idataco
  - https://github.com/arbor-jjones/malware/create_voldump.py
  - https://github.com/arbor-jjones/malware/ida_load_mem.py
  - https://gist.github.com/arbor-jjones/18dd572e6b3e391e8418

Future Work

- Add path-finding capabilities
- Direct comments to API call arguments with values
- Clean up filter code to allow for arg- or API call-specific filtering
- Rename vars / dwords used to store GetProcAddress result
- Rename unknown calls
- Determine way to achieve 'persistence' for names / ops (allow more 'undo')
  - SQLite?
  - Marks?
- Batch mode to markup / rename things in IDB
- Support other sandboxes where possible

Questions/Comments/Feedback

ARBOR NETWORKS

ARBOR SERT
Security Engineering & Response Team

Thank You!