

Tracking mimikatz by Sysmon and Elasticsearch

Aug 26, 2017

*Interfaculty Initiative in Information Studies Graduate School of
Interdisciplinary Information Studies,
The University of Tokyo*

Wataru Matsuda, Mariko Fujimoto¹

Profile

- Name: Wataru Matsuda
- Position/Affiliation: Project researcher, Secure Information Society Research Group, the University of Tokyo (SiSOC)
- Job description:
 - Analysis and publication on cyber security
 - Education for human resources for cyber security in critical infrastructure
 - Presentations and lectures in seminars/universities etc.
- Publication : CSIRT – from building to running – (coauthor)



Profile

- Name: Mariko Fujimoto
- Position/Affiliation: Project researcher, Secure Information Society Research Group, the University of Tokyo (SiSOC)
- Job description:
 - Analysis and publication on cyber security
 - Education for human resources for cyber security in critical infrastructure
 - Presentations and lectures in seminars/universities etc.
- Works: Articles of Vulnerability in Apache Commons Collections
- <https://codezine.jp/article/detail/9150>

シリアライズ・デシリアライズとは

シリアライズとは、プログラムで扱うオブジェクトやデータを保存したまま送受信したり、ファイルに保存するなどの処理を指します。また、ネットワークやデータベースなどにも応用されています。デシリアライズとは、シリアライズされたデータを元のオブジェクトやデータに変換する処理を指します。シリアライズとデシリアライズは、データの保存や送受信に不可欠な処理です。



This article explains deserialize which triggered Apache Commons Collections vulnerability

About Secure Information Society Research Group, the University of Tokyo

- SiSOC-TOKYO researches on Internet security through collaboration with industry, academia and government.
 - SiSOC-TOKYO gathers human resources through collaboration among industries, academia and government to research on social and international issues and widely reports on the analysis results.
 - SiSOC-TOKYO promotes interdisciplinary research, human resource education and policy recommendation against issues on cyber space and security from a macro and long-term perspective.

Agenda

- Background
- Challenges
- Previous Researches
- Proposed Methods
- Result
- Demonstration
- Conclusion

Background

Background

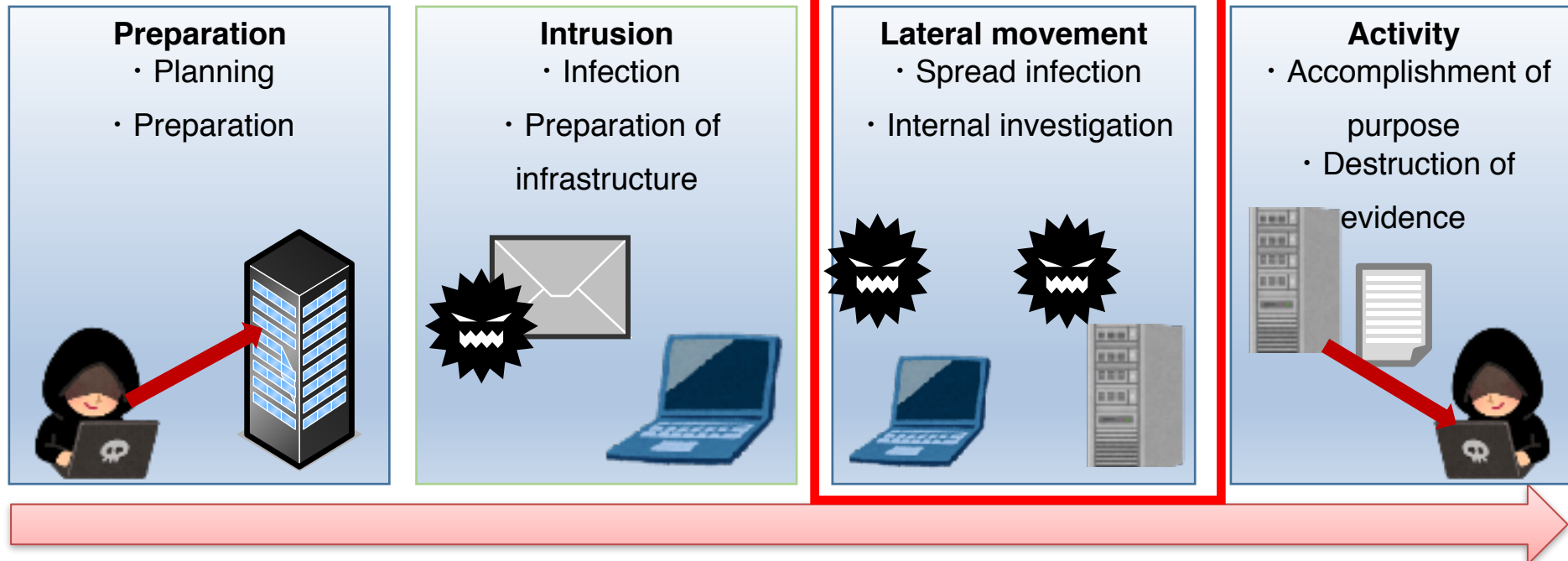
- Active Directory is often leveraged in targeted attacks
- Attack tool “mimikatz” is often used for compromising Active Directory



**mimikatz hacked
my Active Directory!!**

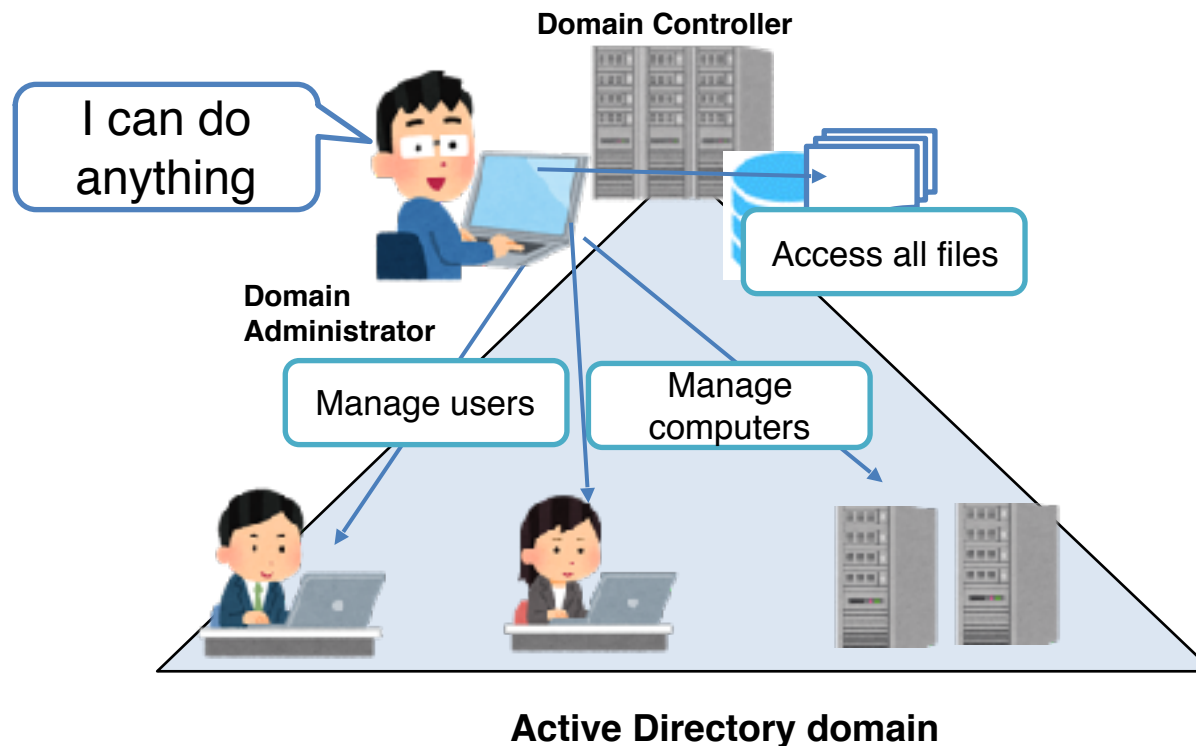
Lateral movement in targeted attacks

- Targeted attacks today are becoming more sophisticated
- Attackers remain within the systems and conduct lateral movement
- It is difficult to completely prevent intrusions – early detection of lateral movement is a key for minimizing damage



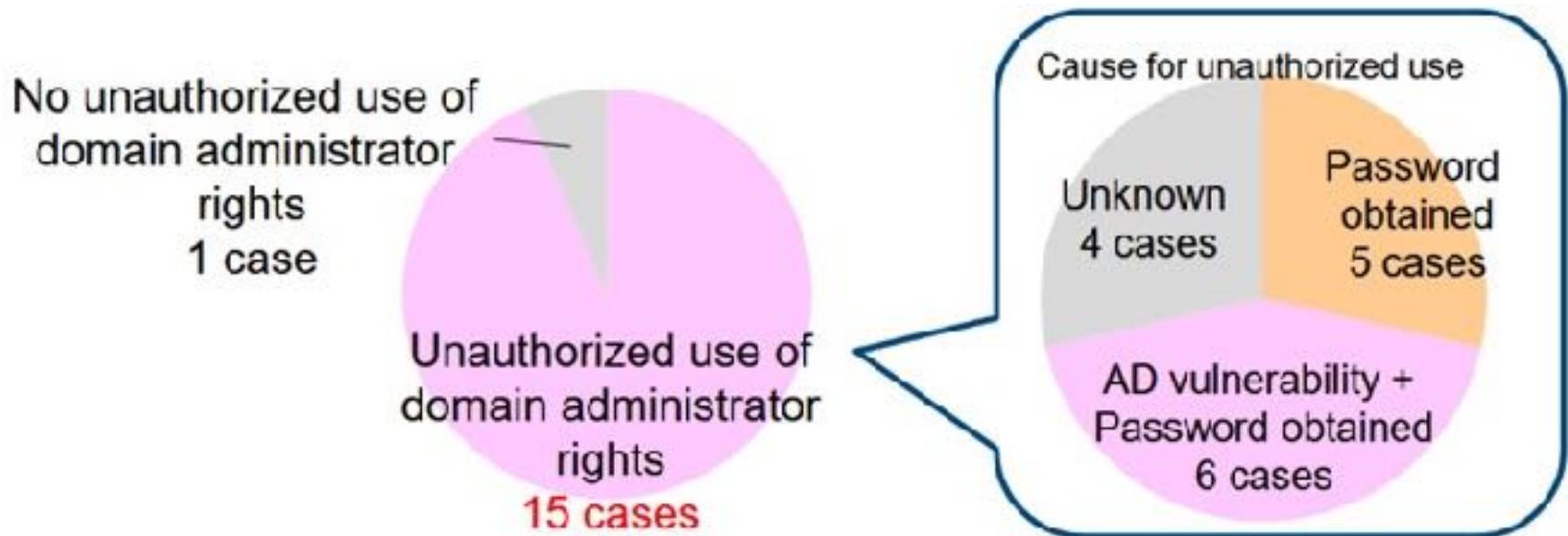
Overview of Active Directory

- What is Active Directory?
 - A system to centrally manage Windows computers and users
- What is a domain?
 - A management unit for computers and users
- What is a domain controller?
 - An authentication server to manage domains



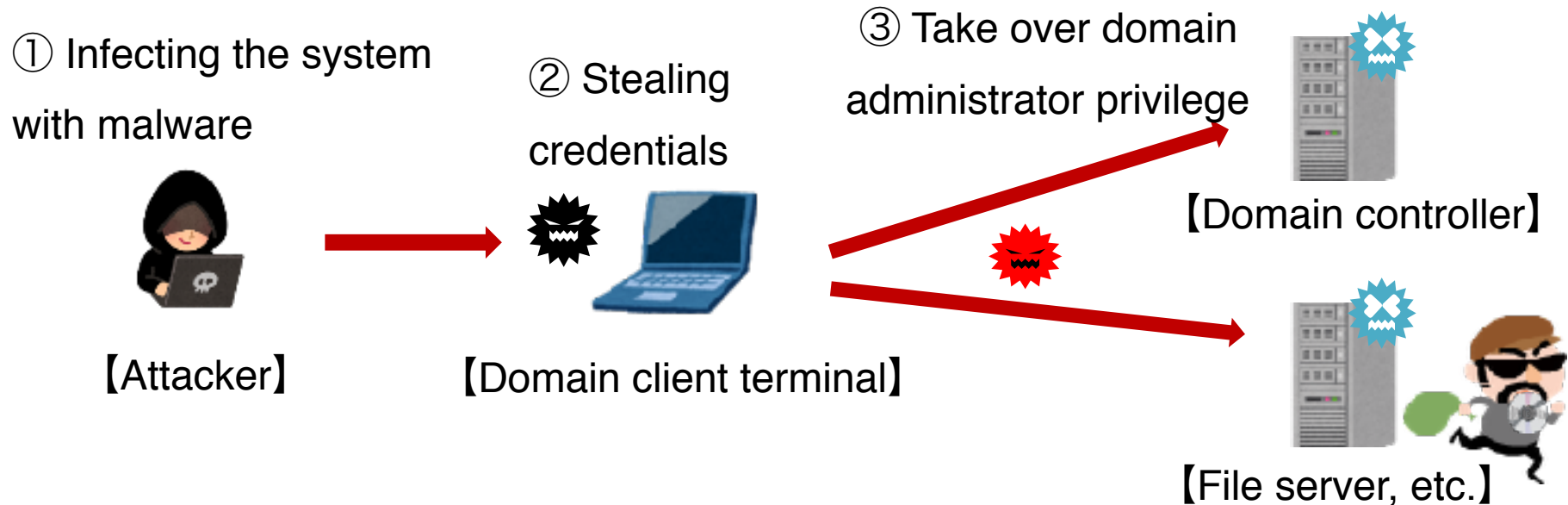
Attack on Active Directory

- According to JPCERT/CC, in many targeted attack cases, Active Directory was under attack
(from <https://www.first.org/resources/papers/conf2016/FIRST-2016-105.pdf>)
- Number of confirmed cases for unauthorized use of Domain Administrator privilege handled by JPCERT/CC in 2015




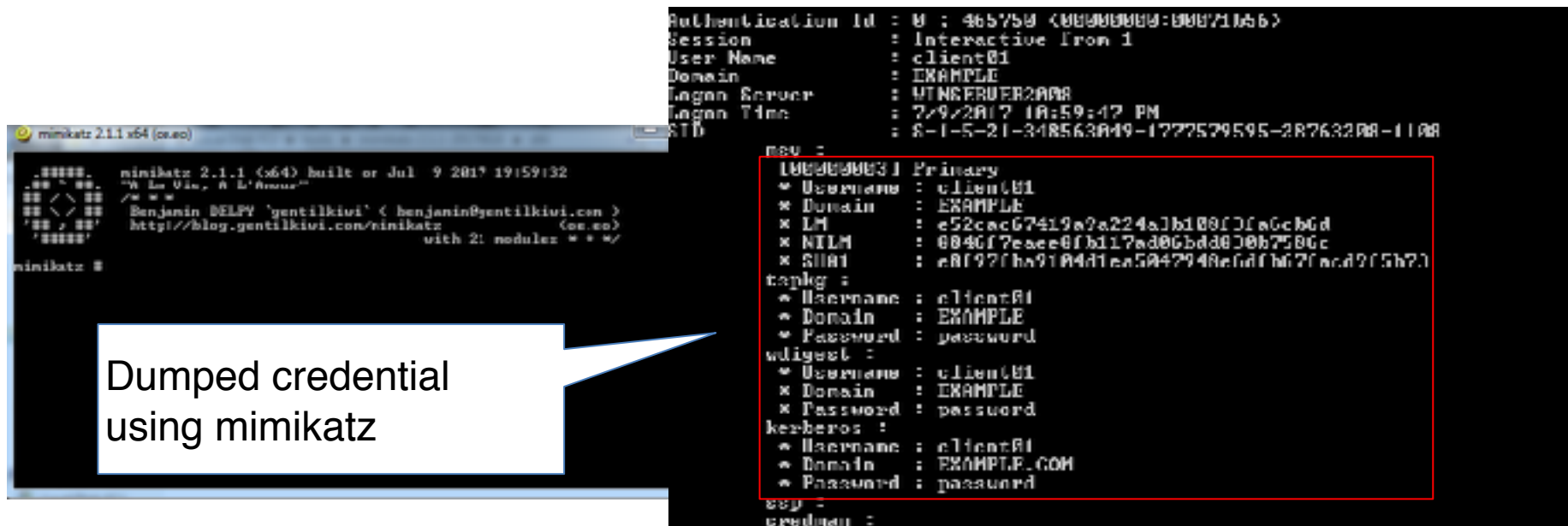
Attack on Active Directory

- After intruding into an organization, attacker will try to steal the domain administrator right and spread the infection to eventually gain access to information



What's "mimikatz"?

- A common attack tool against Windows computers
 - Show credentials stored in a computer (required local admin privilege)
 - Create forged Kerberos ticket / import to memory
- Updates often released with enhanced features
- Trademark is 



```

mimikatz 2.1.1 x64 (cs.exe)

..... mimikatz 2.1.1 (x64) built on Jul  9 2017 19:59:32
..  /  .. "W Lm Vlm, A L'Amour"
..  /  .. /w w w
'.. j ..' Benjamin DELPY 'gentilkiwi' ( benjamin@gentilkiwi.com )
'.. j ..' http://blog.gentilkiwi.com/mimikatz (cs.exe)
'.....' with 21 modules * * w/

mimikatz #

Authentication Id : 0 : 465750 (00000000:000710a6)
Session           : Interactive from 1
User Name         : client01
Domain           : EXAMPLE
Logon Server      : WINSERVER2008
Logon Time        : 7/9/2017 10:59:47 PM
SID              : S-1-5-21-348563049-1772579595-28763208-1108

lsasrv :
LSASSRV0003 Primary
* Username : client01
* Domain   : EXAMPLE
* LM       : e52cac67419a224a1b100f3fa6cb6d
* NtLm     : 0040f7eace0fb117ad06bdd030b7586c
* SHA1     : a8f92fba9104d1ea5047948e6dfbf7f6cd9f5b73
token :
* Username : client01
* Domain   : EXAMPLE
* Password : password
wdigest :
* Username : client01
* Domain   : EXAMPLE
* Password : password
kerberos :
* Username : client01
* Domain   : EXAMPLE.COM
* Password : password

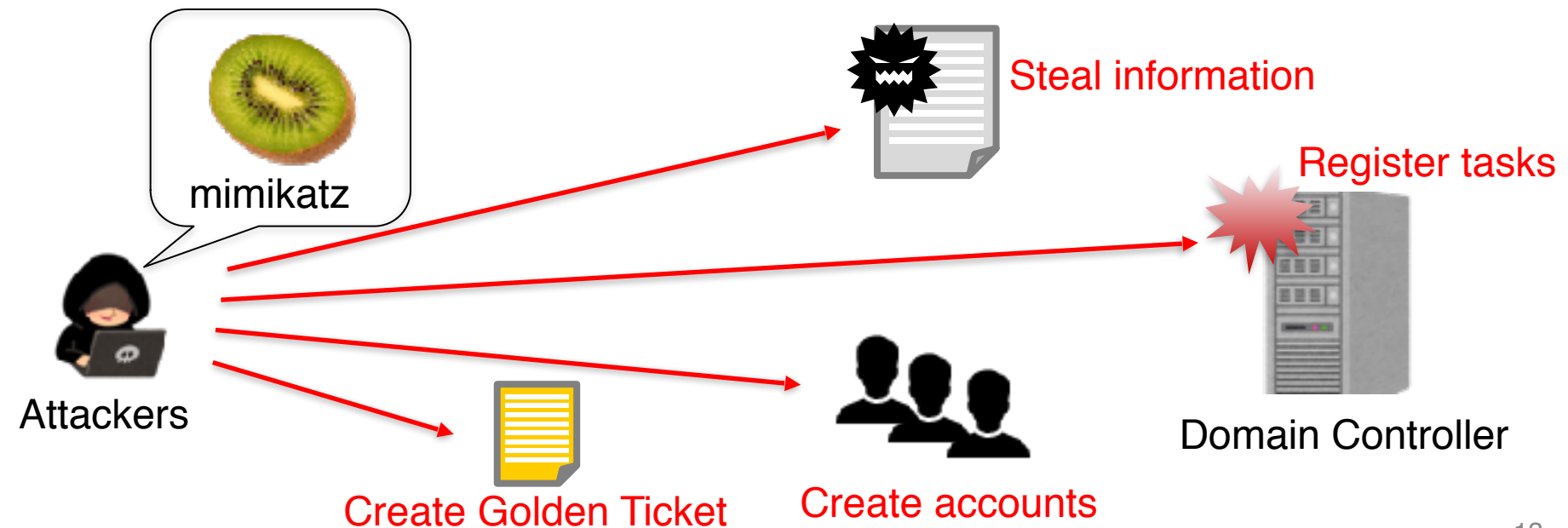
ssd -
credman :
  
```

Dumped credential using mimikatz

Attack flow using mimikatz

After intrusion, attackers use mimikatz to...

- Steal credentials stored in computers
- Access other computers using the stolen credentials
- Gain Domain Administrator privilege by leveraging Active Directory vulnerability (MS14-068)
- Steal confidential information stored in the computers
- Prepare a backdoor “Golden Ticket” to continue attacks



Challenges

Challenges

- It is difficult to detect execution of programs including mimikatz since it is not logged under the Windows default settings
- Previous research suggests using Sysmon to detect DLL files loaded by mimikatz. However, it was tested under specific Windows/mimikatz versions and may cause false detection in other environments
- In this research, we analyzed DLLs that are loaded in each Windows/mimikatz version to find out which DLL loading should be detected in order to increase accuracy
- This presentation also introduces methods of even more effective detection

What's Sysmon?

- A free tool provided by Microsoft. It monitors running processes and loaded DLLs and records in windows event log
- It enables tracking process including mimikatz and is useful for security purposes

Image:
Parent process name
which loads DLL.

ImageLoaded:
DLL information loaded
by mimikatz.exe.

```
Image loaded:  
UtcTime: 2017-08-05 06:52:50.455  
ProcessGuid: {87186166-6b41-5985-0000-001088103c0b}  
ProcessId: 10572  
Image: C:\tools\20170801\x64\mimikatz.exe  
ImageLoaded: C:\Windows\System32\wintrust.dll  
Hashes: MD5=5385E2D12C3F91828CD988FE6B20A506  
Signed: true  
Signature: Microsoft Windows  
SignatureStatus: Valid
```


Previous Research

Previous Research

- Some research has proposed methods using Sysmon to detect DLLs loaded by mimikatz
 - Detecting In-Memory mimikatz
Jake Liefer
<https://securityriskadvisors.com/blog/post/detecting-in-memory-mimikatz/>
 - Chronicles of a Threat Hunter: Hunting for In-Memory mimikatz with Sysmon and ELK - Part I (Event ID 7)
Roberto Rodriguez
<https://cyberwardog.blogspot.jp/2017/03/chronicles-of-threat-hunter-hunting-for.html>

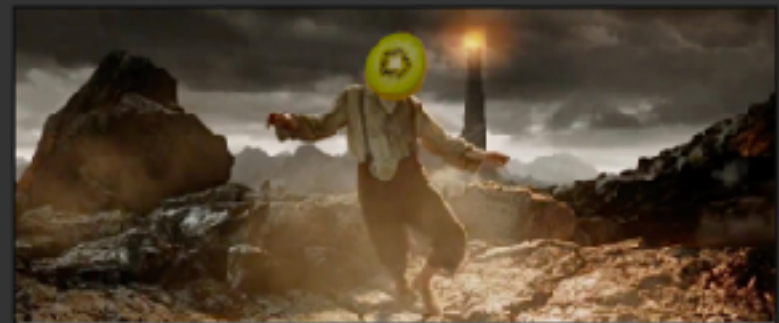


Detecting In-Memory Mimikatz

May 18, 2016 | Posted in [Blue Teams](#) by [Jake Liefer](#)

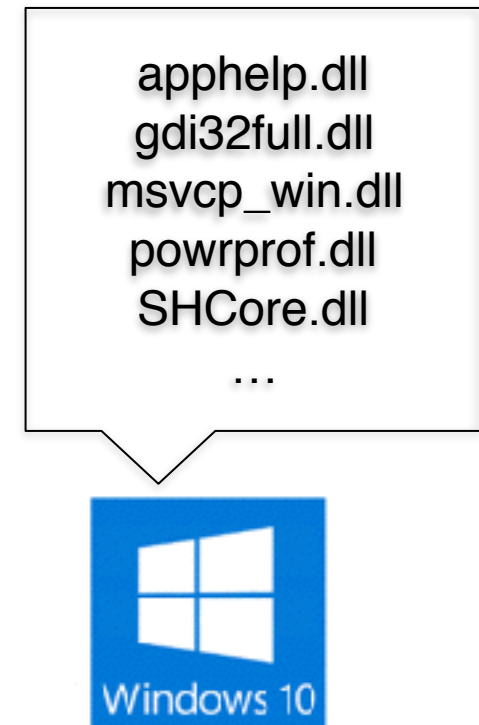
One of the most pressing internal network security issues is limiting the ability of an attacker to perform privilege escalation. In my experience, once administrative level access is obtained to a Windows system it is trivial for an attacker to dump user credentials and pivot throughout the network, eventually gaining the most privileged accounts without detection.

Chronicles of a Threat Hunter: Hunting for In-Memory Mimikatz with Sysmon and ELK - Part I (Event ID 7)



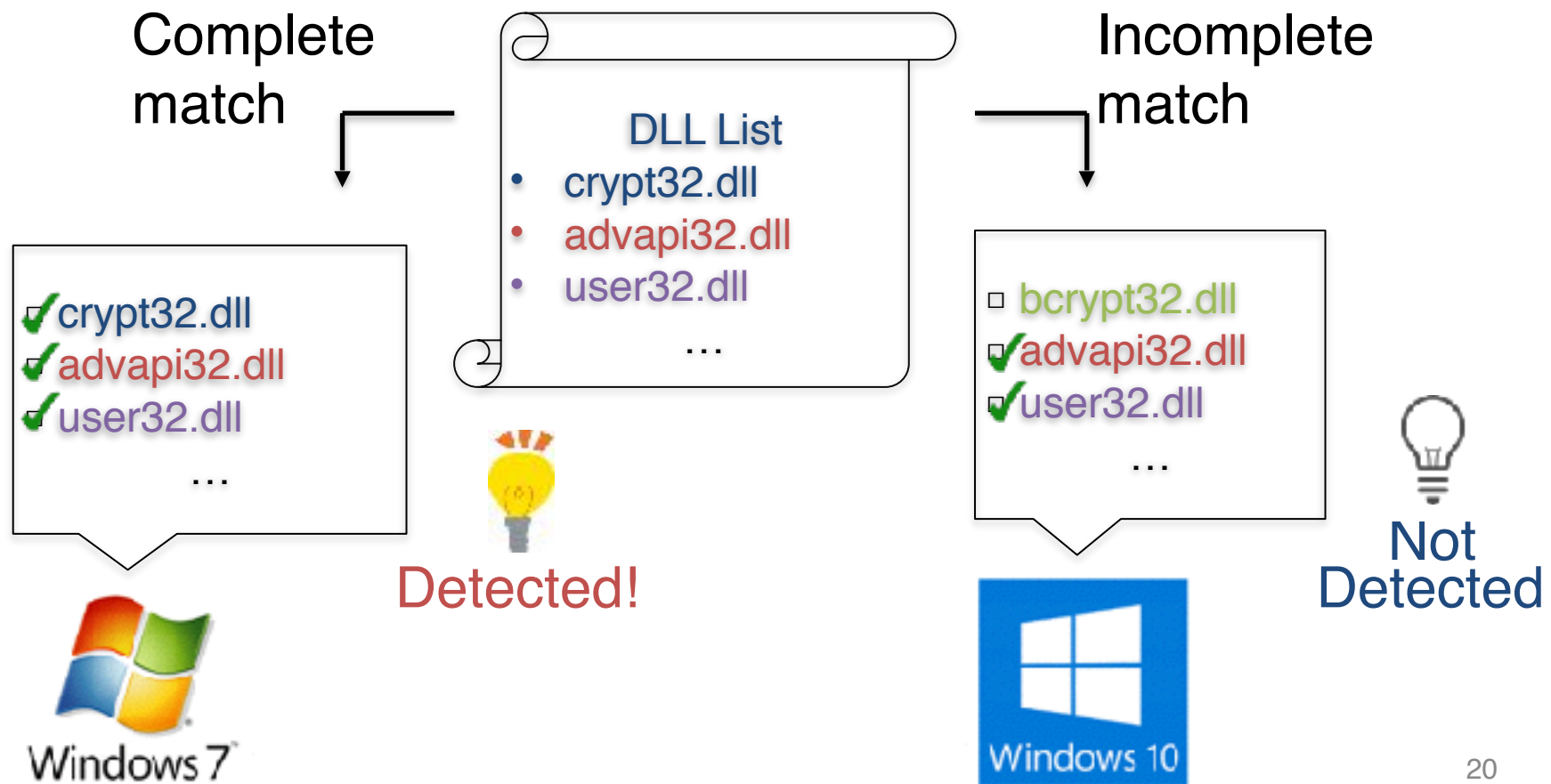
Previous Research

- These proposals focus on the detection methods but do not take environments into consideration such as Windows/mimikatz versions
- We have found that loaded DLLs are different depending on Windows/mimikatz versions



Observations on Previous Research

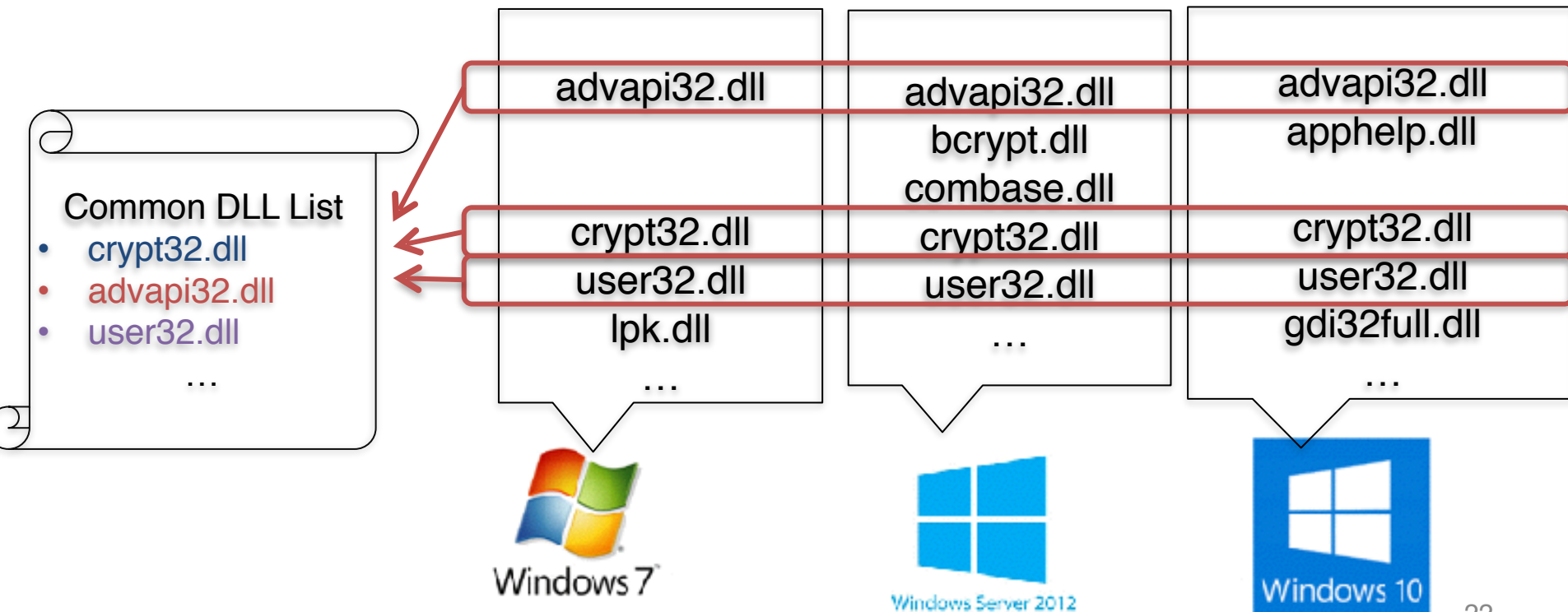
- Due to the difference in DLLs are loaded in different environment, it is necessary to clarify which DLL loading should be detected in order to identify mimikatz execution



Proposed Methods

Proposed Methods

- We examine difference of mimikatz DLL loading with various Windows/ mimikatz versions
- We propose that false negative rate can be reduced by focusing on common DLLs that are loaded regardless of environment
- Additionally, detection accuracy can be improved by checking for the DLL loading that are specific to an environment



Proposed Methods

- We created a list of DLLs(Common DLL List) that are commonly loaded in the combination of the below environment

Windows Client OS

- Windows 7 64bit
- Windows 7 32bit
- Windows 8.1 64bit
- Windows 8.1 32bit
- Windows 10 64bit
- Windows 10 32bit



Windows Server OS

- Windows Server 2008 R2 64bit
- Windows Server 2012 R2 64bit
- Windows Server 2016 64bit



mimikatz



- mimikatz 2.0 alpha (May 2 2015)
- mimikatz 2.1 (May 1 2016)
- mimikatz 2.1.1 (Aug 1 2017)

Proposed Methods

- mimikatz versions are selected as follows:
 - mimikatz 2.1.1 (Aug 1 2017): Latest version
 - mimikatz 2.1 (May 1 2016): Released around the time when difference in loaded DLL was reported*
 - mimikatz 2.0 alpha (May 2 2015): Released when targeted attacks started to increase in Japan

* Detecting In-Memory mimikatz

Jake Liefer

<https://securityriskadvisors.com/blog/post/detecting-in-memory-mimikatz/>

Detection rate of the proposed methods

- We checked if mimikatz execution is correctly detected based on the Common DLL List
- How to evaluate detection rate:
 - Performs operation that are typical for general business operation
 - Run mimikatz several times during above operation. We used mimikatz binary and run as administrator
 - Compare the loaded DLLs in Sysmon log with the Common DLL List
- Environment:
 - Client device: Windows 7/10(x64)
 - DC : Windows Server 2008 R2



- Power on
- Logon / Logoff
- E-mail
- Microsoft Word
- Microsoft Excel
- Microsoft PowerPoint
- Internet Explorer
- File server access
- Command prompt
- Power shell

Run mimikatz
during operation

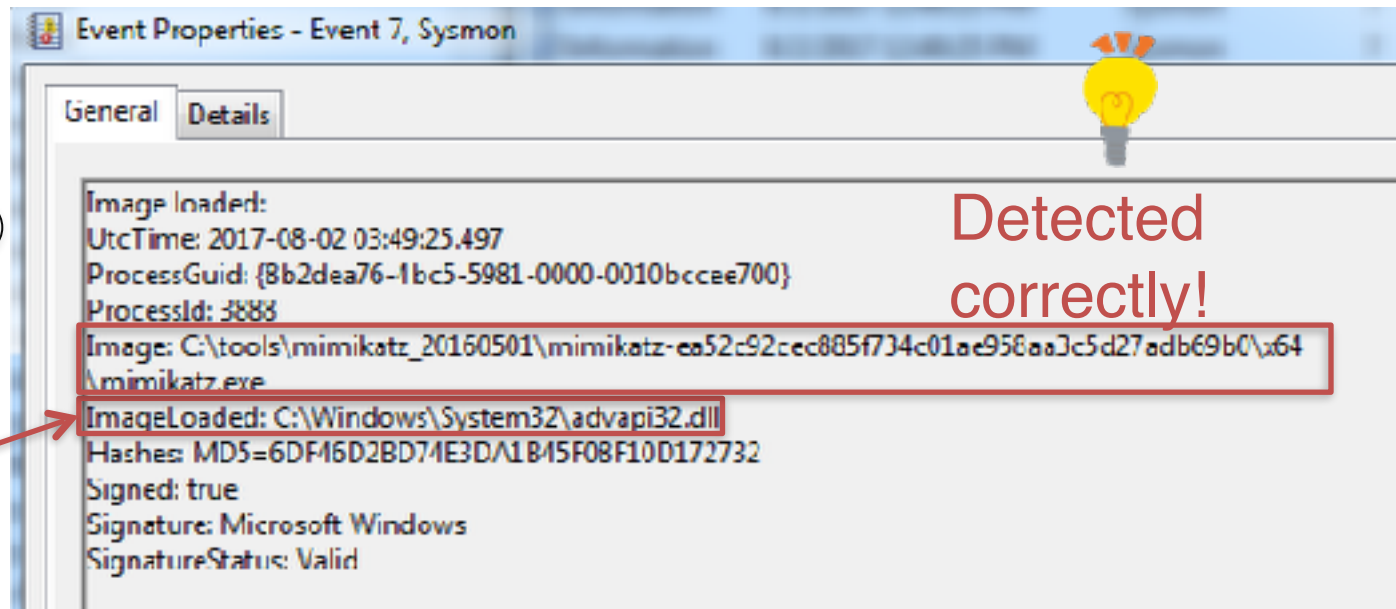
Detection rate of the proposed methods

1. Check for processes and DLLs etc. in the windows event logs
 2. Check if any of the DLLs match the Common DLL List
 3. Check the process that loads all DLLs in the Common DLL List
- If it is “mimikatz.exe”, mimikatz execution is correctly detected

Common DLL List

- crypt32.dll
- advapi32.dll
- user32.dll

...



Result

Excerpt from result

- Difference among Windows versions (7, 8.1, 10) × mimikatz 2.1.1 (Aug 1 2017)
- DLLs that are commonly loaded regardless of the Windows version are highlighted in yellow

	Win7x64_Mimikatz 2.1.1 (Aug 1 2017)	Win8.1x64_Mimikatz 2.1.1 (Aug 1 2017)	Win10x64_Mimikatz 2.1.1 (Aug 1 2017)
C:\Windows\System32\gdi32full.dll	-	-	○
C:\Windows\System32\hid.dll	○	○	○
C:\Windows\System32\imm32.dll	○	○	○
C:\Windows\System32\kernel.appcore.dll	-	○	○
C:\Windows\System32\kernel32.dll	○	○	○
C:\Windows\System32\KernelBase.dll	○	○	○
C:\Windows\System32\logoncli.dll	○	○	○
C:\Windows\System32\lpk.dll	○	-	-

⋮

Excerpt from result

- Difference among mimikatz versions (Windows 7 64bit)
- DLLs that are commonly loaded regardless of the mimikatz version are highlighted in yellow

	Win7x64_Mimikatz 2.0 (May 2 2015)	Win7x64_Mimikatz 2.1 (May 1 2016)	Win7x64_Mimikatz 2.1.1 (Aug 1 2017)
C:\Windows\System32\advapi32.dll	○	○	○
C:\Windows\System32\apphelp.dll	-	-	-
C:\Windows\System32\bcrypt.dll	○	○	-
C:\Windows\System32\bcryptprimitives.dll	-	-	-
C:\Windows\System32\cfgmgr32.dll	-	○	○
C:\Windows\System32\combase.dll	-	-	-
C:\Windows\System32\crypt32.dll	○	○	○
C:\Windows\System32\cryptbase.dll	-	-	○
C:\Windows\System32\cryptdll.dll	○	○	○

⋮

Result

List of DLLs that are commonly loaded in supported Windows and 3 mimikatz versions (Common DLL List)

No	Common DLLs	No	Common DLLs
1	C:\Windows\System32\advapi32.dll	11	C:\Windows\System32\rpcrt4.dll
2	C:\Windows\System32\crypt32.dll	12	C:\Windows\System32\rsaenh.dll
3	C:\Windows\System32\cryptdll.dll	13	C:\Windows\System32\saml.dll
4	C:\Windows\System32\gdi32.dll	14	C:\Windows\System32\sechost.dll
5	C:\Windows\System32\imm32.dll	15	C:\Windows\System32\secur32.dll
6	C:\Windows\System32\kernel32.dll	16	C:\Windows\System32\shell32.dll
7	C:\Windows\System32\KernelBase.dll	17	C:\Windows\System32\shlwapi.dll
8	C:\Windows\System32\msasn1.dll	18	C:\Windows\System32\sspicli.dll
9	C:\Windows\System32\msvcrt.dll	19	C:\Windows\System32\user32.dll
10	C:\Windows\System32\ntdll.dll	20	C:\Windows\System32\vaultcli.dll

Result

- Total amount of processes:3408
- False positive rate:0%
- False negative rate:0%

OS	Total amount of processes	False positive rate(%)	False negative rate(%)
Windows 7(x64)	1102	0	0
Windows Server 2008 R2	2016	0	0
Windows 10(x64)	290	0	0

How to use the analysis results

We propose to compare the DLLs with the Common DLL List, then compare with the DLL list for each environment to improve detection accuracy and detect mimikatz execution effectively

Step 1 (Quick investigation)

- Extract Process ID, DLL (ImageLoaded), Parent Process (Image) etc. from Event ID7 (Image loaded)
- If there is any process that loads all the DLLs in the Common DLL List, it is likely that mimikatz was executed on the computer. Go on to Step 2



Step 2 (Detailed investigation)

Compare the loaded DLLs with the DLL List for the specific Windows version. If there is any process that loads all the DLLs in the list, it is likely that mimikatz was executed on the computer.

Observation on the analysis result

- More DLLs tend to be loaded in newer versions of Windows and mimikatz
- Even for the same Windows/mimikatz versions, there are differences in loaded DLLs depending on applied Windows patches or software versions (e.g. PowerShell 2.0/5.0)

	Win7x64 (PowerShell2.0) Mimikatz 2.1.1 (Aug 1 2017)	Win7x64 (PowerShell5.0+latest patches) Mimikatz 2.1.1 (Aug 1 2017)
C:\Windows\System32\advapi32.dll	○	○
C:\Windows\System32\apphelp.dll	-	-
C:\Windows\System32\bcrypt.dll	-	○
C:\Windows\System32\bcryptprimitives.dll	-	○
C:\Windows\System32\cfgmgr32.dll	○	○
C:\Windows\System32\combase.dll	-	-
C:\Windows\System32\crypt32.dll	○	○

⋮

Demonstration

Create Common DLL List and detect mimikatz using each event log

- We provide sample tools for helping to create Common DLL List and detect mimikatz from exported event log
- Event logs can be exported as CSV files using windows built-in function
- It is also possible to detect execution of mimikatz using exported event log
- Sample programs on github
 - https://github.com/sisoc-tokyo/mimikatz_detection/tree/master/javaTool

1: Create Common DLL List

① Sysmon records DLLs to event logs

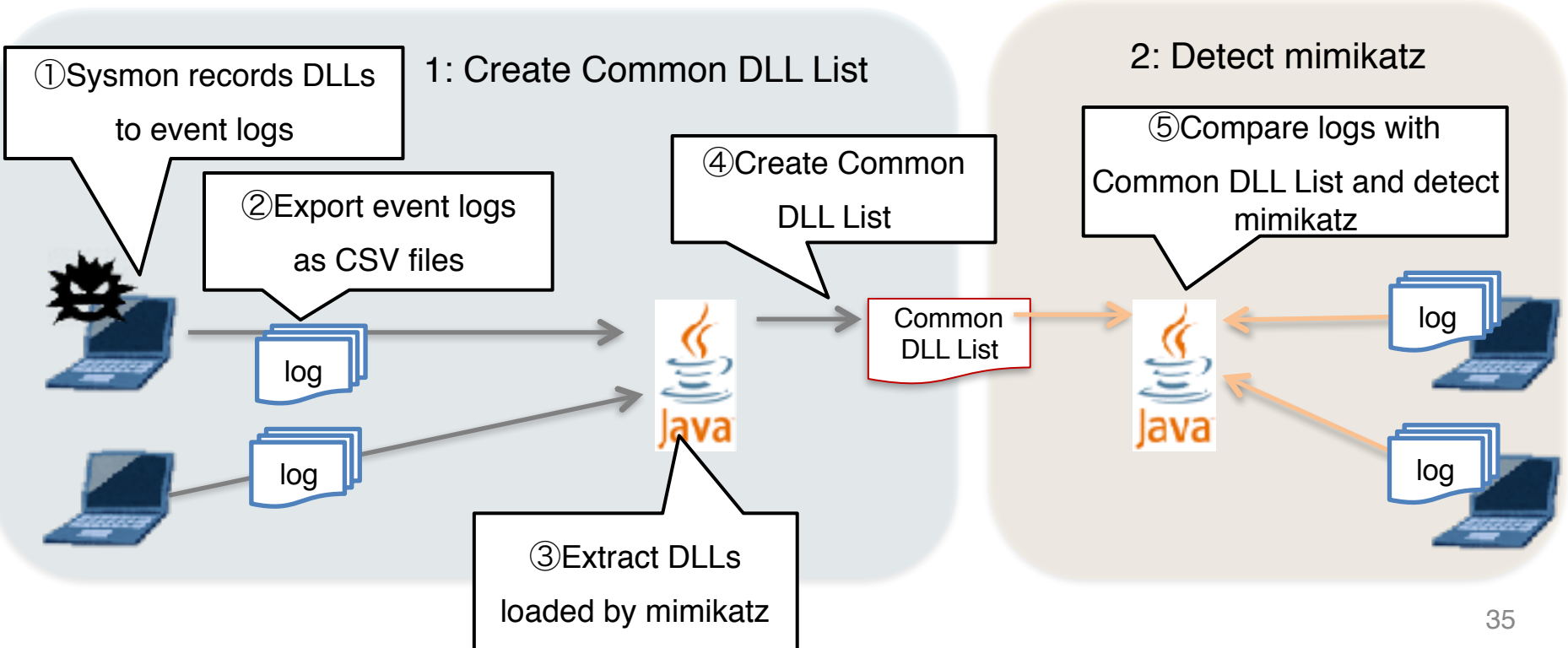
② Export event logs as CSV files

④ Create Common DLL List

③ Extract DLLs loaded by mimikatz

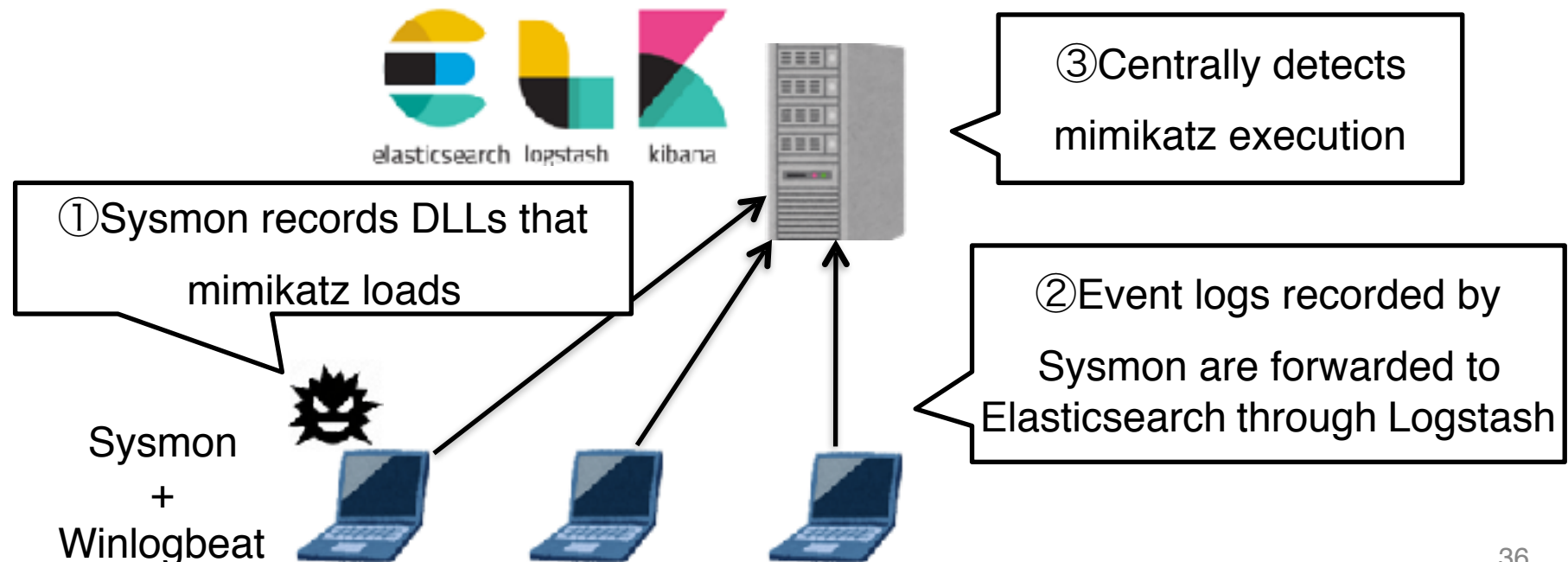
2: Detect mimikatz

⑤ Compare logs with Common DLL List and detect mimikatz

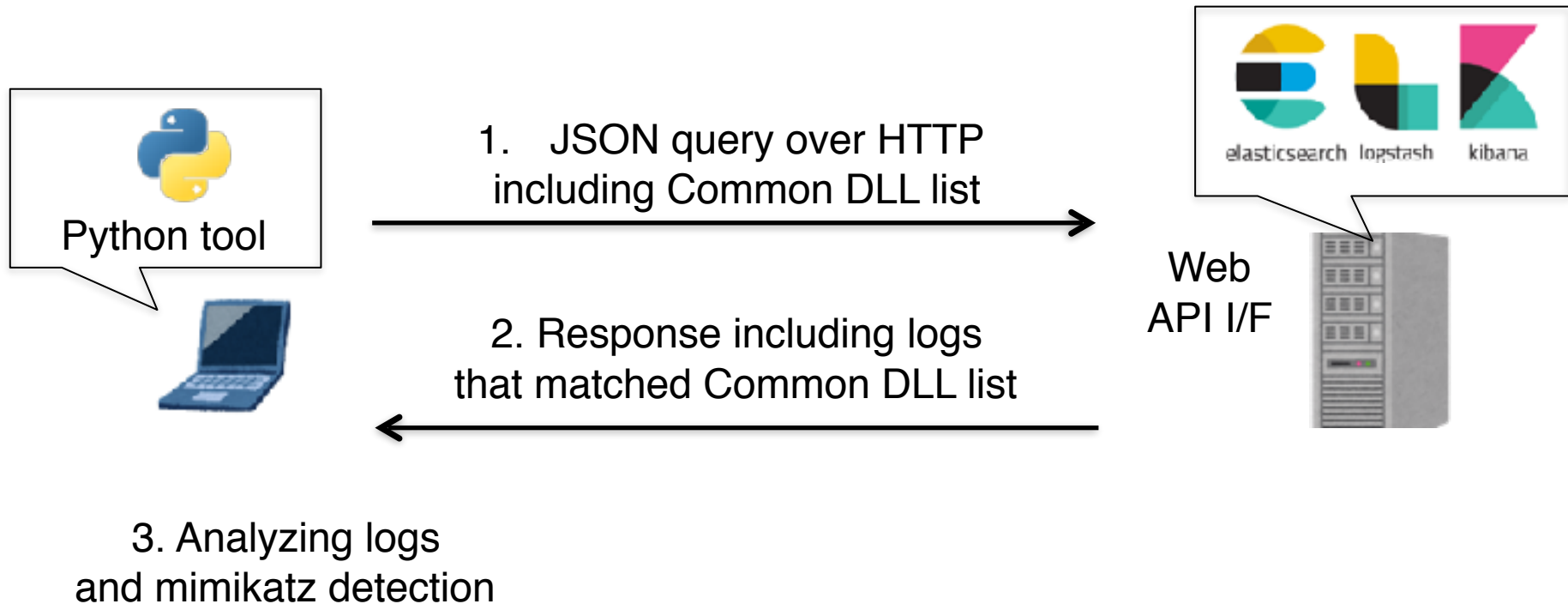


Centralized detection using ELK

- Detection can be even more effective by collecting and analyzing Sysmon logs using ELK (Elasticsearch + Logstash + Kibana)
- If Winlogbeat is installed in computers and servers, logs can be gathered in ELK server and enables centralized analysis
- We released sample programs for detection on github
 - https://github.com/sisoc-tokyo/mimikatz_detection/tree/master/pythonTool/sysmon_detect



Centralized detection using ELK



Reference

(Ref.) Our Works

- Our works are published in the following github repository
- https://github.com/sisoc-tokyo/mimikatz_detection
 - Investigation results of list of DLLs loaded in each environment
 - Tools to create Common DLL List from event logs and detect processes that matches the Common DLL List (Java)
 - A tool to detect processes that matches Common DLL List from Elasticsearch results (Python 3)

(Ref.) Future Works

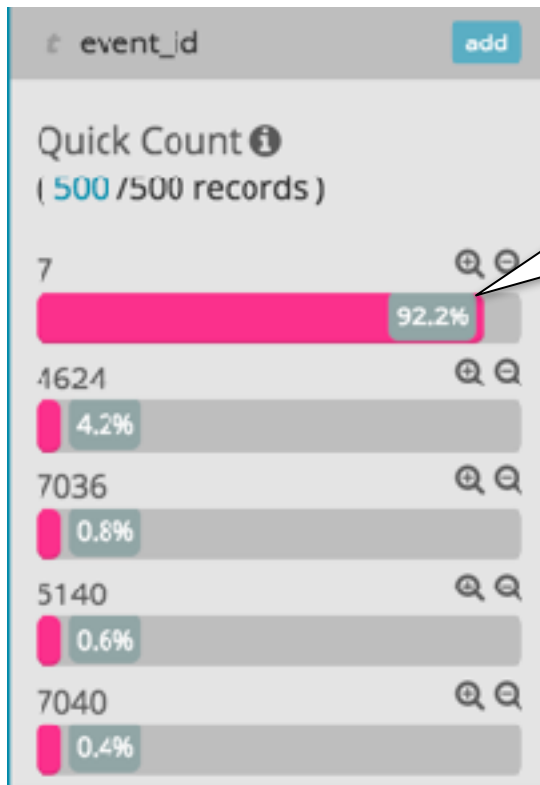
- Agentless log forwarding to log server
 - It is sometimes difficult to install log forward agent in actual environment
 - WMI(Windows Management Instrumentation) is one of solutions for agentless log forwarding
 - WMI is the infrastructure for management data and operations on Windows-based operating systems

(Ref.) Future Works

- Invoke-Mimikatz (under investigation)
 - Invoke-Mimikatz is a PowerShell module which loads mimikatz in memory dynamically using PowerShell to dump credentials without writing on a disk
 - If a process that loads all DLLs in the Common DLL Lists is PowerShell.exe, there is a possibility that Invoke-Mimikatz was executed. So Detailed investigation is required
 - Common DLL list can detect invoke-Mimikatz so far but we should conduct more investigation

(Ref.) Reduce Sysmon log volume

- Since a large amount of logs are recorded in Sysmon logs (Event ID 7:Image loaded), log volume in general is likely to increase
- We can change Sysmon to record specific DLLs loading



Most of event logs are Sysmon log (Event ID 7)

(Ref.) Reduce Sysmon log volume

- Configure Sysmon config file to record only DLLs in the Common DLL List
- Config file needs to be reloaded after the configuration
Sysmon.exe -c Sysmon_config.xml
- For instance, we can limit logging the following DLL loading
 - C:\Windows\System32\advapi32.dll
 - C:\Windows\System32\crypt32.dll

Please write down all DLLs
in the Common DLL List

```
<!-- Event ID 7 == Image Loaded. -->  
<ImageLoad onmatch="include">  
  <ImageLoaded condition="is">C:\Windows\System32\advapi32.dll</ImageLoaded>  
  <ImageLoaded condition="is">C:\Windows\System32\crypt32.dll</ImageLoaded>  
</ImageLoad>
```

- Log amount was reduced from 90% to 20% on our environment

Conclusion

Conclusion

- Early detection of mimikatz execution is effective in preventing further lateral movement in targeted attacks
- DLLs that mimikatz loads are different according to the environment. By detecting DLLs that are commonly loaded (Common DLL List), it is possible to reduce false negative
- Our method proposes comparing loaded DLLs with:
 - 1) Common DLL List to reduce false negative
 - 2) DLL List for each environment to reduce false positive

Thank you for your attention!

Wataru Matsuda: wataru.m@iii.u-tokyo.ac.jp

Mariko Fujimoto: mariko.f@iii.u-tokyo.ac.jp