

Monitoring - Detecting Attacks with MITRE ATT&CK

Michael Schneider

Offense Department, scip AG
misc@scip.ch
<https://www.scip.ch>

Marc Ruef (Editor)

Research Department, scip AG
maru@scip.ch
<https://www.scip.ch>

Abstract: Understanding the techniques used in attacks helps in detecting them. MITRE ATT&CK Enterprise Matrix includes actual attack statistics and techniques. Running it requires configuration of additional Windows audit settings. Correlations between events can be used to identify attacks.

Keywords: Basel, Detect, Exploit, Firewall, Framework, GitHub, Hardening, Microsoft, Mimikatz, Password

1. Preface

This paper was written in 2019 as part of a research project at scip AG, Switzerland. It was initially published online at <https://www.scip.ch/en/?labs.20190711> and is available in English and German. Providing our clients with innovative research for the information technology of the future is an essential part of our company culture.

2. Introduction

A Windows 10 client logs thousands of events each day during normal operation. These events provide information about the state of the system, its activities and its users. Analyzing these logs can provide a lot of information, ranging from the user's work habits based on login/logout operations and device startup/shutdown, to hard disk failures caused by multiple unusual write errors and even attacks on the system.

There are countless techniques that can be used to attack a Windows system. The successful detection of attacks requires an understanding of the technique being used. But it also requires the system to be configured to log the relevant events in the first place. Then from all this data you need to pick out the events that were triggered by the respective technique.

3. MITRE ATT&CK Enterprise Matrix

MITRE ATT&CK Enterprise Matrix [1] is a collection of tactics and techniques based on data taken from actual attacks. *ATT&CK* is publicly available and can be used free of charge. The April 2019 version of *ATT&CK Enterprise Matrix* includes 12 tactics and 244 techniques. For example, the *credential access* [2] tactic involves techniques for gaining access to login credentials, including *brute force* [3], *credential dumping* [4] and *forced authentication* [5]. Each technique includes a description, a list of *examples* detailing how the technique is used along with possible countermeasures that are divided into categories of *mitigation* and *detection*.

The *ATT&CK Enterprise Matrix* is, among other things, a good foundation for identifying attacks. The recommended measures relate specifically to the technique in question. For example, in the case of *Kerberoasting* [6] the recommended procedure is to monitor the Kerberos event log for *Kerberos Service Ticket Operations* on accounts that issue a large number of requests within the span of just a few minutes, particularly if the ticket request contains the encryption algorithm *RC4* (type 0x17). The recommendations for dealing with the technique of *credential dumping* are more general. Recommendations include checking to make sure *LSASS* is started as a protected process and that attempts to access the *LSASS* process are monitored. The running of applications and scripts should be monitored for suspicious behavior as well. One of the best-known examples of credential dumping is the tool known as *Mimikatz* [7], which is not only used by security researchers but also by *many groups who use it to launch real attacks* [8]. This is why we are currently configuring a test system that uses built-in Windows features to detect *Mimikatz* attacks.

4. Configuring Audit Settings

The article covering *Windows 10 client hardening* [9] already discussed several additional settings for the audit configuration. The latest version of the *hardening checklist* [10] includes additional settings and has been updated for *Windows 10 Build 1903*. To detect when *Mimikatz* is being run requires some additional audit settings in Windows. These include some of the following settings which, incidentally, can also be used to detect other attack techniques:

- Configuring the audit policy with the *Windows security baseline* [11]
- Configuring the Advanced Audit Policy with the *Windows security baseline* [12]
- Enabling audit settings for incoming and outgoing NTLM traffic
- Enabling audit settings for rules controlling *attack surface reduction* [13]
- Enabling log settings for Windows Firewall

- Enabling use of Credential Guard (not supported by all systems)
- Enabling *LSA Audit Mode* [14] and configuring it as a protected process
- Enabling PowerShell monitoring
- Installing Sysmon using the configuration of *SwiftOnSecurity* [15] as a template

4.1. Monitoring Mimikatz

The Windows client runs the LSASS process in protected mode, which can be verified by checking the system event log when starting the operating system. The event with the source *Wininit* and event ID 12 is: *LSASS.exe was started as a protected process with level: 4*. If Mimikatz now attempts to read the login credentials of users who are logged in, the operation will fail:

```
mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::logonpasswords
ERROR kuhl_m_sekurlsa_acquireLSA ; Handle on
memory (0x00000005)
```

Sysmon can be used to detect when *Mimikatz* starts, for example, by monitoring the *Process Create* event. If a modified version of *Mimikatz* is used and the name, description and other properties of the application have been concealed, this event is not necessarily reliable. When starting *Mimikatz*, the *Sensitive Privilege Use* task with event ID 4673 will also appear in the security event log as *Failed*. An attempt will be made to acquire *SeTcbPrivilege* privileges. If the process ID has the same ID as the *Sysmon* event, this is a red flag for suspicious activity.

In the next step, the *Mimikatz driver* is loaded to disable *Protected Mode* for the LSASS process in the system memory.

```
mimikatz # !+
[*] 'mimidrv' service not present
[+] 'mimidrv' service successfully registered
[+] 'mimidrv' service ACL to everyone
[+] 'mimidrv' service started

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # !processprotect /remove
/process:lsass.exe
Process : lsass.exe
PID 652 -> 00/00 [0-0-0]
```

This activity can be detected as event *Driver Loaded* in the *Sysmon* log. In addition, an entry for loading of the driver will also appear in the system event log with the *Service Control Manager* as the source and event ID 7045. The installation of a driver is also documented in the security event log and will have event ID 4697.

Once the LSASS process is no longer protected, the user's login credentials can be read from the process memory.

```
mimikatz # sekurlsa::logonpasswords

Authentication Id : 0 ; 2496472
(00000000:002617d8)
Session           : Interactive from 1
User Name          : admin
Domain             : W10
```

```
Logon Server       : W10
Logon Time         : 31/02/2019 08:10:41
SID                : S-1-5-21-2094461431-
2471580941-81854671-1002
...
```

Reading the login credentials generates the event *An attempt was made to access an object* with the category *Kernel Object* and event ID 4663 in the security event log. This means that an application has successfully accessed the LSASS process object.

Expanding the *Sysmon* configuration to include the following setting can make it easier to detect attacks on the LSASS process:

```
<!--SYSMON EVENT ID 10 : INTER-PROCESS
ACCESS-->
<!--DATA: UtcTime, SourceProcessGuid,
SourceProcessId, SourceThreadId, SourceImage,
TargetProcessGuid, TargetProcessId,
TargetImage, GrantedAccess, CallTrace-->
<ProcessAccess onmatch="include">
  <TargetImage
condition="contains">lsass.exe</TargetImage>
</ProcessAccess>
```

Analyzing how *Mimikatz* is executed and applying these insights makes it possible to detect the attack. In the simplest scenario, you can monitor when *mimikatz.exe* is run and when the *mimidrv.sys* driver is loaded. By correlating the *Process Create* and *Driver Loaded* events as well as attempts to access the LSASS process, you can also detect a disguised version of *Mimikatz* or an application that uses similar mechanisms.

5. Moving Forward

Detecting *Mimikatz* is only one first step towards being able to detect all of the attack techniques covered in the *ATT&CK Enterprise Matrix*. The documented techniques do, however, provide assistance and are therefore a good place to start building a detection framework. Another helpful tool is Andrea Fortuna's *collection of event log IDs* [16]. In addition, the mechanisms for detecting suspicious activities can usually be used for more than just a single technique, so you don't have to set up a separate detection rule for each individual technique. It should also be noted that other attacks and activities that are beyond the scope of *MITRE ATT&CK* should be taken into account as well.

You can identify attacks on the client system by integrating an alarm into the central infrastructure or by using a central monitoring system to carry out the necessary analytics. In centralized analytics systems, you can also incorporate data from other systems to provide additional clues. The history from the previous few days should also be taken into account, because an attack can span several days or even weeks.

6. External Links

- [1] <https://attack.mitre.org/>
- [2] <https://attack.mitre.org/tactics/TA0006/>
- [3] <https://attack.mitre.org/techniques/T1110/>
- [4] <https://attack.mitre.org/techniques/T1003/>
- [5] <https://attack.mitre.org/techniques/T1187/>
- [6] <https://attack.mitre.org/techniques/T1208/>

- [7] <https://github.com/gentilkiwi/mimikatz/>
- [8] <https://attack.mitre.org/software/S0002/>
- [9] <https://www.scip.ch/en/?labs.20161215>
- [10] https://github.com/0x6d69636b/windows_hardening/
- [11] <https://blogs.technet.microsoft.com/secguide/2019/05/23/security-baseline-final-for-windows-10-v1903-and-windows-server-v1903/>
- [12] <https://blogs.technet.microsoft.com/secguide/2019/05/23/security-baseline-final-for-windows-10-v1903-and-windows-server-v1903/>
- [13] <https://docs.microsoft.com/en-us/windows/security/threat-protection/windows-defender-exploit-guard/attack-surface-reduction-exploit-guard>
- [14] <https://docs.microsoft.com/en-us/windows-server/security/credentials-protection-and-management/>
- [15] <https://github.com/SwiftOnSecurity/sysmon-config>
- [16] <https://www.andreafortuna.org/2019/06/12/windows-security-event-logs-my-own-cheatsheet/>