

Detection Engineering with SIGMA

Defend against APT targeting Japan

Rintaro Koike
Shota Nakajima

\$ whoami

■ 小池 優太郎 (NTTセキュリティ・ジャパン株式会社)

- 脅威リサーチやマルウェア解析業務に従事
- nao_secという謎の同人サークルの主宰
- 猫とコーヒーが好き



■ 中島 将太 (株式会社サイバーディフェンス研究所)

- 脅威リサーチやマルウェア解析業務に従事
- 昼間はただのセキュリティエンジニア
- 最近はソーシャルエンジニアリング技術を勉強中



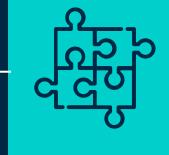
TABLE OF CONTENTS



01

Introduction

Workshopの背景と
大まかな説明をし
ます



02

Sigma writing

Sigmaの仕様やツ
ールについて紹介
します



03

Hands-on

ハンズオン形式で
Sigmaルールの作
成方法を学びます



04

Exercise

イベントログを解
析して、Sigmaル
ールを作成します

Motivation & Goal

トラフィックやファイルではなく、振る舞いを検出したい

→日本を標的としたAPTから組織を守るために検知ロジックを作成

- オープンな脅威情報を読み解く
- 有効な検知ロジックの組み立て
- 誤検知を最小限に

Target

- セキュリティに関わるログファイルを扱う人
 - インシデントレスポンダー
 - SOCアナリスト
 - フォレンジスト
- マルウェアや侵害のシグネチャを作成する人
 - マルウェアアナリスト
 - 脅威アナリスト

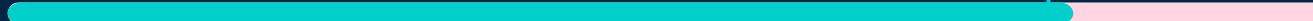
Announcement

講義で使用するファイルについて

- 指定した環境以外では取り扱わないこと
 - マルウェア本体は含んでいないが念のため...
- ワークショップ終了後は削除すること
- 第三者および外部Webサービス（VirusTotalなど）に提供・アップロードしないこと

Introduction

01

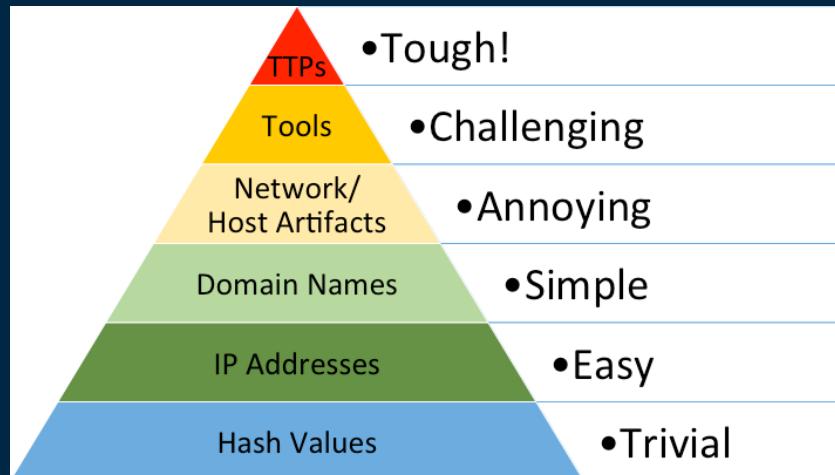


Threat Detection

- 実世界での現実的なセキュリティ対策
 - WAF、IDS/IPS、アンチウイルス、EDR...
 - 検知ロジック
 - 製品固有の記述言語・形式
 - Snort/Suricata
 - Yara
 - 一般的な組織で記述される検知ロジックの要素
 - ファイルのハッシュ値
 - ドメイン
 - IPアドレス
- 攻撃者にとって変更が容易

Pyramid of Pain

- IoCについて、攻撃者の痛みの度合いを示した図
- <http://detect-respond.blogspot.com/2013/03/the-pyramid-of-pain.html>



Sigma

様々なログに対して適用可能な汎用的なシグネチャ記述フォーマット

- シグネチャ（ルール）を定義し、合致するイベントを検出
- 様々な製品のクエリに変換することが可能

“**Sigma** is for log files what **Snort** is for
network traffic and **YARA** is for files”

Sigma for InfoSec

具体的にどのような場面で使用するのか？

■ Analysis

- 類似挙動を見つけてファミリを特定する
- 侵害で使われている手法（権限昇格、DLL Side-Loadingなど）を特定する

■ Detection

- IR中に見つけた侵害挙動と類似するイベントを検索する
- 特定の通信先やファイルパスへアクセスするイベントを検索する

Sigma in Service

■ VirusTotal

- サンドボックスのログに対してSigmaルールが適用される
- <https://support.virustotal.com/hc/en-us/articles/360015738658-Sigma-rules>

The screenshot shows the VirusTotal interface with the 'DETECTION' tab selected. It displays a section titled 'Crowdsourced Sigma Rules' with a count of 12. Below this, there are four categories: CRITICAL 1, HIGH 2, MEDIUM 4, and LOW 5. A warning icon indicates 1 match for rule APT 37, which is described as targeting South Korea, Japan, Vietnam, and the Middle East across various industry verticals.

DETECTION DETAILS RELATIONS BEHAVIOR CONTENT TELEMETRY COMMUNITY 12

Crowdsourced Sigma Rules ⓘ

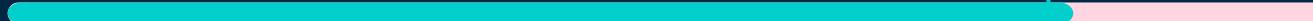
CRITICAL 1 HIGH 2 MEDIUM 4 LOW 5

1 match for rule **APT 37** by Ariel Millahuel from SOC Prime Threat Detection Marketplace
↳ *APT37 has likely been active since at least 2012 and focuses on targeting the public and private sectors primarily in South Korea. In 2017, APT37 expanded its targeting beyond the Korean peninsula to include Japan, Vietnam and the Middle East, and to a wider range of industry verticals, including chemicals, electronics, manufacturing, aerospace, automotive and healthcare entities*



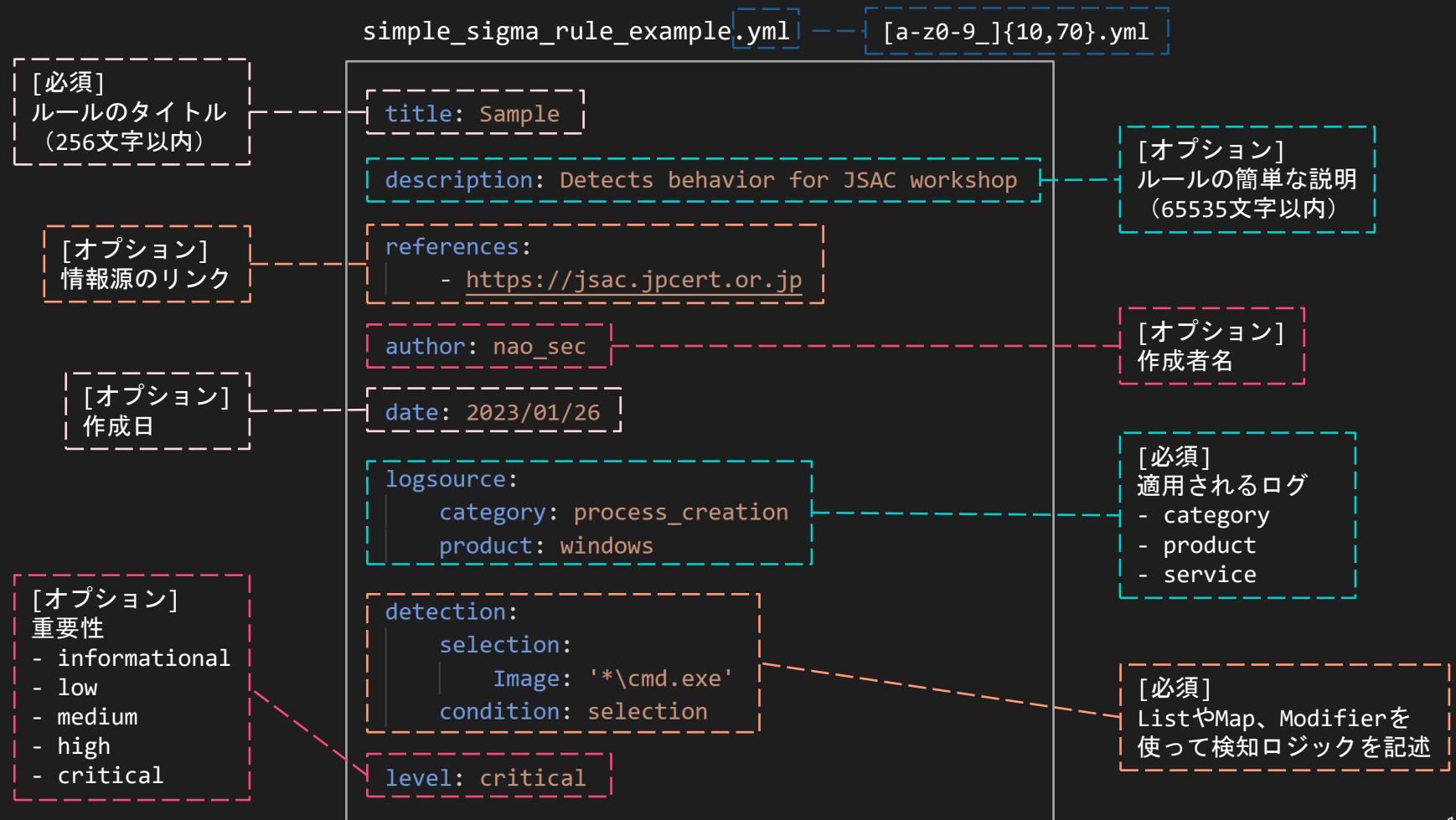
Sigma writing

02



How to write Sigma Rule





Log-source section

検知ルールを適用するログデータを示す

- category
 - ログのカテゴリ (e.g. firewall, web, antivirus, process_creation)
- product
 - 製品名 (e.g. windows, linux, apache)
- service
 - サービス名 (e.g. sshd, applocker)

Detection section

検知ロジックを記述するセクション

- 要素と条件で構成される
- ListとMapが使用可能
- 小文字と大文字は区別されない
 - 正規表現では区別される

Wildcard

- ?: 単一文字列として置き換えられる
 - cm?.exe -> cmd.exe
- *: 無制限の長さの文字列として置き換えられる
 - power*ll.exe -> powershell.exe

Escape

- 特殊な用途の文字を通常の文字列として扱いたい場合は直前にバックスラッシュ¥を入れる
 - ワイルドカード文字である?と*はエスケープが必要
 - ただの¥を使用したい場合は¥¥とする

List

- ハイフンから始まる
- 文字列あるいはマップで記述された要素のリスト
- リストはORで評価される

```
detection:  
  keywords:  
    - EVILSERVICE  
    - svchost.exe -n evil
```

“EVILSERVICE” か “svchost.exe -n evil”
という文字列を含むか

```
detection:  
  selection:  
    - Image|endswith: \\example.exe  
    - Description|contains: Test executable
```

Imageが “\\example.exe” で終わるか
Descriptionに “Test executable” を含むか

Map

- キーと値で構成される
 - キーはログデータのフィールド
 - 値は文字列か整数値
- マップはANDで評価される

```
detection:  
    selection:  
        EventLog: Security  
        EventID:  
            - 517  
            - 1102  
    condition: selection
```

EventLogフィールドの値が “Security” かつ
EventIDが 517 か 1102

Modifier

修飾子	説明
contains	フィールド内のどこかと一致する(値をワイルドカード*で囲う)
all	値のリストの論理演算子をORからANDに変更する
base64	Base64エンコードする
base64offset	Base64エンコードする(全てのバリエーションを含む)
endswith	値でフィールドが終了する
startswith	値でフィールドが開始する
utf16le	UTF16-LEエンコードする
utf16be	UTF16-BEエンコードする
wide	utf16leと同一
utf16	UTF16エンコードする(Byte order markを先頭に追加)
windash	"-"を"/"に置き換えたものも許容する

base64offset

- Base64エンコードは前後の文字列によって結果が変わる

Hello -> SGVsbG8=

AHello -> QUhlbGxv

HelloA -> SGVsbG9B

AHelloA -> QUhlbGxvQQ==

- これら全てでマッチするように静的な部分のみを検索対象とする

windash

- “-”を”/”に置き換えたものも許容する修飾子
- 主にWindowsでのコマンドラインオプションで使用される
- <https://github.com/SigmaHQ/sigma/issues/3749>

```
certutil.exe /urlcache /split /f http://7-zip.org/a/7z1604-x64.exe 7zip.exe  
certutil.exe -urlcache -split -f http://7-zip.org/a/7z1604-x64.exe 7zip.exe
```

Condition

- Detectionの条件を記述する
- 様々な記述方法がある
 - AND
 - OR
 - NOT
 - all of them
 - N of them (Nは数字)
 - ()

```
selection1 and selection2
selection1 or selection2
not selection
all of selection*
1 of selection*
selection1 and (selection2a or selection2b)
```

Sigma Rule Example

- https://github.com/SigmaHQ/sigma/blob/master/rules/windows/process_creation/proc_creation_win_apt_taidoor.yml

```
detection:  
    selection1:  
        CommandLine|contains:  
            - 'dll,MyStart'  
            - 'dll MyStart'  
    selection2a:  
        CommandLine|endswith: ' MyStart'  
    selection2b:  
        CommandLine|contains: 'rundll32.exe'  
    condition: selection1 or ( selection2a and selection2b )
```

コマンドラインに "dll,MyStart" あるいは
"dll MyStart" が含まれているか

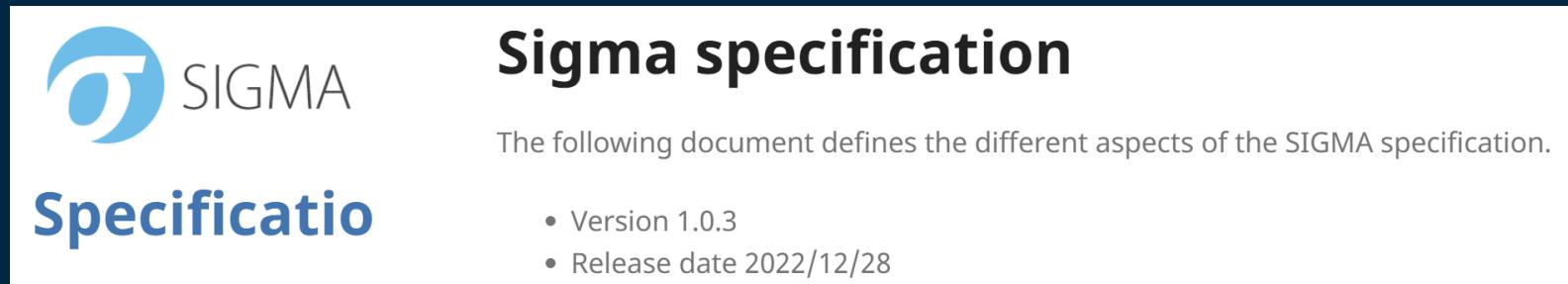
あるいは

コマンドラインが " MyStart" で終わり
かつ "rundll32.exe" が含まれているか



Detailed specifications

- 細かい仕様は公式ドキュメントで確認する
 - https://sigmahq.github.io/sigma-specification/Sigma_specification.html



The screenshot shows a white rectangular area on a dark blue background. In the top-left corner, there is a logo consisting of a blue stylized 'T' icon followed by the word 'SIGMA' in a light gray sans-serif font. Below the logo, the word 'Specification' is written in a large, bold, blue sans-serif font. To the right of the logo, the title 'Sigma specification' is displayed in a large, bold, black sans-serif font. Underneath the title, a subtitle reads 'The following document defines the different aspects of the SIGMA specification.' Below the subtitle, two bullet points are listed: 'Version 1.0.3' and 'Release date 2022/12/28'. The rest of the page is blank white space.

For more examples

- 公開されているSigmaルールリポジトリの例
 - <https://github.com/SigmaHQ/sigma/tree/master/rules>
 - <https://github.com/The-DFIR-Report/Sigma-Rules>
 - <https://github.com/joesecurity/sigma-rules>
 - <https://github.com/mbabinski/Sigma-Rules>
 - <https://github.com/mdecrevoisier/SIGMA-detection-rules>
 - <https://github.com/blacklanternsecurity/sigma-rules>
 - <https://github.com/WithSecureLabs/lazarus-sigma-rules>

Test Asset

■ 作成した検知ロジックのテスト

- <https://github.com/SigmaHQ/sigma/blob/master/.github/workflows/sigma-test.yml>
- https://github.com/SigmaHQ/sigma/blob/master/tests/test_rules.py

The screenshot shows the GitHub Actions logs for a workflow named 'test-sigma'. The workflow completed successfully yesterday in 4m 52s. The log details the following steps:

- > Set up job (2s)
- > Run actions/checkout@v3.2.0 (7s)
- > Set up Python 3.11 (0s)
- > Install dependencies (5s)
- > Test Sigma Rule Syntax (11s)
- > Test Sigma Rules (4m 24s)
- > Post Set up Python 3.11 (0s)
- > Post Run actions/checkout@v3.2.0 (0s)
- > Complete job (0s)

Tools for Sigma

Sigmaルールを処理することができる有名なツール

- Chainsaw
- Zircolite
- Hayabusa
- Uncoder
- VSCode Extension



Chainsaw

- <https://github.com/WithSecureLabs/chainsaw>
- イベントログなどにSigmaルールを適用可能なツール
- Rust製でWindows/Linux/Mac用のバイナリが用意されている
- 操作も簡潔で使いやすい

Zircolite

- <https://github.com/wagga40/Zircolite>
- イベントログなどにSigmaルールを適用可能なツール
- Python製、WindowsやLinux向けのバイナリも配布している
- デフォルトでSysmon for Linuxのログが捌ける
- SigmaルールはJSONに変換しないといけない

Hayabusa

- <https://github.com/Yamato-Security/hayabusa>
- イベントログなどにSigmaルールを適用可能なツール
- Rust製でWindows向けのバイナリを配布している
- Hayabusa専用のルール形式を使う
 - SigmaルールをHayabusaルール形式に変換可能
- 日本人が開発に関わっており、日本語の公式ドキュメントがある

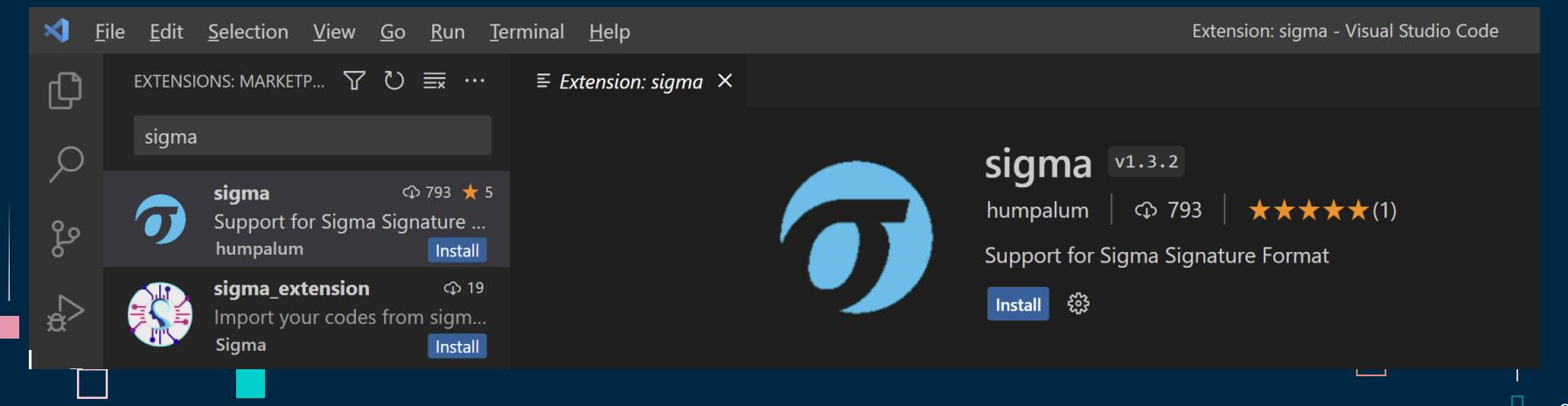
Uncoder

- <https://uncoder.io/>
- SOC Primeが提供しているWebサービス
- Web上でSigmaルールを様々な製品用に変換できる

The screenshot shows the Uncoder web application interface. At the top, there's a search bar with the text "sigma: Empire PowerShell Launch Parameters". Below the search bar, there are several tabs: "Sigma" (which is selected), "Elast...", "QRa...", "Splunk", "Splu...", "Microsoft...", and "Translate". The "Translate" tab has a dropdown menu set to "Microsoft Defender for Endpoint". The main area contains two code snippets. The left snippet is a Sigma rule:1 title: Empire PowerShell Launch Parameters
2 description: Detects suspicious powershell command line parameters
 used in Empire
3 status: experimental
4 references:
5 - https://github.com/EmpireProject/Empire/blob
 /c2ba61ca8d2031dad0cf1cd5770ba723e8b710db/lib/common/helpers
 .py#L165
6 - https://github.com/EmpireProject/Empire/blob
 /e37fb2eebf8ff8f5a0a689f1589f424906fe13055/lib/modules/powershell
 /persistence/powerbreach/deaduser.py#L191The right snippet is the converted Microsoft Defender for Endpoint rule:DeviceProcessEvents | where ProcessCommandLine in~ (@'* -NoP -sta -NonI -W Hidden -Enc *', @'* -noP -sta -w 1 -enc *)At the bottom right, there are buttons for "Suggest translation" and "Copy". A status message says "Translating to: Microsoft Defender for Endpoint".

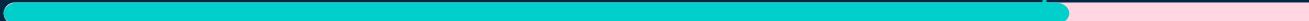
VSCode Extension

- <https://marketplace.visualstudio.com/items?itemName=humpalum.sigma>
- VSCode上でSigmaルールを記述する際に便利な拡張



Hands-on

03



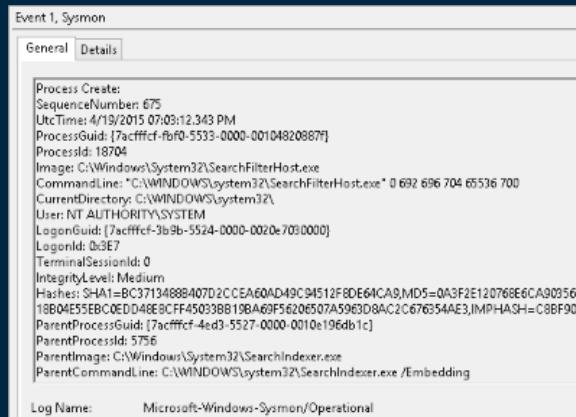
Logging Environment

ExerciseやHands-onで使用するログ

- 仮想環境にSysmonをインストール
 - <https://github.com/Neo23x0/sysmon-config/blob/master/sysmonconfig-trace.xml>
- インターネット接続有
 - 名前解決はhostsファイルを使ったものもあり、必ずしもDNSが使われたわけではない

Sysmon

- Microsoftが提供するシステムのアクティビティを監視してログインするツール
 - ファイル、プロセス、レジストリ、ネットワークの操作など様々なイベントをイベントログに記録する
 - 無償で簡易なEDR製品のように利用することが可能



Sysmon for Linux

- <https://github.com/Sysinternals/SysmonForLinux>
- Linux向けのSysmon
 - Windows版と（大体）同じようにロギングできる
 - 主要なディストリビューションであればインストールは容易

```
<Event>
  <System>
    <Provider Name="Linux-Sysmon" Guid="{ff032593-a8d3-4f13-b0d6-01fc615a0f97}" />
    <EventID>1</EventID>
    <Version>5</Version>
    <Level>4</Level>
    <Task>1</Task>
    <Opcode>0</Opcode>
    <Keywords>0x8000000000000000</Keywords>
    <TimeCreated SystemTime="2023-01-02T06:29:23.620103000Z" />
    <EventRecordID>1502</EventRecordID>
    <Correlation />
    <Execution ProcessID="1130" ThreadID="1130" />
    <Channel>Linux-Sysmon/Operational</Channel>
    <Computer>ubuntu</Computer>
    <Security UserId="0" />
  </System>
  <EventData>
    <Data Name="RuleName">-</Data>
    <Data Name="UtcTime">2023-01-02 06:29:43.012</Data>
    <Data Name="ProcessGuid">{55f63222-79d7-63b2-a584-6b0000000000}</Data>
    <Data Name="ProcessId">2283</Data>
    <Data Name="Image">/usr/bin/python3.8</Data>
  </EventData>

```

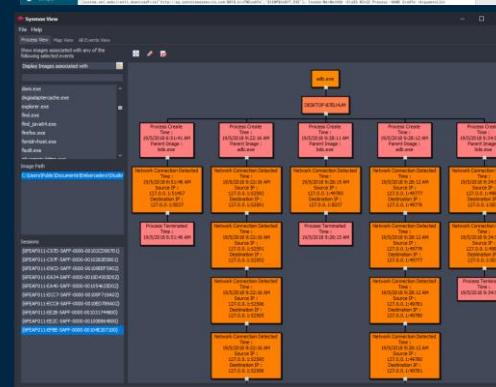
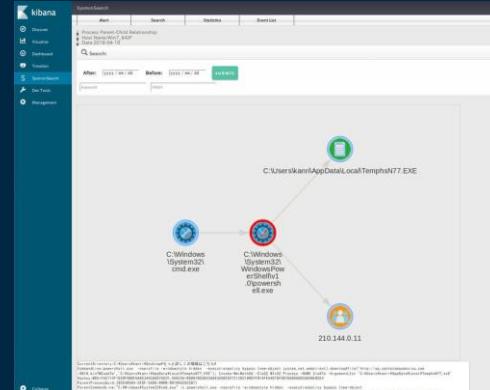
Utilities for Sysmon

■ SysmonSearch

- JPCERT/CCがログを可視化して
端末の不審な挙動を調査するために開発

■ Sysmon View

- Sysmonログを可視化するツール



Sysmon Event ID (1/3)

イベントID	操作	詳細
1	プロセスの作成	プロセスが別のプロセスを作成したときに記録される
2	プロセスのファイルの作成時間の変更	ファイル作成時刻がプロセスによって変更されたときに記録される
3	ネットワーク接続	TCP/UDP接続を記録する
4	Sysmonサービスの状態変更	Sysmonサービスの状態が変更されたときに記録される
5	プロセスの終了	プロセスが終了したときに記録される
6	ドライバーのロード	ドライバがロードされたときに記録される
7	イメージのロード	DLLなどがプロセスにロードされたときに記録される
8	リモートプロセスの作成	プロセスが別のプロセスでスレッドを作成
9	RawAccessRead	\$_\$_\$_をつかったプロセスの読み取り処理を記録する
10	プロセスアクセス	プロセスが別のプロセスを開いたときに記録される

Sysmon Event ID (2/3)

イベントID	操作	詳細
11	ファイルの作成	プロセスがファイルの作成または上書きされたときに記録される
12	Registryイベント(作成と削除)	レジストリのキーと値が作成または削除されたときに記録される
13	Registryイベント(値のセット)	レジストリの値がセットされたときに記録される
14	Registryイベント(名前変更)	レジストリのキーと値の名前が変更されたときに記録される
15	FileCreateStreamHash	名前付きファイルストリームが作成されるときに記録される
16	サービスのコンフィグの変更	Sysmonのコンフィグファイルが変更されたときに記録される
17	Pipeイベント(作成)	名前付きパイプが作成されたときに記録されます
18	Pipeイベント(接続)	名前付きパイプの接続が確率されたときに記録される
19	WmiEventFilterアクティビティ	WMIのEventFilterを記録する
20	WmiEventConsumerアクティビティ	WMIのEvent Consumerを記録する

Sysmon Event ID (3/3)

イベントID	操作	詳細
21	WmiEventConsumerToFilter	Event FilterとEvent Consumerのバインドを記録する
22	DNSイベント	DNSクエリの実行時に記録される
23	アーカイブされたファイルの削除	Sysmonがアーカイブしたファイルの削除時に記録される
24	クリップボードの変更	クリップボードの内容が変更されたときに記録される
25	プロセス改ざん	HollowingやHerpaderpingを検知すると記録される
26	ファイル削除	ファイルの削除時に記録される
27	FileBlockExecutable	Sysmonが実行ファイルを検知してブロックすると記録される
28	FileBlockShredding	Sdeleteなどでファイルをshreddingすると記録される
255	エラー	Sysmonのエラーログが記録される

Sysmon Event Field

Sysmonの各イベントにはFieldが存在し、SigmaのMapで指定できる

- <https://github.com/olafhartong/sysmon-cheatsheet/blob/master/Sysmon-Cheatsheet.pdf>

EventID 1 Process Create	
UtcTime	Time in UTC when event was created
ProcessGuid	Process Guid of the process that got spawned/created (child)
ProcessId	Process ID used by the OS to identify the created process (child)
Image	File path of the process being spawned/created. Considered also the child or source process
FileVersion	Version of the image associated with the main process (child)
Description	Description of the image associated with the main process (child)
Product	Product name the image associated with the main process (child) belongs to
OriginalFileName	OriginalFileName from the PE header, added on compilation
Company	Company name the image associated with the main process (child) belongs to
CommandLine	Arguments which were passed to the executable associated with the main process
CurrentDirectory	The path without the name of the image associated with the process
User	Name of the account that created the process (child) . It usually contains domain name and username
LogonGuid	Logon GUID of the user who created the new process. Value that can help you correlate this event with others that contain the same Logon GUID
LogonId	Login ID of the user who created the new process. Value that can help you correlate this event with others that contain the same Logon ID
TerminalSessionId	ID of the session the user belongs to
IntegrityLevel	Integrity label assigned to a process
Hashes	Full hash of the file with the algorithms in the HashType field
ParentProcessGuid	ProcessGUID of the process that spawned/created the main process (child)
ParentProcessId	Process ID of the process that spawned/created the main process (child)
ParentImage	File path that spawned/created the main process
ParentCommandLine	Arguments which were passed to the executable associated with the parent process
ParentUser	Name of the account that created the parent process. It usually contains domain name and username

EVTX Conversion

EVTXファイルは扱いにくいのでEvtxECmdで変換

- <https://ericzimmerman.github.io/#!index.md>

CSVに変換する場合

```
$ EvtxECmd.exe -f blacktech.evtx --csv ./
```

JSONに変換する場合

```
$ EvtxECmd.exe -f blacktech.evtx --json ./
```

Timeline Explorer

CSVに変換したイベントログを見やすく表示

- <https://ericzimmerman.github.io/#!index.md>

The screenshot shows the Timeline Explorer application window. The title bar reads "Timeline Explorer v1.3.0.0". The menu bar includes File, Tools, Tabs, View, and Help. A tab labeled "20230114153906_EvtxCmd_Output.csv" is selected. The main pane displays a table of event log entries. The columns are: L, Tag, Re..., Event..., Time Created, Event..., Level, Provider, Chann..., Process Id, Computer, User Id, Map Description, User Na..., Remote Host, Payload Data. The data shows numerous events from Microsoft Edge (Micro...), mostly FileCreate operations, occurring on 2022-11-28 at 01:00:39. The process ID is consistently 3140. The user ID is often 110 or 111. The payload data includes ProcessID values like 6580, 6580, 6580, etc. The bottom of the window shows a search bar with "Event Id = 11 And User Name Contains IEUser" and a status bar indicating "Total lines 50,006 Visible lines 0 Open files: 1 Search options".

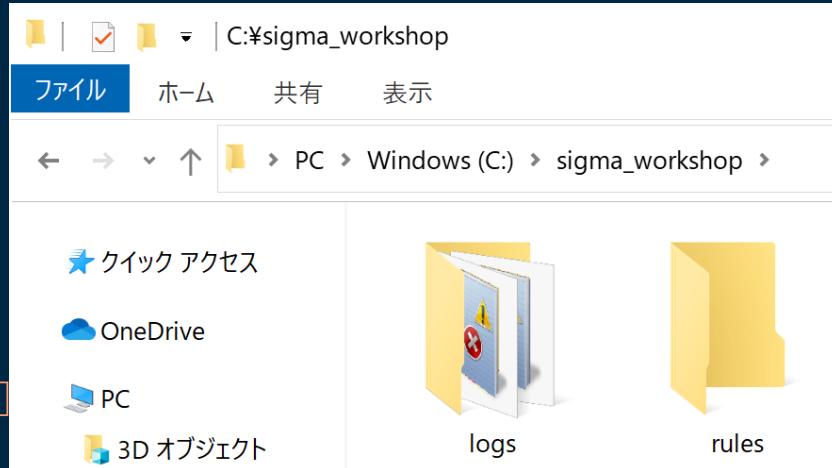
L	Tag	Re...	Event...	Time Created	Event...	Level	Provider	Chann...	Process Id	Computer	User Id	Map Description	User Na...	Remote Host	Payload Data
110		110	25865	2022-11-28 01:00:39	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 6580,	
111		111	25866	2022-11-28 01:00:39	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 6580,	
112		112	25867	2022-11-28 01:00:39	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 6580,	
113		113	25868	2022-11-28 01:00:39	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 6580,	
160		160	25915	2022-11-28 01:00:40	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 6580,	
161		161	25916	2022-11-28 01:00:40	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 6580,	
175		175	25930	2022-11-28 01:00:40	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 6580,	
176		176	25931	2022-11-28 01:00:40	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 6580,	
177		177	25932	2022-11-28 01:00:40	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 6580,	
180		180	25935	2022-11-28 01:00:40	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 6580,	
367		367	26122	2022-11-28 01:00:42	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 7700,	
542		542	26297	2022-11-28 01:00:45	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 7328,	
616		616	26371	2022-11-28 01:00:46	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 7104,	
664		664	26419	2022-11-28 01:00:47	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 5220,	
668		668	26423	2022-11-28 01:00:47	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 7328,	
674		674	26429	2022-11-28 01:00:47	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 7328,	
679		679	26434	2022-11-28 01:00:47	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 7328,	
680		680	26435	2022-11-28 01:00:47	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 7328,	
1066		10...	26821	2022-11-28 01:00:50	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 7316,	
1076		10...	26831	2022-11-28 01:00:50	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 7316,	
1092		10...	26847	2022-11-28 01:00:50	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 7316,	
1093		10...	26848	2022-11-28 01:00:50	11	Info	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 7316,	
1096		10...	26850	2022-11-28 01:00:50	11	Tra...	Microso...	Micro...	3140	MSEDGEW...	S-1-5...	FileCreate	MSEDGEWI...	ProcessID: 7316,	

Environment Setup

Setup

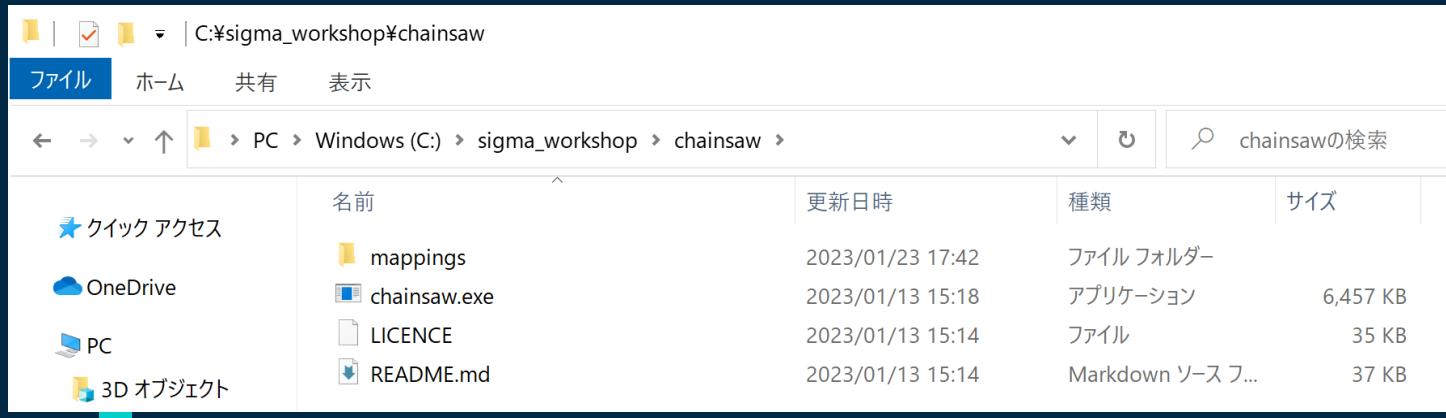
1. SlackからZIPファイルをダウンロード
2. 当日に配布したパスワードで下記パスへZIPの中身を展開

C:¥sigma_workshop



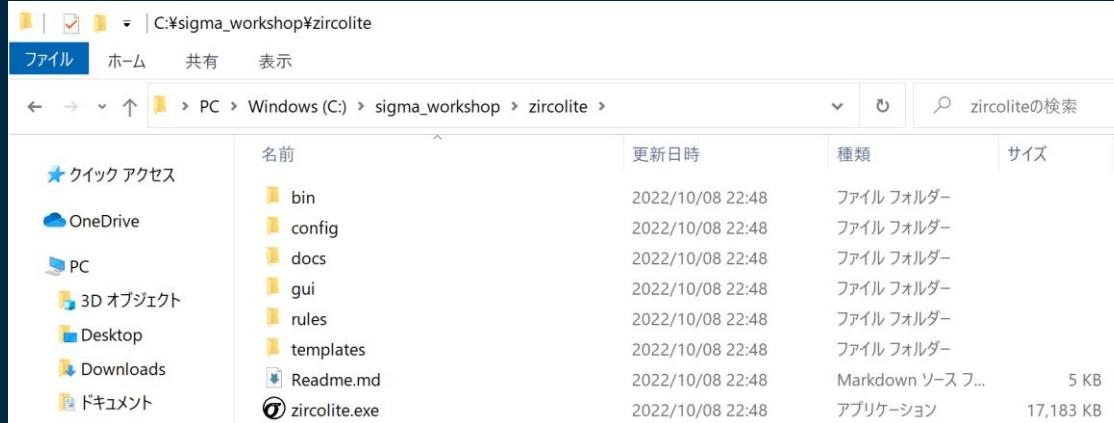
Chainsaw

1. 下記から *chainsaw_x86_64-pc-windows-msvc.zip* をダウンロード
 - <https://github.com/WithSecureLabs/chainsaw/releases/tag/v2.3.1>
2. ZIPを展開して出てきた*chainsaw*ディレクトリを
C:\\$sigma_workshop\\$chainsaw となるように配置



Zircolite

1. 下記URLから ***zircolite_win10_x64_2.9.7.7z*** をダウンロード
 - <https://github.com/wagga40/Zircolite/releases/tag/2.9.7>
2. 7ZIPを展開して出てきた***zircolite_win10***ディレクトリを
C:\¥sigma_workshop¥zircolite となるように配置
3. ***zircolite_win10.exe*** を ***zircolite.exe*** にリネーム

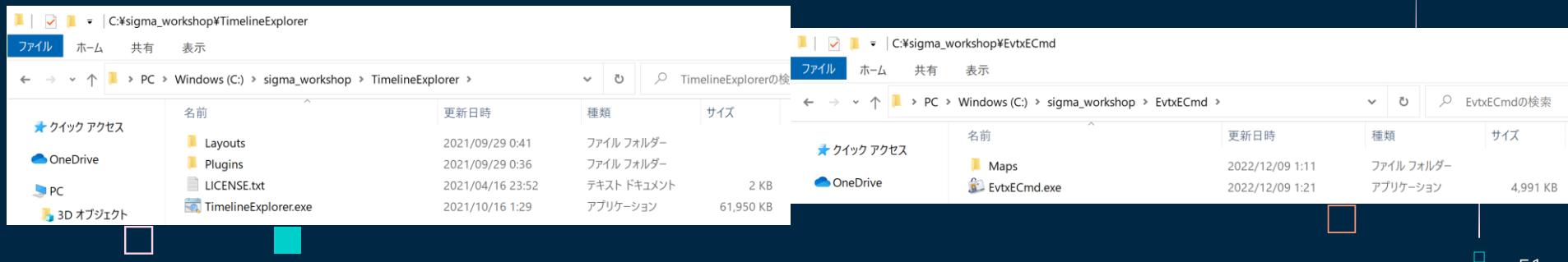


Timeline Explorer & EvtxECmd

1. 下記URLからTimeline ExplorerとEvtxECmdをダウンロード
 - インストールされている .NET Frameworkのバージョンに合わせる
 - <https://ericzimmerman.github.io/#!index.md>
2. ZIPを展開して出てきたファイルをそれぞれ

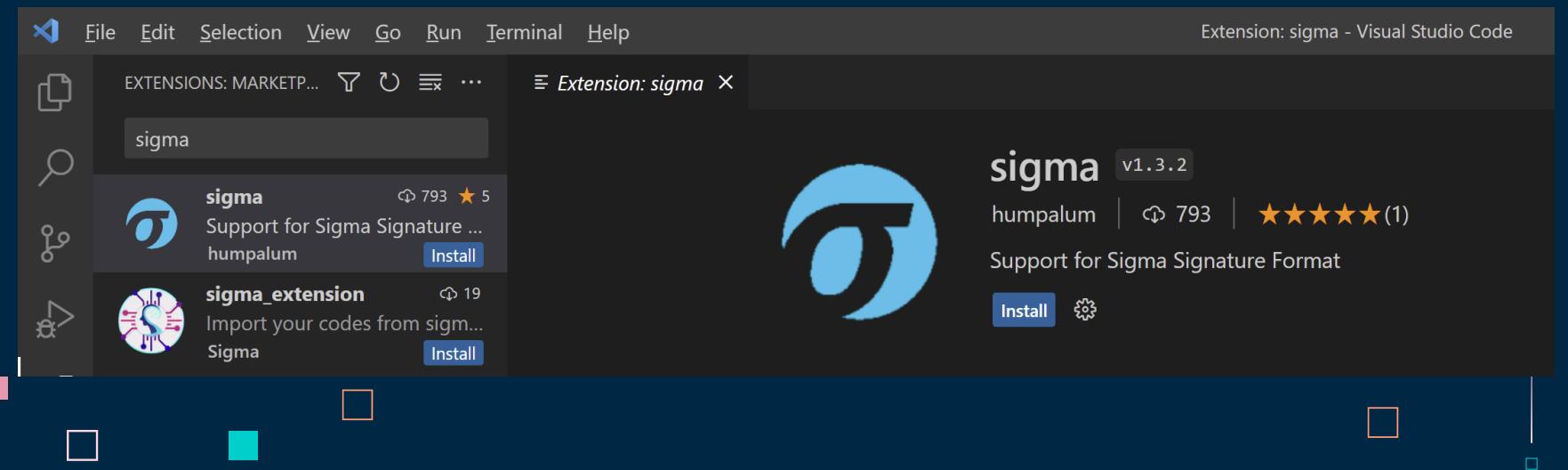
C:\\$sigma_workshop\\$TimelineExplorer と

C:\\$sigma_workshop\\$EvtxECmd となるように配置



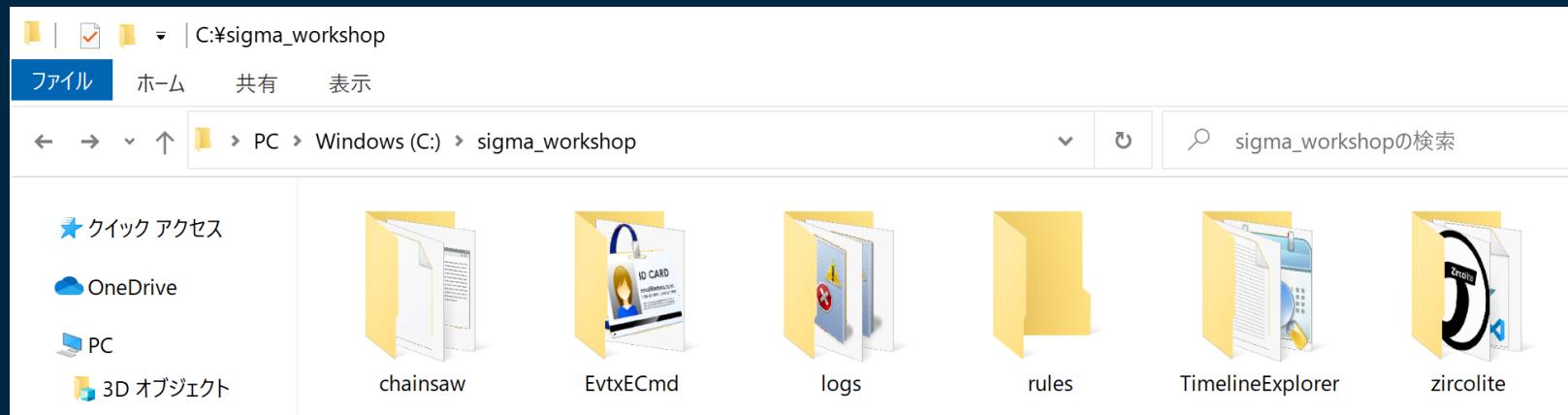
VSCode Extension

1. VSCodeを開き、Extensionsで *sigma* と検索
2. *humpalum.sigma* をインストール



C:¥sigma_workshop

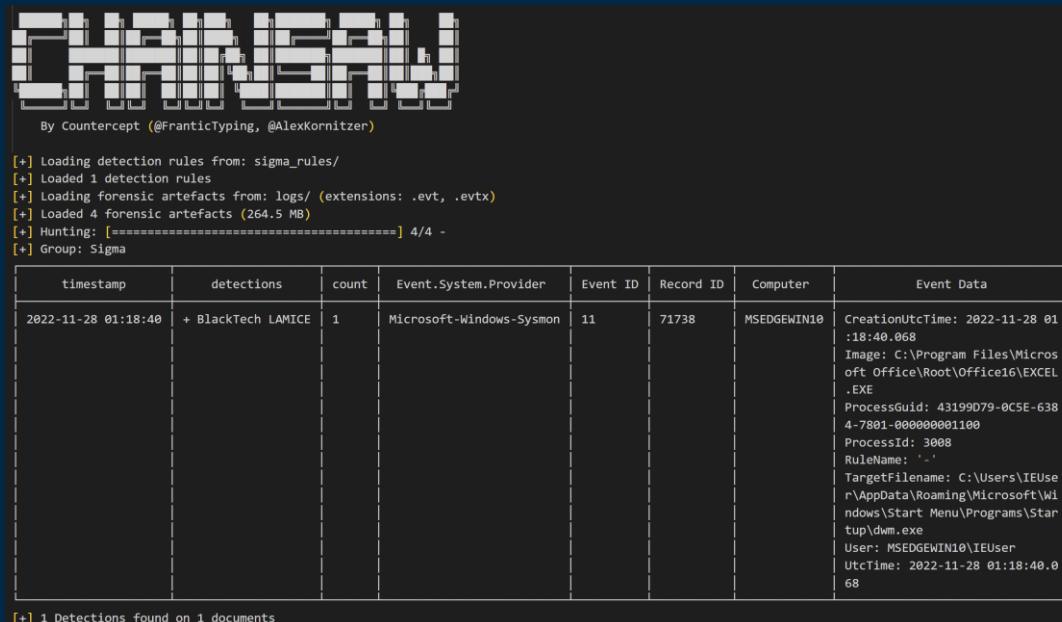
最終的に以下の構成となる



Sysmon with Chainsaw

Chainsaw

```
$ chainsaw hunt logs/ -s sigma_rules/ ¥  
--mapping mappings/sigma-event-logs-all.yml
```



The screenshot shows the Chainsaw application interface. At the top, there's a large watermark-style logo for "CHAINSAW" composed of small squares. Below it, the text "By Countercept (@FranticTyping, @AlexKornitzer)". The main area displays a terminal-like log output:

```
[+] Loading detection rules from: sigma_rules/  
[+] Loaded 1 detection rules  
[+] Loading forensic artefacts from: logs/ (extensions: .evt, .evtx)  
[+] Loaded 4 forensic artefacts (264.5 MB)  
[+] Hunting: [=====] 4/4 -  
[+] Group: Sigma
```

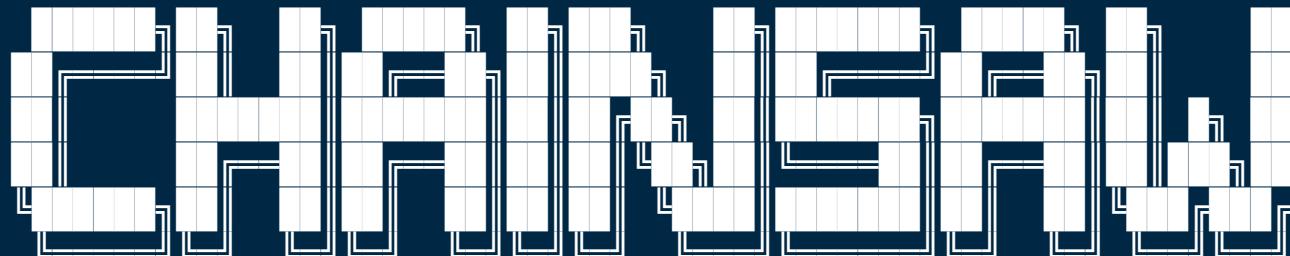
Below the log, there's a table with the following columns: timestamp, detections, count, Event.System.Provider, Event ID, Record ID, Computer, and Event Data. One row of data is shown:

timestamp	detections	count	Event.System.Provider	Event ID	Record ID	Computer	Event Data
2022-11-28 01:18:40	+ BlackTech LAMICE	1	Microsoft-Windows-Sysmon	11	71738	MSEDGEWIN10	 CreationUtcTime: 2022-11-28 01 :18:40.068 Image: C:\Program Files\Micros oft Office\Root\Office16\EXCEL .EXE ProcessGuid: 43199D79-0C5E-638 4-7801-00000001100 ProcessId: 3008 RuleName: '_' TargetFilename: C:\Users\IEUser r\AppData\Roaming\Microsoft\Wi ndows\Start Menu\Programs\Star tup\down.exe User: MSEDGEWIN10\IEUser UtcTime: 2022-11-28 01:18:40.0 68

At the bottom, the message "[+] 1 Detections found on 1 documents".

Chainsaw Lint

```
$ chainsaw lint --kind sigma rules/
```



By Countercept (@FranticTyping, @AlexKornitzer)

```
[+] Validating as sigma for supplied detection rules...
[!] sample_rule.yml: invalid condition: identifier not found - selection_one
[+] Validated 0 detection rules out of 1
```

Sysmon for Linux with Zircolite

Sysmon for Linux

- デフォルトでは/var/log/syslogに吐き出される
- ログフォーマット (Ubuntuの場合)

- MjH:i:s ubuntu sysmon:

```
<Event><System>...</System><EventData>...</EventData></Event>
```

```
Jan  2 08:29:23 ubuntu sysmon: <Event><System><Provider Name="Linux-Sysmon" Guid="
{ff032593-a8d3-4f13-b0d6-01fc615a0f97}" /><EventID>1</EventID><Version>5</Version><Level>4</
Level><Task>1</Task><Opcode>0</Opcode><Keywords>0x8000000000000000</Keywords><TimeCreated
SystemTime="2023-01-02T06:29:23.620103000Z" /><EventRecordID>1502</EventRecordID><Correlation
><Execution ProcessID="1130" ThreadID="1130" /><Channel>Linux-Sysmon/Operational</
Channel><Computer>ubuntu</Computer><Security UserID="0" /></System><EventData><Data Name="RuleName">-</
Data><Data Name="UtcTime">2023-01-02 06:29:43.012</Data><Data Name="ProcessGuid">
{55f63222-79d7-63b2-a584-6b0000000000}</Data><Data Name="ProcessId">2283</Data><Data Name="Image">/usr/
bin/python3.8</Data><Data Name="FileVersion">-</Data><Data Name="Description">-</Data><Data
Name="Product">-</Data><Data Name="Company">-</Data><Data Name="OriginalFileName">-</Data><Data
Name="CommandLine">/usr/bin/python3 /usr/bin/gnome-terminal</Data><Data Name="CurrentDirectory">/home/
alice</Data><Data Name="User">alice</Data><Data Name="LogonGuid">{55f63222-0000-0000-e803-000000000000}-
</Data><Data Name="LogonId">1000</Data><Data Name="TerminalSessionId">3</Data><Data
Name="IntegrityLevel">no level</Data><Data Name="Hashes">-</Data><Data Name="ParentProcessGuid">
{55f63222-79c7-63b2-c557-3a0d40560000}</Data><Data Name="ParentProcessId">1829</Data><Data
Name="ParentImage">/usr/bin/gnome-shell</Data><Data Name="ParentCommandLine">/usr/bin/gnome-shell</
Data><Data Name="ParentUser">alice</Data></EventData></Event>
```

Event - System

```
<System>
  <Provider Name="Linux-Sysmon" Guid="{ff032593-a8d3-4f13-b0d6-01fc615a0f97}" />
  <EventID>1</EventID>
  <Version>5</Version>
  <Level>4</Level>
  <Task>1</Task>
  <Opcode>0</Opcode>
  <Keywords>0x8000000000000000</Keywords>
  <TimeCreated SystemTime="2023-01-02T06:29:23.620103000Z" />
  <EventRecordID>1502</EventRecordID>
  <Correlation />
  <Execution ProcessID="1130" ThreadID="1130" />
  <Channel>Linux-Sysmon/Operational</Channel>
  <Computer>ubuntu</Computer>
  <Security UserId="0" />
</System>
```

Event - EventData

```
<EventData>
    <Data Name="RuleName">-</Data>
    <Data Name="UtcTime">2023-01-02 06:29:43.012</Data>
    <Data Name="ProcessGuid">{55f63222-79d7-63b2-a584-6b0000000000}</Data>
    <Data Name="ProcessId">2283</Data>
    <Data Name="Image">/usr/bin/python3.8</Data>
    <Data Name="FileVersion">-</Data>
    <Data Name="Description">-</Data>
    <Data Name="Product">-</Data>
    <Data Name="Company">-</Data>
    <Data Name="OriginalFileName">-</Data>
    <Data Name="CommandLine">/usr/bin/python3 /usr/bin/gnome-terminal</Data>
    <Data Name="CurrentDirectory">/home/alice</Data>
    <Data Name="User">alice</Data>
    <Data Name="LogonGuid">{55f63222-0000-0000-e803-000000000000}</Data>
    <Data Name="LogonId">1000</Data>
    <Data Name="TerminalSessionId">3</Data>
    <Data Name="IntegrityLevel">no level</Data>
    <Data Name="Hashes">-</Data>
    <Data Name="ParentProcessGuid">{55f63222-79c7-63b2-c557-3a0d40560000}</Data>
    <Data Name="ParentProcessId">1829</Data>
    <Data Name="ParentImage">/usr/bin/gnome-shell</Data>
    <Data Name="ParentCommandLine">/usr/bin/gnome-shell</Data>
    <Data Name="ParentUser">alice</Data>
</EventData>
```

Zircolite

- Sigmaルールを YAML から JSON に変換しないといけない

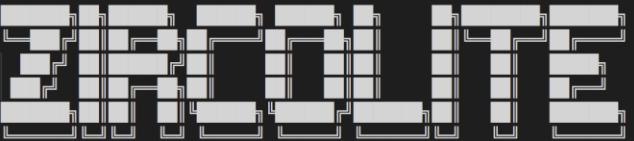
- <https://github.com/wagga40/Zircolite/blob/master/docs/Usage.md#generate-your-own-rulesets>

```
$ sigmac -t sqlite ¥
-c sysmon.yml ¥
-c zircolite.yml ¥
-d rules/ ¥
--output-fields title,id,description,author,falsepositives,level,filename ¥
--output-format json ¥
-r ¥
-o rules_linux_original.json ¥
--backend-option table=logs
```



Sysmon for Linux with Zircolite

```
$ zircolite --events syslog.log ¥  
--ruleset rules/[rules_linux].json --sysmon4linux
```



= Standalone SIGMA Detection tool for EVTX/Auditd/Sysmon Linux =-

```
[+] Checking prerequisites  
[+] Extracting EVTX Using 'tmp-KIKFNGFU' directory  
100%|██████████| 1/1 [00:00<00:00, 1.02it/s]  
[+] Processing EVTX  
100%|██████████| 1/1 [00:00<00:00, 8.26it/s]  
[+] Creating model  
[+] Inserting data  
100%|██████████| 3190/3190 [00:00<00:00, 23981.54it/s]  
[+] Cleaning unused objects  
[+] Loading ruleset from : rules/rules_linux_original.json  
[+] Executing ruleset - 1 rules  
- BlackTech mabackdoor [critical] : 1 events  
100%|██████████| 1/1 [00:00<00:00, 494.90it/s]  
[+] Results written in : detected_events.json  
[+] Cleaning  
Finished in 1 seconds
```

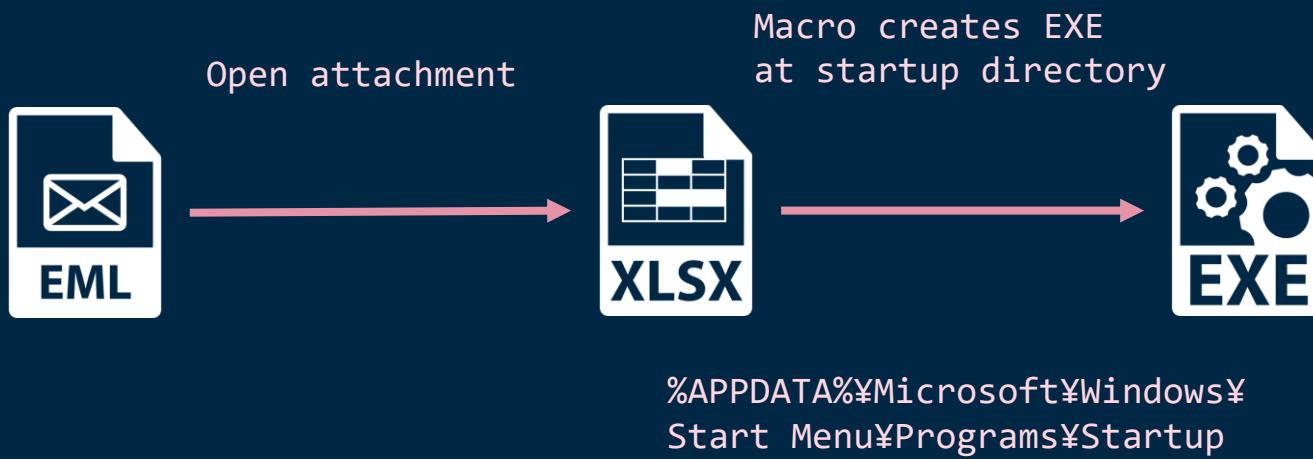
BlackTech LAMICE

BlackTech LAMICE

- BlackTechが2017年から使う悪性マクロ付きExcelドキュメント
 - JSAC 2022でTrendMicroが発表^[1]
- 長期間利用されている
 - 2017年: TSCookie
 - 2018年: IconDown
 - 2021年: Flagpro
- ペイロードの書き込みと永続化だけ行う
 - 直接実行はしない

[1] https://jsac.jpcert.or.jp/archive/2022/pdf/JSAC2022_8_hara_en.pdf

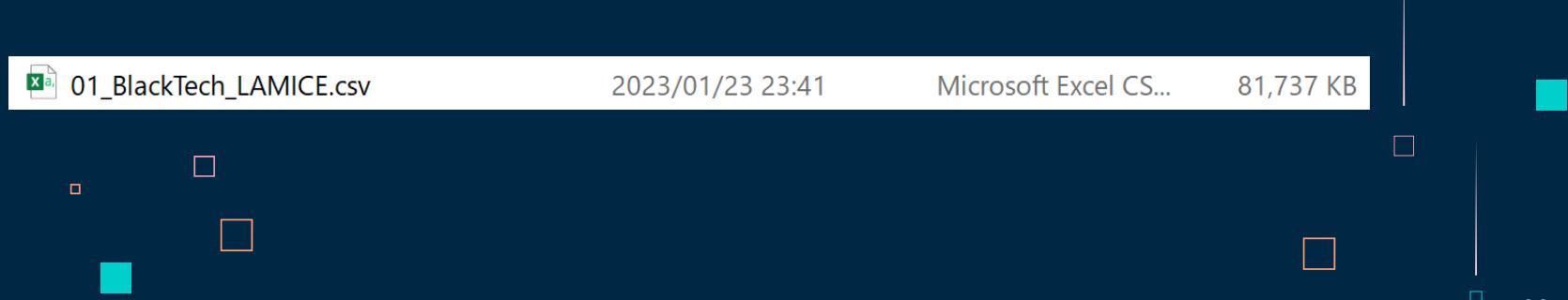
TTPs



(01/10) Convert Evtx to CSV

Timeline Explorerでログを見るために *C:\\$sigma_workshop\Logs* 配下にある *01_BlaсkTech_LAMICE.evtx* を EvtxECmd で変換

```
$ EvtxECmd -f C:\$sigma_workshop\logs\01_BlaсkTech_LAMICE.evtx  
--csv C:\$sigma_workshop --csvf 01_BlaсkTech_LAMICE.csv
```



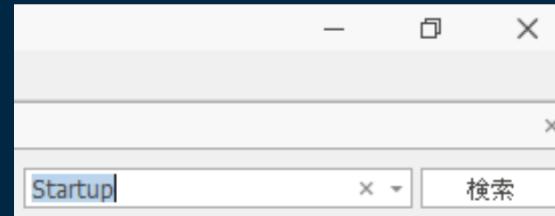
(02/10) View CSV in Timeline Explorer

Timeline Explorerを開きを `01_BlackTech_LAMICE.csv` をドラッグ
アンド ドロップする

Line	Tag	Record Number	Event Record Id	Time Created	Event Id	Level	Provider	Channel
1		1	25756	2022-11-28 01:0...	12	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
2		2	25757	2022-11-28 01:0...	12	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
3		3	25758	2022-11-28 01:0...	7	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
4		4	25759	2022-11-28 01:0...	7	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
5		5	25760	2022-11-28 01:0...	7	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
6		6	25761	2022-11-28 01:0...	7	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
7		7	25762	2022-11-28 01:0...	7	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
8		8	25763	2022-11-28 01:0...	7	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
9		9	25764	2022-11-28 01:0...	7	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
10		10	25765	2022-11-28 01:0...	7	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
11		11	25766	2022-11-28 01:0...	7	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
12		12	25767	2022-11-28 01:0...	7	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
13		13	25768	2022-11-28 01:0...	7	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
14		14	25769	2022-11-28 01:0...	12	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
15		15	25770	2022-11-28 01:0...	10	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
16		16	25771	2022-11-28 01:0...	7	Info	Microsoft-Windows-Sysmon	Microsoft-Windc
17		17	25772	2022-11-28 01:0...	12	Info	Microsoft-Windows-Sysmon	Microsoft-Windc

(03/10) Filter for specific events

LAMICEはStartupにファイルを書き込むので
“Startup” で検索してみる



(04/10) View malicious event

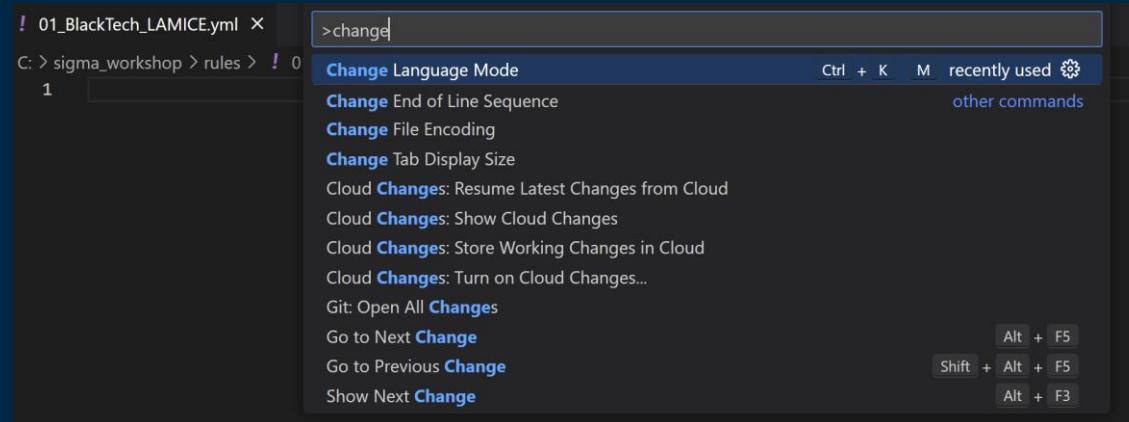
PayloadからJSONを取得可能

- このイベントを検知するようなSigmaルールを執筆する

```
{  
    "@Name": "Image",  
    "#text": "C:\\Program Files\\Microsoft Office\\Root\\Office16\\EXCEL.EXE"  
},  
{  
    "@Name": "TargetFilename",  
    "#text": "C:\\Users\\IEUser\\AppData\\Roaming\\Microsoft\\Windows\\Start Menu\\Programs\\Startup\\dwm.exe"  
},
```

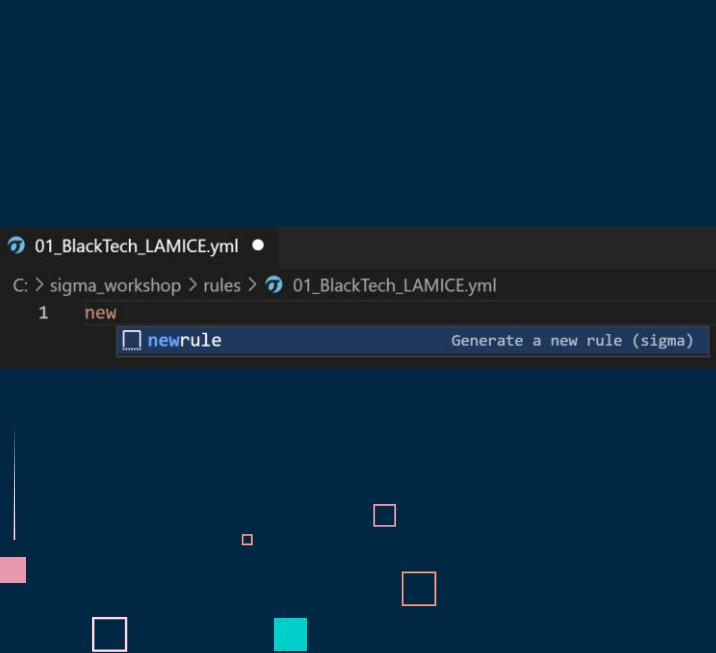
(05/10) Open VSCode

1. *C:¥sigma_workshop¥rules* に *01_BlaсkTech_LAMICE.yml* を作成し、VSCodeで開く
2. *View* → *Command Pallette (Ctrl+Shift+P)* → *Change Language Mode* → *sigma* で言語設定を反映



(06/10) Create basic components

`newrule` を使って雛形を作成



```
C: > sigma_workshop > rules > 01_BlackTech_LAMICE.yml
C: > sigma_workshop > rules > 01_BlackTech_LAMICE.yml
1 new
  newrule Generate a new rule (sigma)
```

```
01_BlackTech_LAMICE.yml 1 •
C: > sigma_workshop > rules > 01_BlackTech_LAMICE.yml
1   title: 01 BlackTech LAMICE
2   id: ed5886bc-f37d-46a7-85ee-5085f1015e0e
3   status: experimental
4   description: Detects
5   author: 
6   date: 2023/01/24
7   references:
8   |
9   | - Add Tag
10  | tags:
11  | |
12  | logsource:
13  |   category: process_creation
14  |   product: windows
15  | detection:
16  |   selection:
17  |   |
18  |   condition: selection
19  | falsepositives:
20  |   - Unknown
21  | level: critical
```

process_creation

- process_access
- registry_event
- ps_script
- file_event
- webserver
- image_load

(07/10) Write basic components

```
01_Bla... X  
C: > sigma_workshop > rules > 01_Bla...  
1   title: 01 BlackTech LAMICE  
2   id: ed5886bc-f37d-46a7-85ee-5085f1015e0e  
3   status: experimental  
4   description: Detects file creation by BlackTech LAMICE  
5   author: nao_sec  
6   date: 2023/01/24  
7   references:  
8     - https://jsac.jpcert.or.jp/archive/2022/pdf/JSAC2022\_8\_hara\_en.pdf  
9     - https://jp.security.ntt/resources/BlackTech\_2021.pdf  
Add Tag  
10  tags:  
11    - attack.persistence  
12    - attack.t1547.001  
13  logsource:  
14    category: process_creation  
15    product: windows  
16  detection:  
17    selection:  
18  
19      condition: selection  
20  falsepositives:  
21    - Unknown  
22  level: critical
```

(08/10) Write detection logic

```
detection:
  selection:
    # ファイル作成イベント
    EventID: 11

    # プロセスパスの末尾が "\\EXCEL.EXE"
    Image|endswith: "\\EXCEL.EXE"

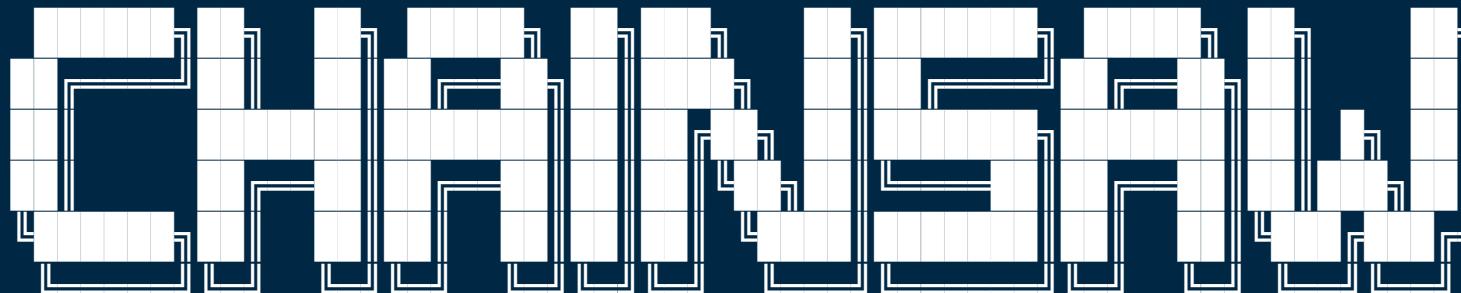
    # ファイルパスにスタートアップを含む
    TargetFilename|contains: "\\AppData\\Roaming\\Microsoft\\Windows\\Start Menu\\Programs\\Startup\\"

    # ファイルパスの末尾が ".exe"
    TargetFilename|endswith: ".exe"

  condition: selection
```

(09/10) Test by Chainsaw Lint

```
$ chainsaw lint --kind sigma ..../rules/01_BlaсkTech_LAMICE.yml
```



By Countercept (@FranticTyping, @AlexKornitzer)

- [+] Validating as sigma for supplied detection rules...
- [+] Validated 1 detection rules out of 1

(10/10) Hunt by Chainsaw

```
$ chainsaw hunt ../logs/ -s ../rules/ --mapping  
mappings/sigma-event-logs-all.yml
```

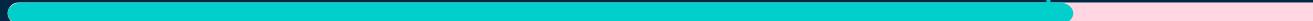
```
[+] Loading detection rules from: ../rules/  
[+] Loaded 1 detection rules  
[+] Loading forensic artefacts from: ../logs/ (extensions: .evtx, .evt)  
[+] Loaded 6 forensic artefacts (396.8 MB)  
[+] Hunting: [=====] 6/6 -  
[+] Group: Sigma
```

timestamp	detections	count	Event. System. Provider	Event ID	Record ID	Computer	Event Data
2022-11-28 01:18:40	+ 01 BlackTech LAMICE	1	Microsoft-Windows-Sysmon	11	71738	MSEdgeWIN10	CreationUtcTime: 2022-11-28 01:18:40.068 Image: C:\Program Files\Microsoft\Office\Root\Office16\EXCEL.EXE ProcessGuid: 43199D79-0C5E-6384-7801-000000001100 ProcessId: 3008 RuleName: '-' TargetFilename: C:\Users\IEUser\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\dwm.exe User: MSEdgeWIN10\IEUser UtcTime: 2022-11-28 01:18:40.068

```
[+] 1 Detections found on 1 documents
```

Exercise

04



LODEINFO

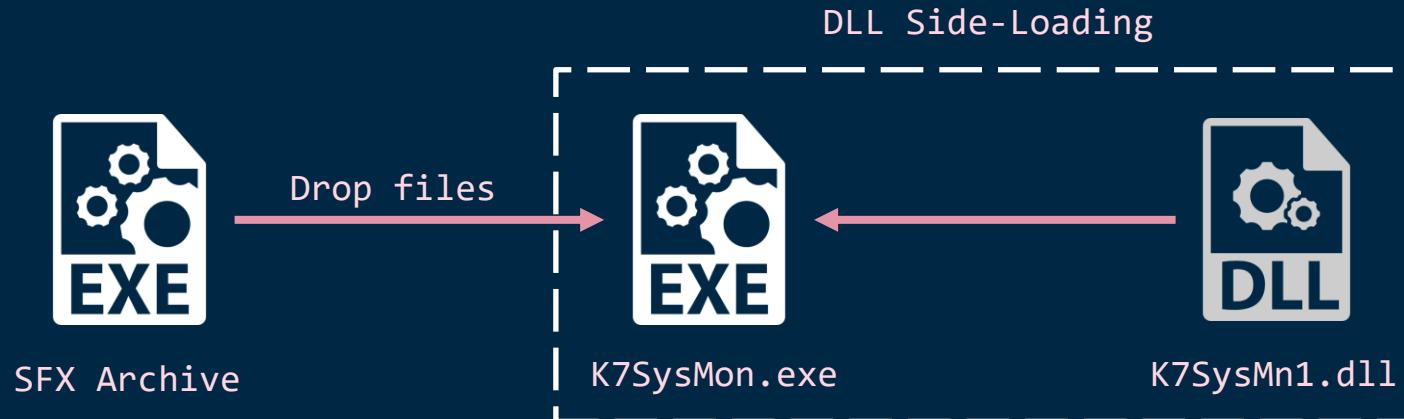
DLL Side-Loading

LODEINFO

- 2019年12月から活動している脅威アクターMirrorFaceが使うRAT
 - APT10が関与しているとも言われている^{[1][2]}
- 2022年も活発に攻撃を行っていた^[3]
 - 詳細はDay 1のエヌ・エフ・ラボラトリーズの発表を参照
- DLL Side-Loadingを利用する
 - 利用するEXEファイルは長期間変化しない

[1] <https://securelist.com/apt10-tracking-down-lodeinfo-2022-part-i/107742/>
[2] <https://securelist.com/apt10-tracking-down-lodeinfo-2022-part-ii/107745/>
[3] <https://www.welivesecurity.com/2022/12/14/unmasking-mirrorface-operation-liberalface-targeting-japanese-political-entities/>

TTPs



Considerations for creating rule

- DLL Side-Loadingで利用されるDLLの作成を検知するルール
 - 利用されるDLLファイル名は？
- DLL Side-LoadingのDLLのロードを検知するルール
 - どのEXEファイルが、どのDLLファイルを読み込む？

Operation RestyLink

LNK file

Operation RestyLink

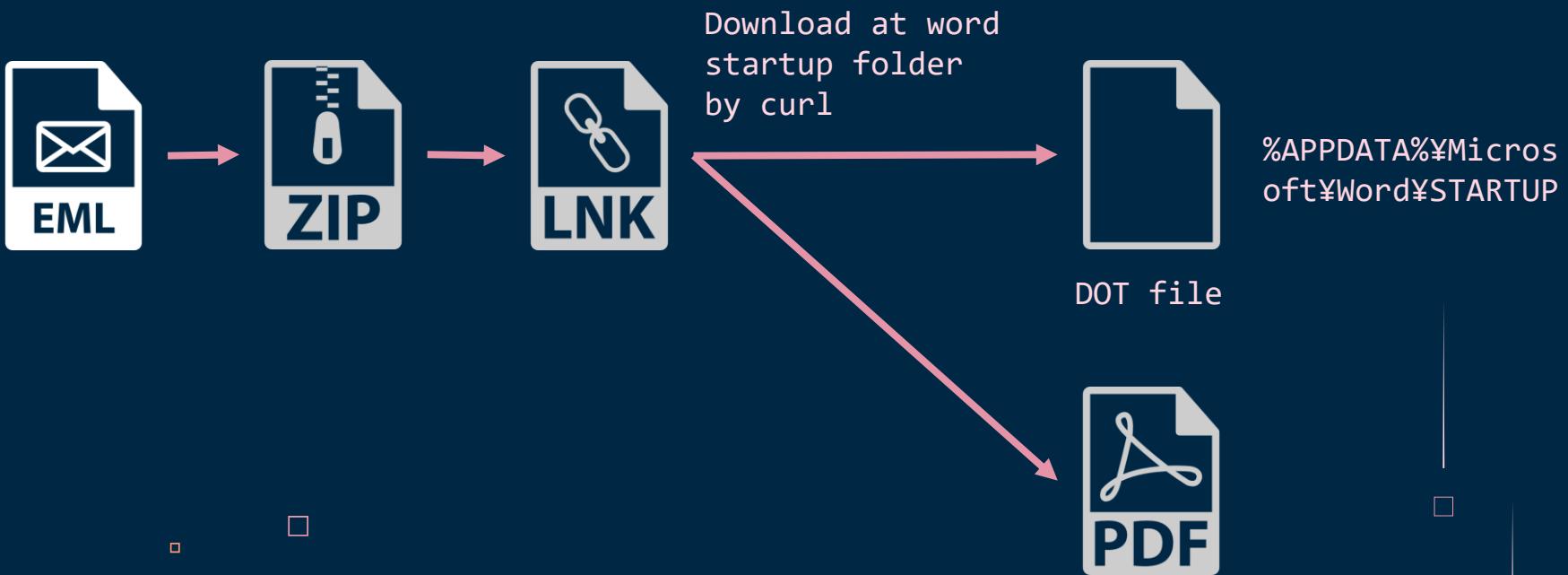
- 2022年3月頃から活動が確認されている攻撃キャンペーン^{[1][2]}
 - 日本語のLNKファイルやISOファイルを利用した攻撃が確認されている
 - 帰属は現時点で不明
 - 7-8月も攻撃が継続^[3]
- LNKファイルの挙動
 - C2サーバからWordのテンプレートファイル(DOT)をダウンロード
 - DOTファイルをWordのスタートアップフォルダに配置

[1] <https://security.macnica.co.jp/blog/2022/05/iso.html>

[2] <https://insight-jp.nttsecurity.com/post/102ho8o/operation-restylink>

[3] https://www.jpcert.or.jp/pr/2022/IR_Report2022Q2.pdf

TTPs



Considerations for creating rule

- LNKファイルの引数の処理を検知するルール
 - LNKファイルで行われる特徴的な挙動は？
- Curlを使ってWordのスタートアップにDOTファイルが作成される処理を検知するルール
 - Wordのスタートアップってどこ？

Lazarus

Operation SnatchCrypto

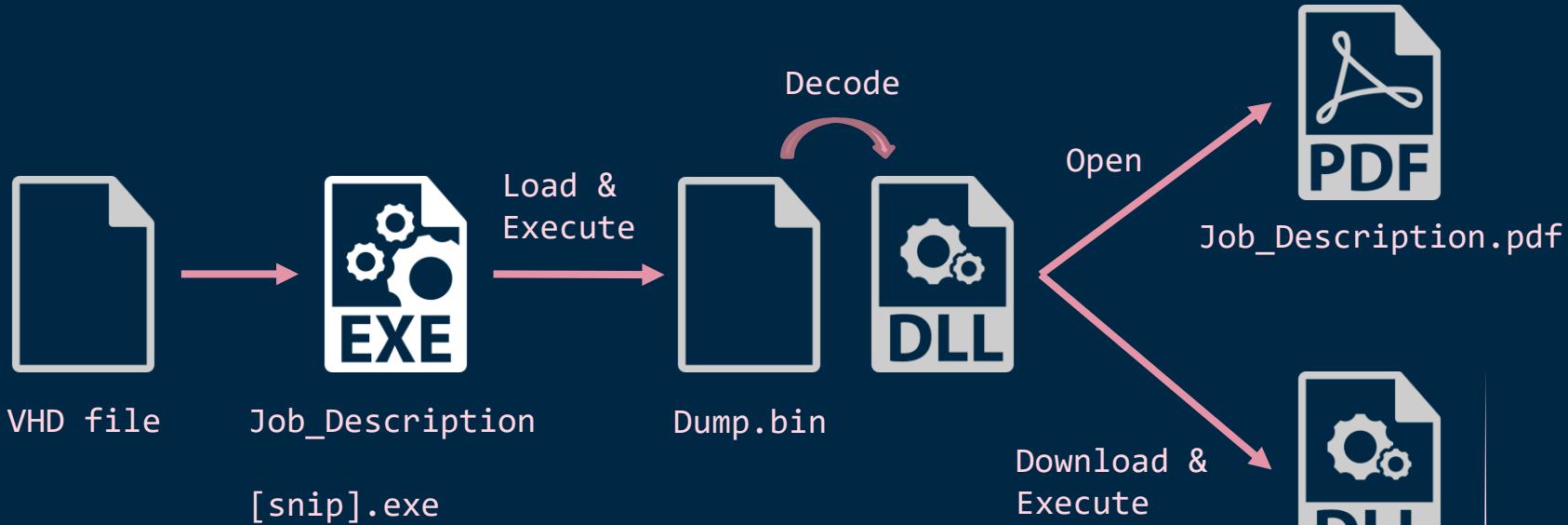
VHD and Files

Lazarus Operation SnatchCrypto

- 2019年から活動が確認されているLazarusのオペレーション
 - 仮想通貨関連事業者を狙う
- 2022年11月にVHDファイルを悪用した攻撃を観測
 - 日本の金融関連企業やベンチャーキャピタルを騙る^[1]
- VHDファイルをマウント後、内部のマルウェアを実行させる
 - VHDファイルには以下のファイルが含まれる
 - ダウンローダーDLLを復号を行うマルウェア
 - 暗号化されたダウンローダーDLL
 - デコイのPDFファイル

[1] <https://securelist.com/bluenoroff-methods-bypass-motw/108383/>

TTPs



Considerations for creating rule

- VHDファイルのマウント（実行）を検知するルール
 - マウント時にどのようなイベントが発生する？
- ファイル名にスペース文字が大量にある実行ファイルの実行を検知するルール
 - Sigmaで大量のスペースを表現するには？

TA410

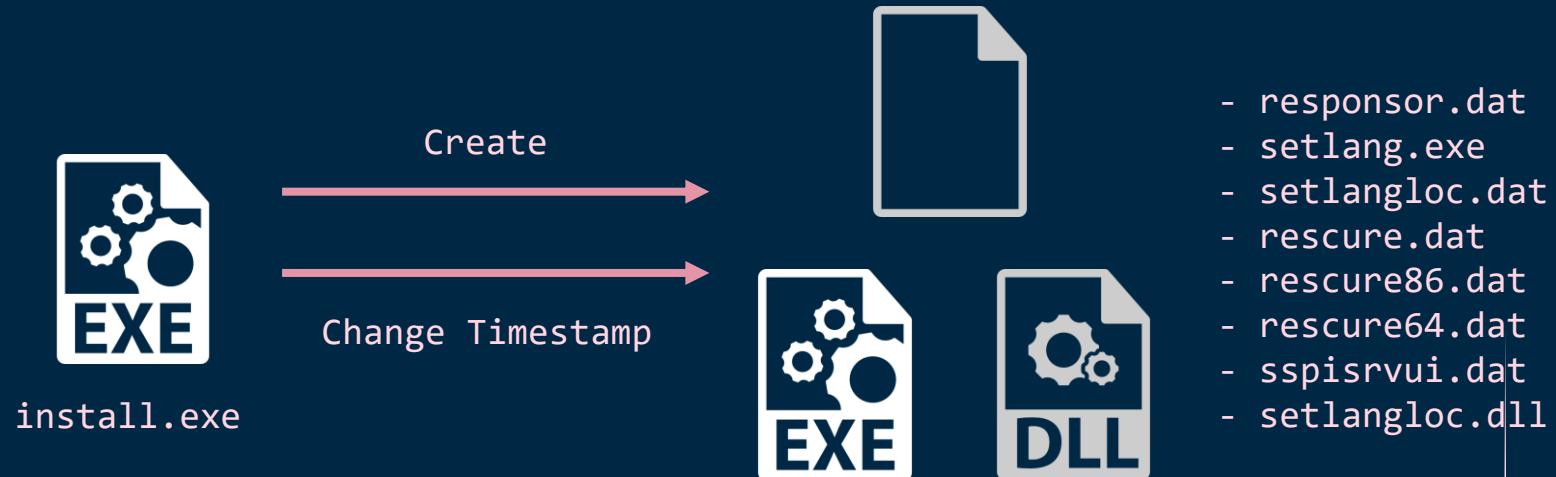
FlowCloud

TA410 FlowCloud

- 中国が背後にいると考えられている脅威アクター
 - 世界中の幅広い業種を対象に攻撃している
 - 2021年8月に日本の産業用機械の製造企業への攻撃が報告されている^[1]
 - 2022年にも日本企業関連の被害が報告されている^[2]
- C++で開発されたRAT
 - リソース領域に複数のモジュールを持つ
 - 作成したモジュールファイルの作成時刻を改ざんする

[1] <https://www.welivesecurity.com/2022/04/27/lookback-ta410-umbrella-cyberespionage-ttps-activity>
[2] https://www.jpcert.or.jp/pr/2022/IR_Report2022Q2.pdf

TTPs



Considerations for creating rule

- 特定のレジストリの書き込みを検知するルール
 - どんなレジストリ操作が行われている？
- 関連モジュールファイルのタイムスタンプの改ざんを検知するルール
 - タイムスタンプの書き換えイベントのIDは？
 - どんなファイルが使用されている？

おまけ

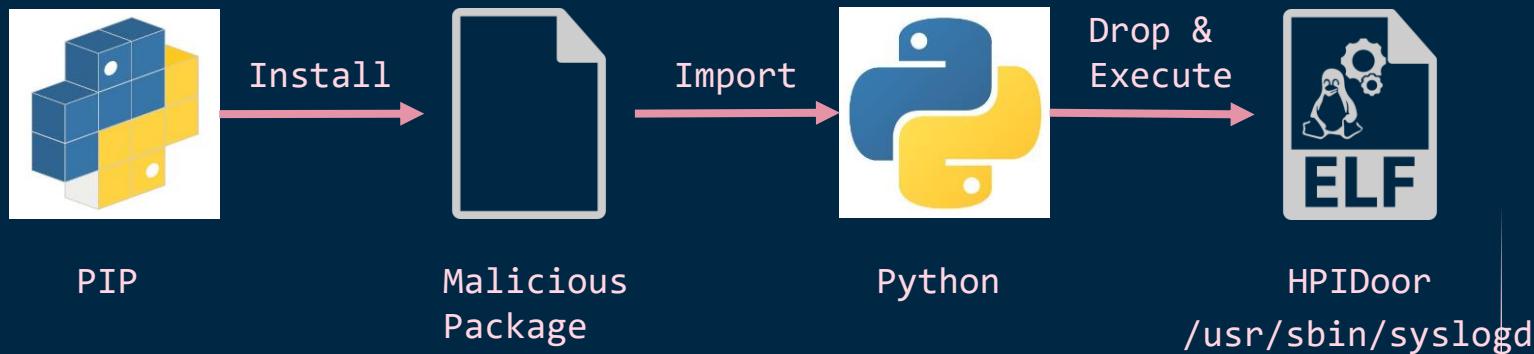
BlackTech HPIDoor

- BlackTechが2021年12月頃から使用しているバックドア
 - 2022年9月にJPCERTがブログで報告^{[1][2]}
- 悪性PyPIパッケージから感染
 - 攻撃者が複数の悪性PyPIパッケージをアップロード・公開
 - 被害者はPIPで誤ってインストール
 - 特定の関数を呼び出すことでELFバイナリが書き込まれ、実行される

[1] <https://blogs.jpcert.or.jp/ja/2022/09/bigip-exploit.html>
[2] <https://jfrog.com/blog/jfrog-discloses-3-remote-access-trojans-in-pypi/>

TTPs

おまけ



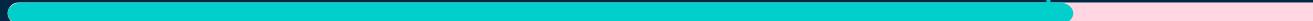
Considerations for creating rule

おまけ

- */usr/sbin/sysLogd* へのファイル書き込み（上書き）を検知するルール
 - 書き込みログは取れているか？要確認！
- */usr/sbin/sysLogd* の実行を検知するルール
 - 親プロセスは？

Summary

05



Summary

- カンファレンスの発表やベンダーのブログなどの公開情報から脅威アクターのTTPsを理解する
 - ハッシュやネットワークのIoCよりも変更されずらい特徴を見つける
- 日本を対象としたAPTで利用されるTTPsを紹介した
 - TTPsをSigmaルールに落とし込むことで脅威アクターの攻撃を検知した
- 書いたSigmaルールをコミュニティに共有してみましょう
 - 日本のセキュリティ業界のコミュニティを育てる

Do you have
any questions?

THANKS



@nao_sec

CREDITS: This presentation template was created by [Slidesgo](#), including icons by [Flaticon](#), and infographics & images by [Frepik](#)
Please keep this slide for attribution