

An abstract graphic on the left side of the slide, consisting of a network of light blue lines and circles of varying sizes, resembling a circuit board or a data network. The lines are vertical and horizontal, with some diagonal connections, and the circles are placed at various points along these lines.

FARMING THE LOOT CAVE

BIG DATA HUNTING IN MEMORY WITH SPLUNK AND TA-VOLATILITY

ANDREW QUILL / CHRISTOPHER BURCH

WHO ARE WE?

- Andrew Quill

- Started 26 years ago with Windows 3.1 and a DOS manual
- Cybersecurity and Application engineering, Houston 2003
- US ARMY Signal Support Systems Specialist and Cyber Network Operator 2007 - 2015
- Independent Researcher and Consultant 2015 - Present
- Associates Degree in Applied Sciences: Application Dev for Windows 2005
- Current Certifications: GSEC, GCED, GCIH, GCIA, GCFE, GCUX, GXPN
- Obsessive dabbler in just about all areas of computer science these days...



WHO ARE WE?

- Christopher Burch

- Began by borking Windows 3.1 with a fudged autoexec.bat 24 years
 - US Air Force Workgroup Administrator 2005 - 2009
 - Systems Administrator / Unix Gray Beard 2009 - 2016
 - Independent Researcher and Consultant 2016 - Present
-
- BS in Software Engineering w/ spec in Network Engineering 2012
 - Current Certifications: GSEC, GPEN, GCUX, CISSP
-
- Can teach you how make a rack on a budget...hint: IKEA



TRADITIONAL MEMORY HUNTING

Performed via:

- Command Line Analysis
- Specialized Commercial Applications (\$\$)
- Non-Repeatable Custom Scripts



FOLKS, THIS IS RECENT!

```
user@ubuntu:~$ vol -f memory_images/example.vmem --profile=WinXPSP2x86 pslist
```

```
Volatile Systems Volatility Framework 2.3_alpha
```

Offset(V)	Name	PID	PPID	Thds	Hnds	Sess	Wow64	Start	Exit
0x819cc830	System	4	0	60	209	-----	0		
0x818efda0	smss.exe	384	4	3	19	-----	0	2011-09-26 01:33:32	
0x81616ab8	csrss.exe	612	384	12	473	0	0	2011-09-26 01:33:35	
0x814c9b40	winlogon.exe	636	384	16	498	0	0	2011-09-26 01:33:35	
0x81794d08	services.exe	680	636	15	271	0	0	2011-09-26 01:33:35	
0x814a2cd0	lsass.exe	692	636	24	356	0	0	2011-09-26 01:33:35	
0x815c2630	vmacthlp.exe	852	680	1	25	0	0	2011-09-26 01:33:35	
0x81470020	svchost.exe	868	680	17	199	0	0	2011-09-26 01:33:35	
0x818b5248	svchost.exe	944	680	11	274	0	0	2011-09-26 01:33:36	
0x813a0458	MsMpEng.exe	1040	680	16	322	0	0	2011-09-26 01:33:36	
0x816b7020	svchost.exe	1076	680	87	1477	0	0	2011-09-26 01:33:36	
0x817f7548	svchost.exe	1200	680	6	81	0	0	2011-09-26 01:33:37	
0x8169a1d0	svchost.exe	1336	680	14	172	0	0	2011-09-26 01:33:37	
0x813685e0	spoolsv.exe	1516	680	14	159	0	0	2011-09-26 01:33:39	
0x818f5cd0	explorer.exe	1752	1696	32	680	0	0	2011-09-26 01:33:45	
0x815c9638	svchost.exe	1812	680	4	102	0	0	2011-09-26 01:33:46	
0x8192d7f0	VMwareTray.exe	1876	1752	3	84	0	0	2011-09-26 01:33:46	
0x818f6458	VMwareUser.exe	1888	1752	9	245	0	0	2011-09-26 01:33:47	
0x8164a020	msseces.exe	1900	1752	11	205	0	0	2011-09-26 01:33:47	
0x81717370	ctfmon.exe	1912	1752	3	93	0	0	2011-09-26 01:33:47	

WHAT CHOICE DO WE HAVE?

- Volatility Unified Output Formats
 - JSON
 - DOT
 - HTML
 - XLSX
 - SQLITE
- FireEye's Mandiant Redline
- PassMark Software's Volatility Workbench



THESE ARE OK, BUT WE WANT BOTH



Good Analysis



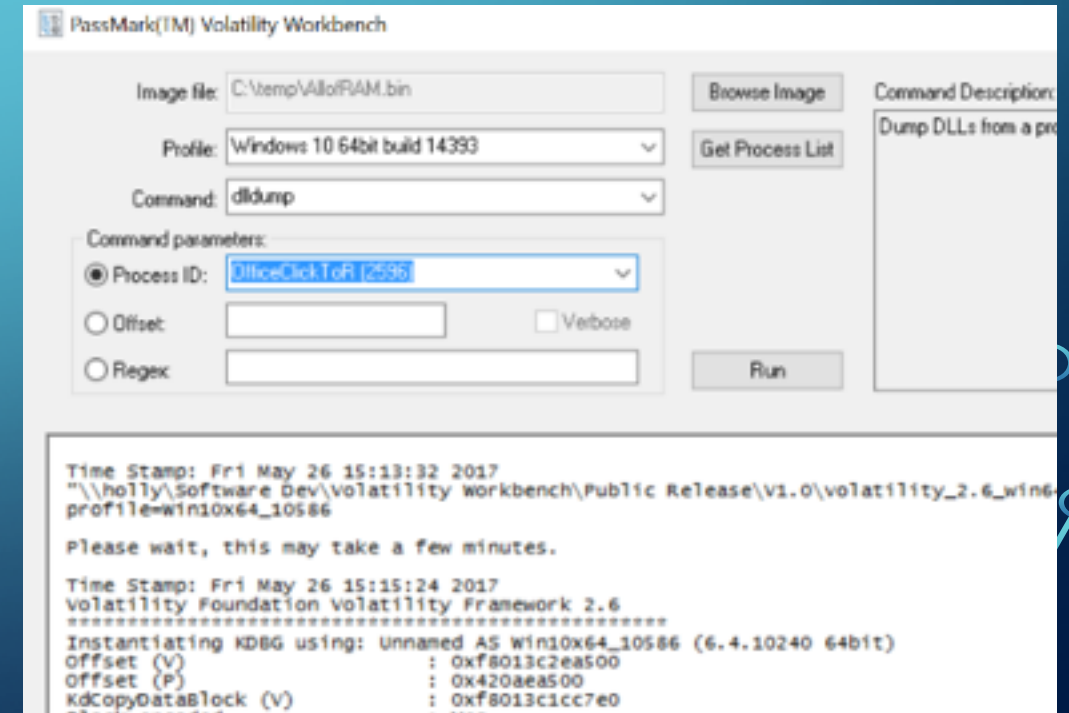
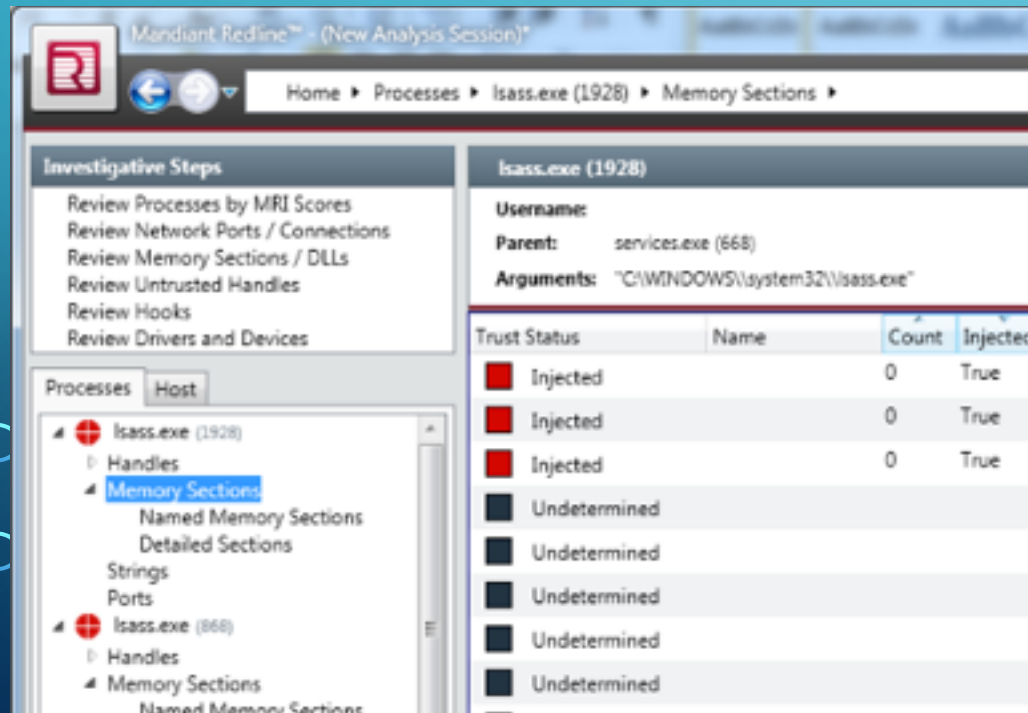
Individually Processed



Automated Commands



No Analysis



THEY CALL IT

"BIG DATA"

imgflip.com

BUT MEMORY ANALYSIS ISN'T BIG DATA!



Splunk is just a “Big Data” Platform



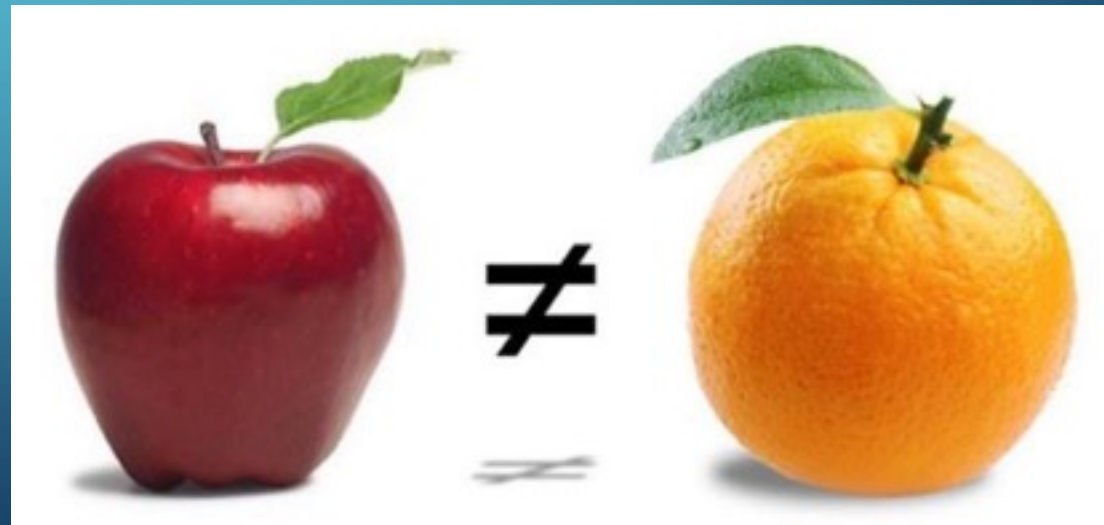
Splunk ingests machine data and applies analytics and visualizations



Splunk can do a lot of that scripting crap for me!

A PLATFORM ALONE ISN'T ENOUGH

- Splunk is good at ingesting JSON data
- Volatility is good at producing JSON data
- Volatility JSON \neq Splunk JSON
- We needed a translator



TA-VOLATILITY

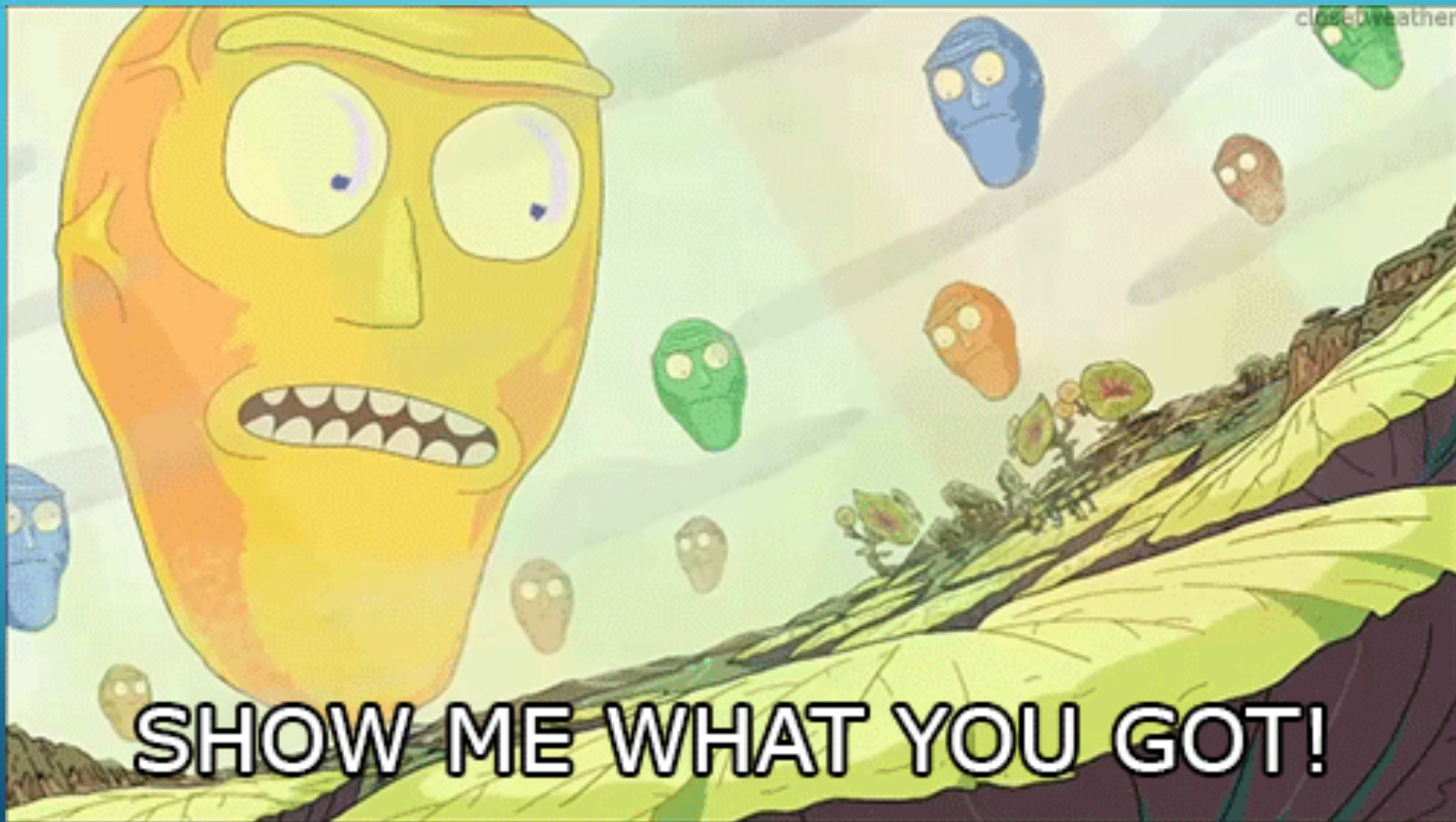
- JSON Translator
- Written in Python2.7
- Utilizes Splunk Multithreading
- Extensible
- Flexible
- Updatable w/o Restarting ANY services
- Look for new release version 2.6.9 - Today!
- <https://splunkbase.splunk.com/app/3919/>



WHAT IT DOESN'T DO

- Run Volatility commands
 - Future goal
- Teach you Splunk Processing Language (SPL)
 - <https://answers.splunk.com/>
 - Splunk Fundamentals I and II
- Perform the analysis for you





MAKE A REFERENCE TABLE FOR LATER USE

```
index::main source="john_doe_pslist.json"  
| table pid,ppid,name,image  
| outputlookup volpids.csv
```



volpids.csv

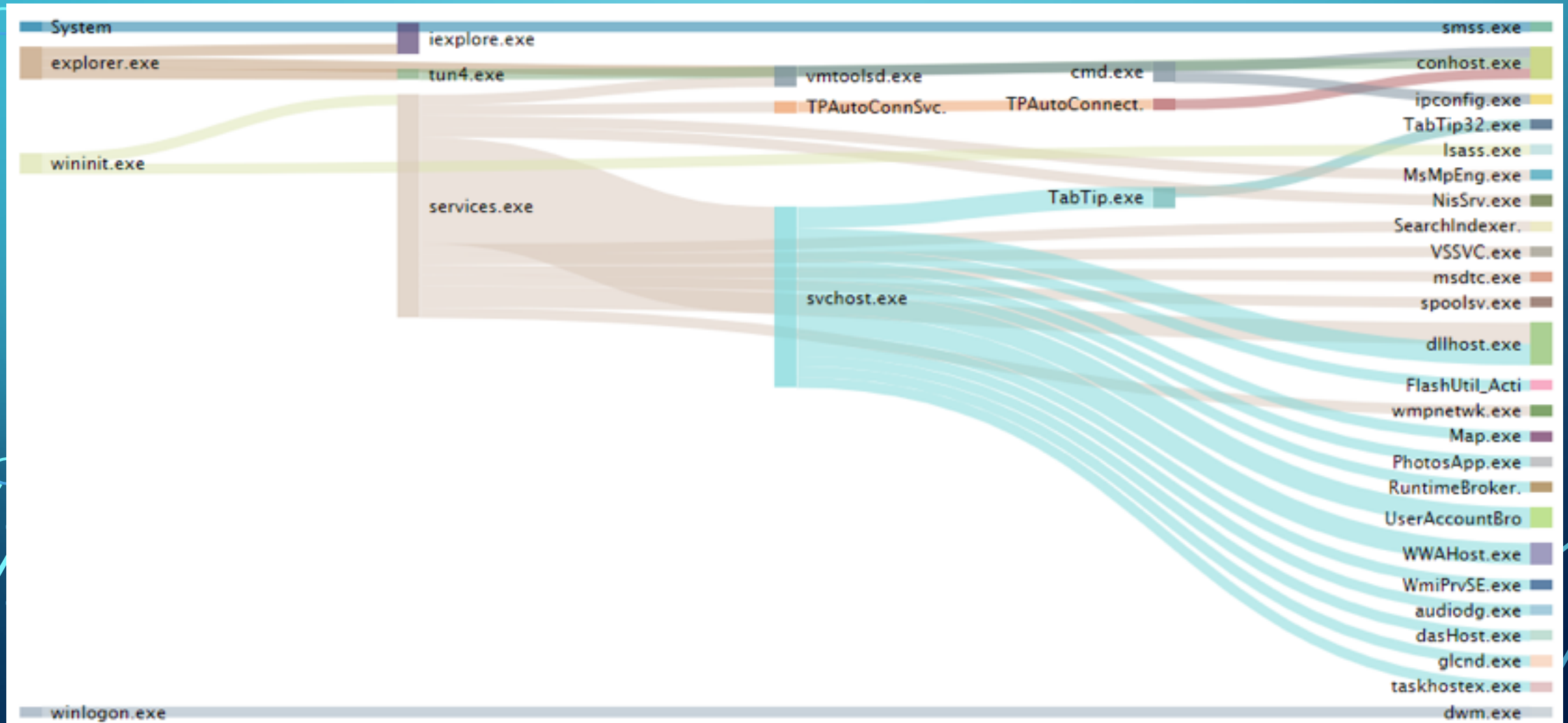
pid ↕ ✎	ppid ↕ ✎	name ↕ ✎	image ↕
3388	4020	ipconfig.exe	john_doe
1540	4020	conhost.exe	john_doe
4020	1308	cmd.exe	john_doe
3412	560	wmpnetwk.exe	john_doe
2120	636	glcnd.exe	john_doe
4016	636	Map.exe	john_doe
4024	636	PhotosApp.exe	john_doe
2108	636	FlashUtil_Acti	john_doe

IDENTIFY ORPHAN PROCESSES VISUALLY

```
index::main source="john_doe_pslist.json"  
| lookup volpids.csv pid as ppid image as image OUTPUTNEW  
name as parent  
| stats count by parent,name
```



volpids.csv



IDENTIFY ORPHAN VIA IN TABULAR FORMAT

```
index::main source="john_doe_pslist.json"  
| lookup volpids.csv pid as ppid image as image OUTPUTNEW  
name as parent  
| table name,pid,parent,ppid,image | where isnull(parent)
```



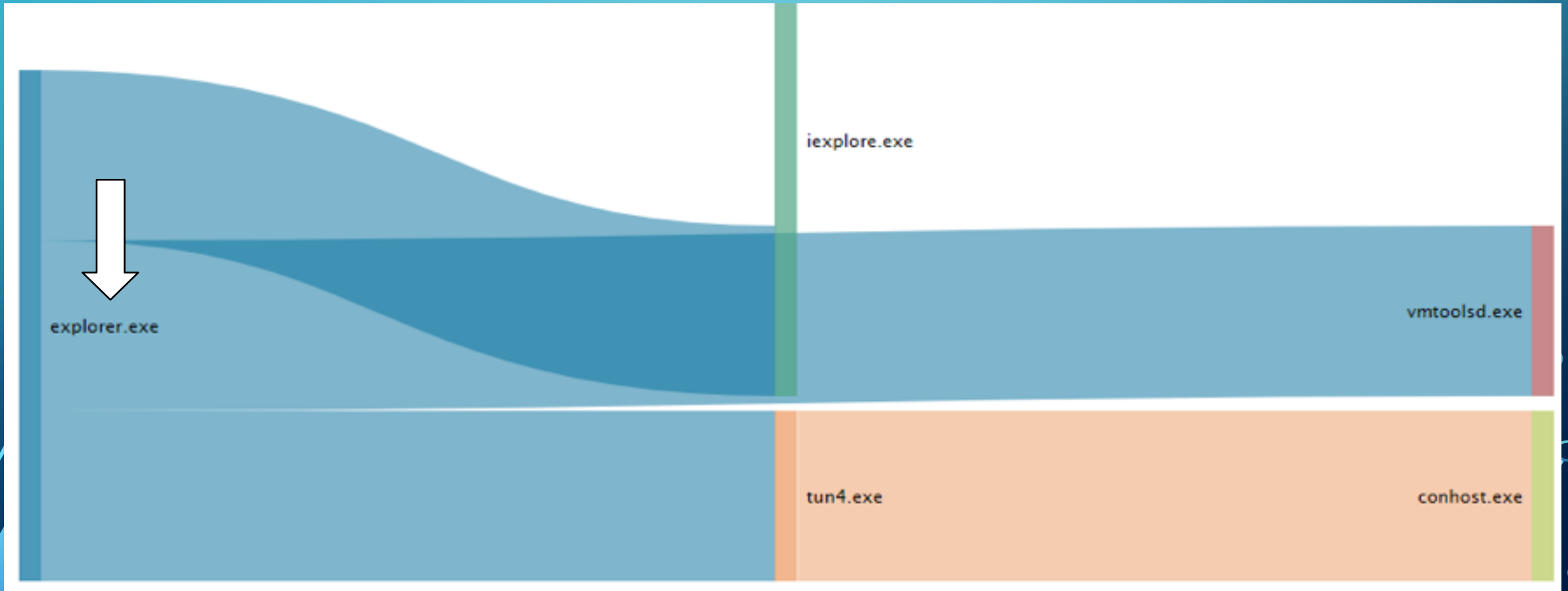
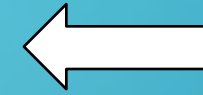
name ↕	pid ↕	parent ↕	ppid ↕	image ↕
explorer.exe	2088		2076	john_doe
winlogon.exe	536		472	john_doe
csrss.exe	492		472	john_doe
wininit.exe	480		388	john_doe
csrss.exe	396		388	john_doe
System	4		0	john_doe

TRACE PROCESS TREES OF ORPHANS VISUALLY

```
index::main source="john_doe_pslist.json"  
| table name,pid,ppid,image  
| listprocess root_process_id=2088 process_field=name  
ppid_field=ppid pid_field=pid  
| lookup volpids.csv pid as ppid OUTPUTNEW name as parent |  
stats count by parent,name
```



volpids.csv

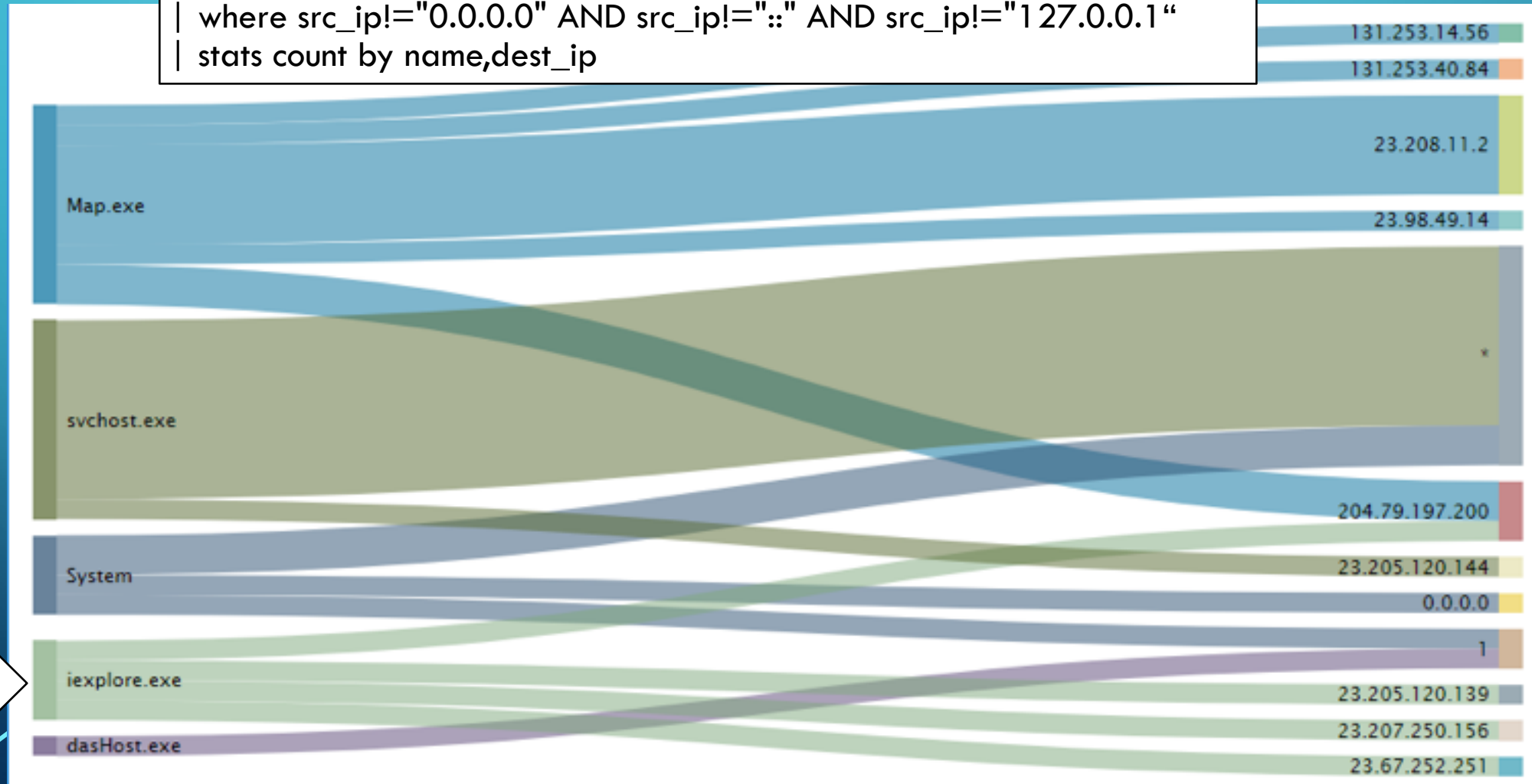


VISUALIZE NETWORK CONNECTIONS BY PROCESS

```
index::main source="john_doe_netscan.json"
| table foreignaddr,localaddr,pid,state
| lookup volpids.csv pid as pid OUTPUTNEW name as name
| rex field=localaddr "(?<src_ip>[^\:]+):(?<src_port>[^\$]+)"
| rex field=foreignaddr "(?<dest_ip>[^\:]+):(?<dest_port>[^\$]+)"
| where src_ip!="0.0.0.0" AND src_ip!="::" AND src_ip!="127.0.0.1"
| stats count by name,dest_ip
```



volpids.csv

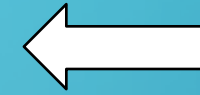


IDENTIFY ANOMALOUS PSXVIEW PROCESSES

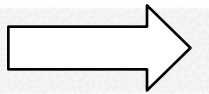
```
index::main source="john_doe_psxview.json"  
| eval total=0  
| foreach csrss pslist pspcid psscan session deskthrd thrdproc  
  [ eval total=if(<<FIELD>>="False",total+1,total) ]  
| table pid,name,image,csrss,pslist,pspcid,psscan,session,deskthrd,thrdproc,total  
| where total>1  
| sort - total
```



volpids.csv



pid ↕	name ↕	image ↕	csrss ↕	pslist ↕	pspcid ↕	psscan ↕	session ↕	deskthrd ↕	thrdproc ↕	total ↕
4684	LogonUI.exe	john_doe	False	False	False	True	False	False	False	6
284	smss.exe	john_doe	False	True	True	True	False	False	True	3
4	System	john_doe	False	True	True	True	False	False	True	3
4020	cmd.exe	john_doe	False	True	True	True	True	False	False	3
628	TabTip.exe	john_doe	False	True	True	True	True	False	False	3
3388	ipconfig.exe	john_doe	False	True	True	True	True	False	False	3



IDENTIFY COMMAND LINES OF INTEREST

```
index::main source="john_doe_cmdline.json" NOT (\\WINDOWS OR \\PROGRAM) |  
lookup volpids.csv name as process image as image OUTPUTNEW pid as pid  
| table pid,process,commandline
```



volpids.csv

pid	process	commandline
3388	ipconfig.exe	
4020	cmd.exe	
3596	tun4.exe	"C:\\Users\\clever\\tun4.exe"
2872	dasHost.exe	dashost.exe {d89b20d2-80fb-41c9-b2341e7f4be898bc}
2572	TabTip32.exe	/loadhooks /Parent:0000000000000990
2448	TabTip.exe	/QuitInfo:00000000000004D0;00000000000004D4;
2136	TPAutoConnect.	TPAutoConnect.exe -q -i vmware -a COM1 -F 30
2060	taskhostex.exe	taskhostex.exe

IDENTIFY MUTANTS/MUTEXES

```
index::main john_doe_handles.json Mutant
| lookup volpids.csv pid as pid OUTPUTNEW name as name
| stats values(details) as details by name
```



volpids.csv

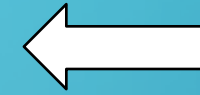
name	details
	{A3BD3259-3E4F-428a-84C8-F0463A9D3EB5}
taskhostx.exe	AccessibilitySoundAgentRunning CicLoadWinStaWinSta0 MSCTF.Asm.MutexDefault1 MSCTF.CtfMonitorInstMutexDefault1
tun4.exe	ZonesCacheCounterMutex ZonesLockedCacheCounterMutex uuJ9IBoKTbgOpUJQ
vmtoolsd.exe	.NET CLR Data_Perf_Library_Lock_PID_51c .NET CLR Networking 4.0.0.0_Perf_Library_Lock_PID_51c .NET CLR Networking_Perf_Library_Lock_PID_51c .NET Data Provider for Oracle Perf Library Lock PID 51c

CHECK MALFIND FOR INJECTION

```
index::main source="john_doe_malfind.json" page_execute_readwrite
| table process,pid,data,protection
| lookup volpids.csv pid as pid OUTPUTNEW ppid as ppid
| lookup volpids.csv pid as ppid OUTPUTNEW name as parent
| decrypt field=data unhex() emit('plaintext')
| table process,pid,parent,plaintext,protection
```



volpids.csv



process	pid	parent	plaintext	protection
wmpnetwk.exe	3412	services.exe o.h.....`.....@.A.....	PAGE_EXECUTE_READWRITE
wmpnetwk.exe	3412	services.exe5.....0.....X.....h.....(.....>.....	PAGE_EXECUTE_READWRITE
Map.exe	4016	svchost.exe8...hA.....(..L.....aL....@.kL....	PAGE_EXECUTE_READWRITE
Map.exe	4016	svchost.exex.....I.....H.@..M.....I.....H.@..M.....I.....	PAGE_EXECUTE_READWRITE
Map.exe	4016	svchost.exej..4.K......D.s....D.s....D.s.....s.....	PAGE_EXECUTE_READWRITE
Map.exe	4016	svchost.exeB.K.....s... .D.s....D.s....D.s.....	PAGE_EXECUTE_READWRITE
Map.exe	4016	svchost.exe@jD.s.....s....D.s.....s.... .s....p..s.....	PAGE_EXECUTE_READWRITE
Map.exe	4016	svchost.exeO.w..}.....s... ..s.....s.....s.....	PAGE_EXECUTE_READWRITE
iexplore.exe	3264	iexplore.exe~.....~.....~.....	PAGE_EXECUTE_READWRITE
UserAccountBro	1104	svchost.exe#.....	PAGE_EXECUTE_READWRITE
UserAccountBro	3916	svchost.exeY.....	PAGE_EXECUTE_READWRITE
audiodg.exe	3216	svchost.exe	@.....mU....C...\.W.i.n.d.o.w.s.\.S.Y.S.T.E.M.3.2.\.k.e.r.n.	PAGE_EXECUTE_READWRITE



BUILD A USERASSIST TIMELINE

```
index::main sourcetype::volatility source="john_doe_userassist.json"  
| eval _time=strptime(lastupdated, "%Y-%m-%d %H:%M:%S")  
| table lastupdated,value  
| sort - lastupdated
```



volpids.csv

lastupdated	value
14:26:02 UTC+0000	
2014-06-17 05:55:15 UTC+0000	%APPDATA%\Microsoft\Internet Explorer\Quick Launch\User Pinned\TaskBar\Internet Explorer.lnk
2014-06-17 05:52:23 UTC+0000	%windir%\system32\notepad.exe
2014-06-17 05:50:53 UTC+0000	%windir%\regedit.exe
2014-06-17 05:49:12 UTC+0000	C:\\Users\\clever\\Desktop\\tun4.exe
2014-06-17 05:39:11 UTC+0000	%windir%\system32\Taskmgr.exe
2014-06-17 05:38:25 UTC+0000	C:\\Users\\clever\\AppData\\Local\\Microsoft\\Windows\\Application Shortcuts\\microsoft.windowscommunicationsapps_8wekyb3d8bbwe\\Microsoft.WindowsLive.People.lnk

IDENTIFY LOADED MODULES BY PROCESS NAME

```
index::main sourcetype::volatility source="john_doe_ldrmodules.json" tun4.exe  
| lookup volpids.csv name as name OUTPUTNEW pid as pid  
| table pid,process,mappedpath,init,load,inmem
```



volpids.csv

pid	process	mappedpath	init	load	inmem
3596	tun4.exe	\\Windows\\SysWOW64\\dhcpcsvc6.dll	True	True	True
3596	tun4.exe	\\Windows\\SysWOW64\\rsaenh.dll	True	True	True
3596	tun4.exe	\\Windows\\SysWOW64\\profapi.dll	True	True	True
3596	tun4.exe	\\Windows\\SysWOW64\\dnsapi.dll	True	True	True
3596	tun4.exe	\\Windows\\SysWOW64\\mswsock.dll	True	True	True
3596	tun4.exe	\\Windows\\SysWOW64\\clbcatq.dll	True	True	True
3596	tun4.exe	\\Windows\\SysWOW64\\bcrypt.dll	True	True	True
3596	tun4.exe	\\Windows\\SysWOW64\\wininet.dll	True	True	True

MULTI-SOURCE ANALYSIS

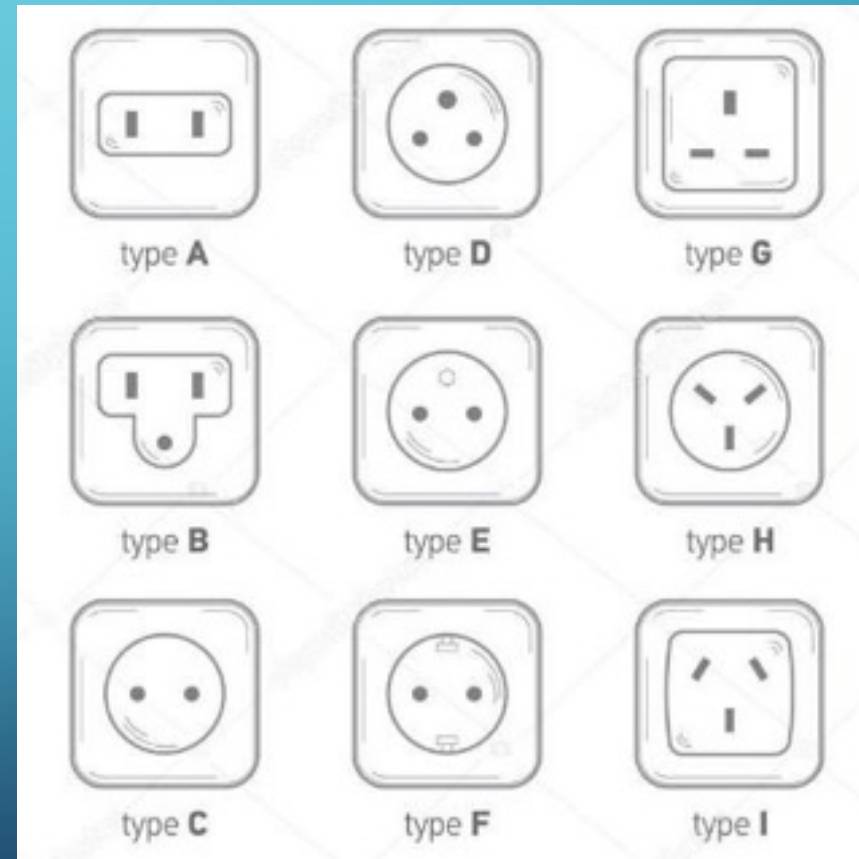
- Frequency of Occurrence by Host
- Unique Relationships
- Markov Chain Anomaly Detection
- Predictive Analytics
- Behavior Baseline Deviation



SPLUNK MACHINE
LEARNING TOOLKIT

WHAT ABOUT "THOSE" PLUGINS?

- Contributed
- Not Unified Output capable
- New Hotness
- Old and Janky



SPICE IT UP WITH SOME JSON

- Identify fields in plugin code in order of presentation
- Open plugin in test editor
- Insert “from volatility.renderers import TreeGrid” in import declarations
- Add generator function
- Add unified_output function
- Save plugin file

SEE, ITS SO EASY!



```
# its a socket!
def render_text(self, outfd, data):
    linux_common.set_plugin_members(self)

    if not self.addr_space.profile.has_type("inet_sock"):
        # ancient (2.6.9) centos kernels do not have inet_sock in debug info
        raise AttributeError, "Given profile does not have inet_sock, please file a bug if the kernel version is > 2.6.11"

    for task in data:
        for ents in task.netstat():
            if ents[0] == socket.AF_INET:
                (_, proto, saddr, sport, daddr, dport, state) = ents[1]
                outfd.write("{0:8s} {1:<16s}:{2:>5} {3:<16s}:{4:>5} {5:<15s} {6:>17s}/{7:<5d}\n".format(proto, saddr, sport, daddr, dport, state, task.comm, task.pid))

            elif ents[0] == 1 and not self._config.IGNORE_UNIX:
                (name, inum) = ents[1]
                outfd.write("UNIX {0:<8d} {1:>17s}/{2:<5d} {3:s}\n".format(inum, task.comm, task.pid, name))
```

```
linux_plugin.LinuxPlugin.__init__(self, config, args, kwargs)
self._config.add_option('IGNORE_UNIX', short_option = 'U', default = False)

def unified_output(self, data):
    return TreeGrid([("Proto", str),
                     ("Local IP", str),
                     ("Local Port", int),
                     ("Remote IP", str),
                     ("Remote Port", int),
                     ("State", str),
                     ("Process", str),
                     ("PID", str),
                     ("Name", str),
                     ],
                    self.generator(data))

def generator(self, data):
    for task in data:
        for ents in task.netstat():
            if ents[0] == socket.AF_INET:
                (_, proto, saddr, sport, daddr, dport, state) = ents[1]
                yield(0, [
                    str(proto),
                    str(saddr),
                    int(sport),
                    str(daddr),
                    int(dport),
                    str(state),
                    str(task.comm),
                    str(task.pid),
                    str(name),
                ])

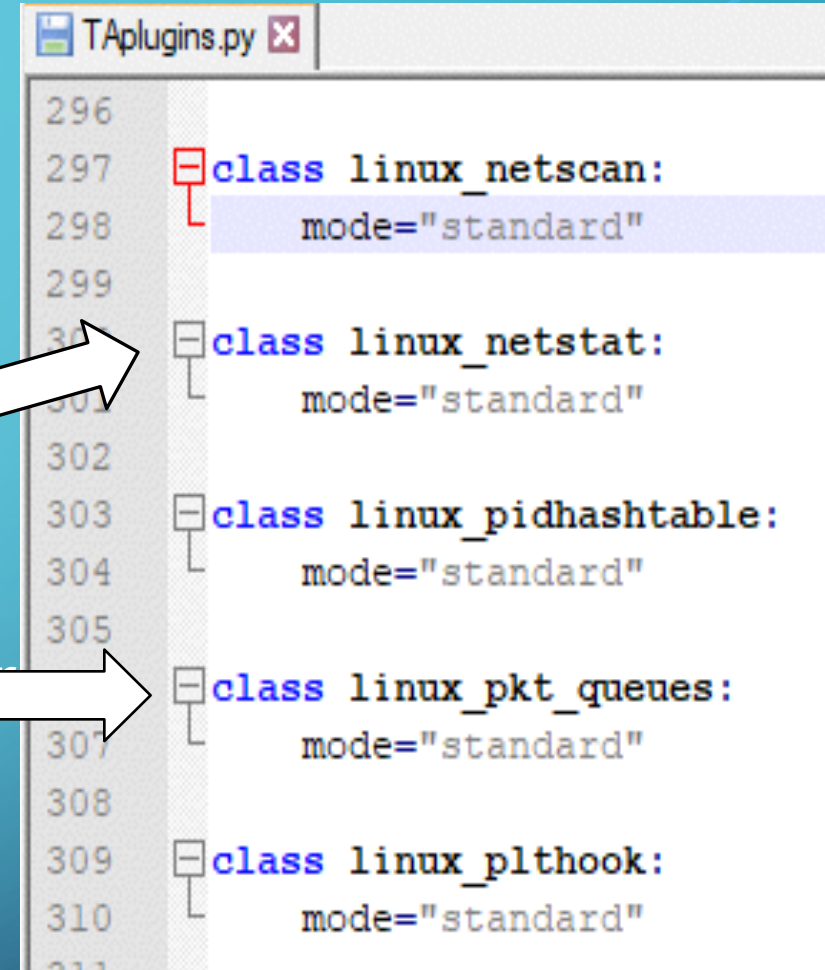
            elif ents[0] == 1 and not self._config.IGNORE_UNIX:
                (name, inum) = ents[1]
                yield(0, [
                    str("UNIX "+str(inum)),
                    "-",
                    0,
                    "-",
                    0,
                    "-",
                    str(task.comm),
                    str(task.pid),
                    str(name),
                ])

# its a socket!
```

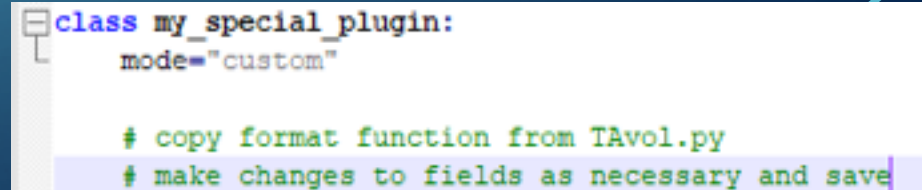

EXTENDING TA-VOLATILITY

- Probably ALREADY in the TAplugins.py classes
- If not, add your plugin name and mode="standard"
- UNLESS you need custom parsing!
 - Format function start

```
def format(self, file_sep, base_dir, arch, plugin, host):  
    #Set the container to None  
    container = None
```



```
296  
297 class linux_netscan:  
298     mode="standard"  
299  
300 class linux_netstat:  
301     mode="standard"  
302  
303 class linux_pidhashtable:  
304     mode="standard"  
305  
306 class linux_pkt_queues:  
307     mode="standard"  
308  
309 class linux_plthook:  
310     mode="standard"  
311
```



```
class my_special_plugin:  
    mode="custom"  
  
    # copy format function from TAvol.py  
    # make changes to fields as necessary and save
```

BUDGETARY STUFF

- Memory metadata really isn't a lot of data
- Everything was performed on a Trial version (500 MB / day)
- Licenses aren't THAT expensive
- License costs are situationally dependent, try speaking human to them

GENERAL SPLUNK LICENSE COSTS

GET ACTIONABLE INTELLIGENCE

Splunk® Enterprise

STARTS AT

\$150

Per Ingested GB, Per Month, Billed Annually*

SEARCH AND ANALYZE

Splunk® Light

STARTS AT

\$75

Per Ingested GB, Per Month, Billed Annually*

EXPLORE

Splunk® Free

WAY AHEAD

- Data Model / Common Information Model
- Rekall Live Memory Agent App
- Memory Analysis Splunk App
- Phantom Automation
- Enterprise Security Compliance





memegenerator.net