



HAYABUSA

[English] | [日本語]

Stable Version v2.1.0 GitHub Downloads 21k GitHub Stars 1k Contributors 13

Black Hat Arsenal Asia 2022 CODE BLUE Bluebox 2022 SECCON 2023 Maintenance Level Actively Developed

rs report A+ codecov 75% Follow @SecurityYamato

Hayabusaについて

Hayabusaは、日本の[Yamato Security](#)グループによって作られたWindowsイベントログのファストフォレンジックタイムライン作成および脅威ハンティングツールです。 Hayabusaは日本語で「ハヤブサ」を意味し、ハヤブサが世界で最も速く、狩猟(hunting)に優れ、とても訓練しやすい動物であることから選ばれました。 Rustで開発され、マルチスレッドに対応し、可能な限り高速に動作するよう配慮されています。 SigmaルールをHayabusaルール形式に変換するツールも提供しています。 Hayabusaの検知ルールもSigmaと同様にYML形式であり、カスタマイズ性や拡張性に優れます。稼働中のシステムで実行してライブ調査することも、複数のシステムからログを収集してオフライン調査することも可能です。また、[Velociraptor](#)と[Hayabusa artifact](#)を用いることで企業向けの広範囲なスレットハンティングとインシデントレスポンスにも活用できます。出力は一つのCSVタイムラインにまとめられ、Excel、Timeline Explorer、Elastic Stack、Timesketch等で簡単に分析できるようになります。

関連プロジェクト

- [EnableWindowsLogSettings](#) - Sigmaベースの脅威ハンティングと、Windowsイベントログのファストフォレンジックタイムライン生成ツール。
- [Hayabusa Rules](#) - Hayabusaのための検知ルール。
- [Hayabusa Sample EVTXs](#) - Hayabusa/Sigma検出ルールをテストするためのサンプルevtxファイル。
- [Takajo](#) - Hayabusa結果の解析ツール。
- [WELA \(Windows Event Log Analyzer\)](#) - PowerShellで書かれたWindowsイベントログの解析ツール。

目次

- [Hayabusaについて](#)

- 関連プロジェクト
 - 目次
 - 主な目的
 - スレット(脅威)ハンティングと企業向けの広範囲なDFIR
 - フォレンジックタイムラインの高速生成
- スクリーンショット
 - 起動画面
 - ターミナル出力画面
 - イベント頻度タイムライン出力画面 (`-T`オプション)
 - 結果サマリ画面 (Results Summary)
 - HTMLの結果サマリ (`-H`オプション)
 - Excelでの解析
 - Timeline Explorerでの解析
 - Criticalアラートのフィルタリングとコンピュータごとのグルーピング
 - Elastic Stackダッシュボードでの解析
 - Timesketchでの解析
- タイムライン結果のインポートと解析について
- jqによるJSON形式の結果の解析
- 特徴&機能
- ダウンロード
- Gitクローン
- アドバンス: ソースコードからのコンパイル (任意)
 - Rustパッケージの更新
 - 32ビットWindowsバイナリのクロスコンパイル
 - macOSでのコンパイルの注意点
 - Linuxでのコンパイルの注意点
 - LinuxのMUSLバイナリのクロスコンパイル
- Hayabusaの実行
 - 注意: アンチウィルス/EDRの誤検知と遅い初回実行
 - Windows
 - Linux
 - macOS
- 主なコマンド
- 使用方法
 - デフォルトのヘルプメニュー
 - `csv-timeline`コマンド
 - `csv-timeline`コマンドの使用例
 - `csv-timeline`コマンドの設定ファイル
 - `json-timeline`コマンド
 - `json-timeline`コマンドの使用例と設定ファイル
 - `logon-summary`コマンド
 - `logon-summary`コマンドの使用例
 - `metrics`コマンド
 - `metrics`コマンドの使用例
 - `metrics`コマンドの設定ファイル
 - `pivot-keywords-list`コマンド

- pivot-keywords-listコマンドの使用例
- pivot-keywords-listの設定ファイル
- update-rulesコマンド
 - update-rulesコマンドの使用例
- level-tuningコマンド
 - level-tuningコマンドの使用例
 - level-tuningの設定ファイル
- set-default-profileコマンド
- list-profilesコマンド
- アドバンス
 - GeolPのログエンリッチメント
 - GeolPの設定ファイル
 - GeolPデータベースの自動アップデート
- サンプルevtxファイルでHayabusaをテストする
- HayabusaのCSVとJSON/L出力
 - 出力プロファイル
 - 1. minimalプロファイルの出力
 - 2. standardプロファイルの出力
 - 3. verboseプロファイルの出力
 - 4. all-field-infoプロファイルの出力
 - 5. all-field-info-verboseプロファイルの出力
 - 6. super-verboseプロファイルの出力
 - 7. timesketch-minimalプロファイルの出力
 - 8. timesketch-verboseプロファイルの出力
 - プロファイルの比較
 - Profile Field Aliases
 - Levelの省略
 - MITRE ATT&CK戦術の省略
 - Channel情報の省略
- その他の省略
 - プログレスバー
 - カラー出力
 - 結果のサマリ (Results Summary)
 - イベント頻度タイムライン
- Hayabusaルール
 - Hayabusa v.s. 変換されたSigmaルール
- その他のWindowsイベントログ解析ツールおよび関連リソース
- Windowsイベントログ設定のススメ
- Sysmon関係のプロジェクト
- コミュニティによるドキュメンテーション
 - 英語
 - 日本語
- 貢献
- バグの報告
- ライセンス
- Twitter

主な目的

スレット(脅威)ハンティングと企業向けの広範囲なDFIR

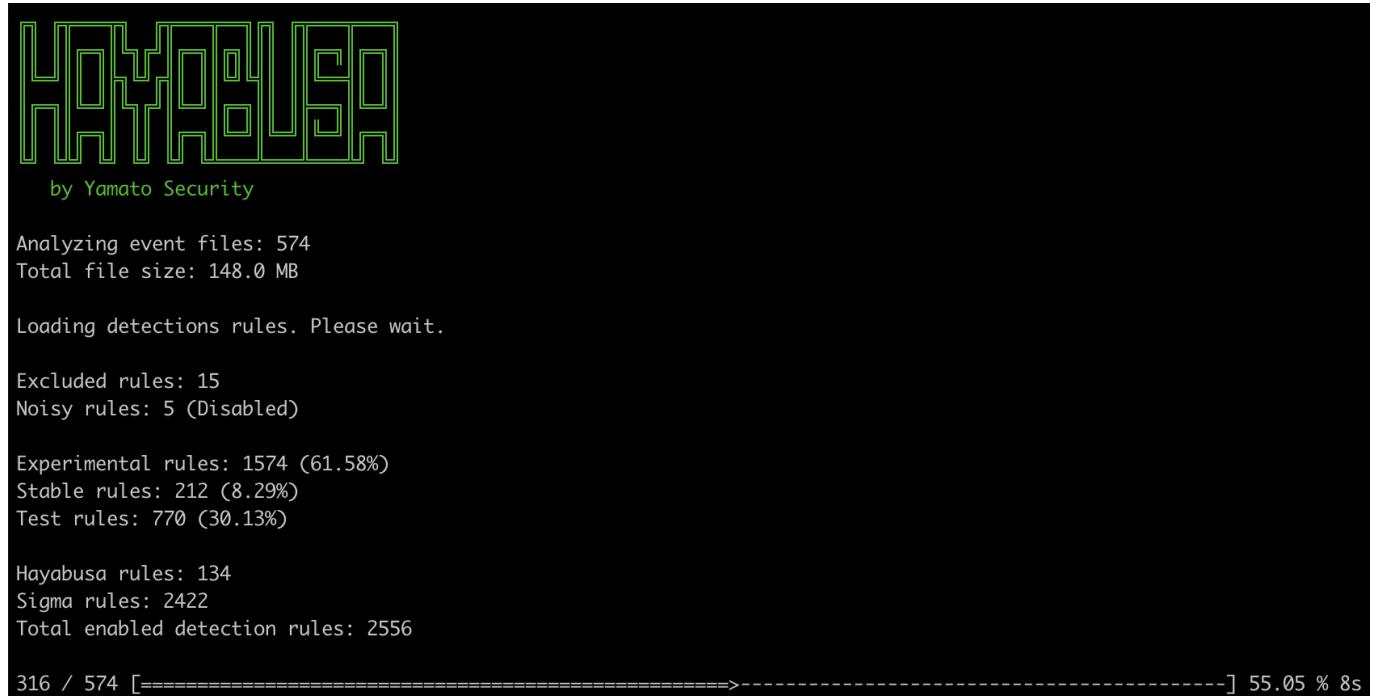
Hayabusaには現在、3250以上のSigmaルールと約150のHayabusa検知ルールがあり、定期的にルールが追加されています。 [Velociraptor](#)の[Hayabusa artifact](#)を用いることで企業向けの広範囲なスレットハンティングだけでなく DFIR(デジタルフォレンジックとインシデントレスポンス)にも無料で利用することができます。 この2つのオープンソースを組み合わせることで、SIEMが設定されていない環境でも実質的に遡及してSIEMを再現することができます。 具体的な方法は[Eric Capuano](#)の[こちら](#)の動画で学ぶことができます。

フォレンジックタイムラインの高速生成

Windowsのイベントログは、1) 解析が困難なデータ形式であること、2) データの大半がノイズであり調査に有用でないことから、従来は非常に長い時間と手間かかる解析作業となっていました。 Hayabusaは、有用なデータのみを抽出し、専門的なトレーニングを受けた分析者だけでなく、Windowsのシステム管理者であれば誰でも利用できる読みやすい形式で提示することを主な目的としています。 Hayabusaは従来のWindowsイベントログ分析解析と比較して、分析者が20%の時間で80%の作業を行えるようにすることを目指しています。

スクリーンショット

起動画面



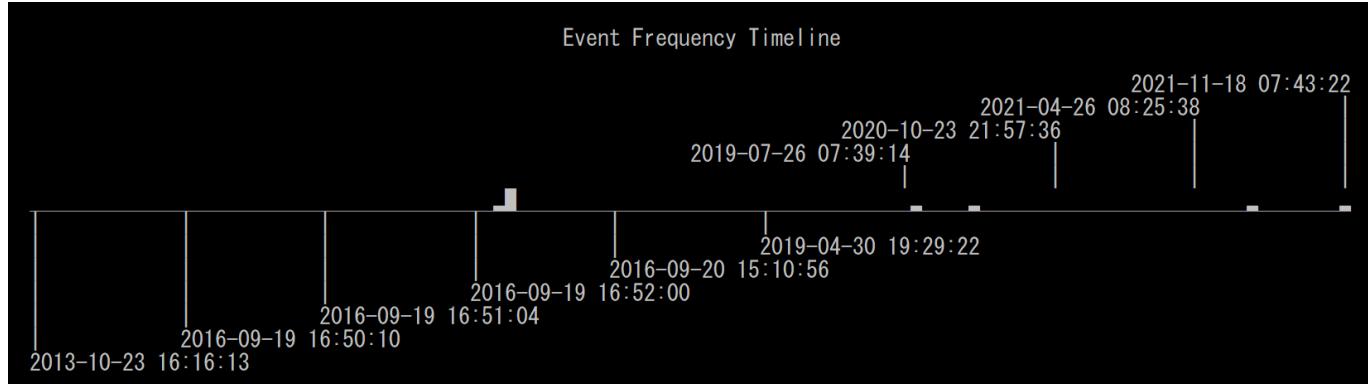
ターミナル出力画面

```

2021-12-13 17:21:30.845 +09:00 || fs03vuln.offsec.lan || Sec || 5145 || info || 1176260 || NetShare File Access || User: admogg | ShareName: \\*\C$ | SharePath: \??\C:\ | Path: Windows\System32\Drivers\etc | IP-Addr: 10.23.23.9 | LID: 0x8ca6e1d
2021-12-13 21:55:45.250 +09:00 || rootdc1.offsec.lan || Sys || 7045 || info || 1467331 || Svc Installed || Svc: BT0BTO | Path: %COMSPEC% /Q /c e
cho cd ^> \\127.0.0.1\%C$\_output 2^>^&1 > %TEMP%\execute.bat & %COMSPEC% /Q /c %TEMP%\execute.bat & del %TEMP%\execute.bat | Acct: Local
System | StartType: demand start
2021-12-13 21:55:45.250 +09:00 || - - - || Rare Service Installations || [condition] count() by ServiceName < 5 in timeframe [result] count:1 ServiceName:BT0BTO timeframe:7d
2021-12-13 21:55:45.250 +09:00 || rootdc1.offsec.lan || Sec || 4697 || info || 236864754 || Svc Installed || Svc: BT0BTO | Path: %COMSPEC% /Q /c
echo cd ^> \\127.0.0.1\%C$\_output 2^>^&1 > %TEMP%\execute.bat & %COMSPEC% /Q /c %TEMP%\execute.bat & del %TEMP%\execute.bat | User: adm
ogg | SvcAccount: LocalSystem | SvcType: 0x10 | SvcStartType: 3 | LID: 0x2cff42b44
2021-12-14 23:42:48.182 +09:00 || rootdc1.offsec.lan || Sec || 4776 || info || 237294513 || NTLM Logon To Local Account || User: hack1 | Comp: a
ttacker | Status: 0x0
2021-12-14 23:42:48.182 +09:00 || rootdc1.offsec.lan || Sec || 4624 || info || 237294514 || Logon (Type 3 Network) || User: hack1 | Comp: attack
er | IP-Addr: 10.23.123.11 | LID: 0x308fabb0c
2021-12-14 23:42:48.182 +09:00 || rootdc1.offsec.lan || Sec || 4624 || med || 237294514 || Pass the Hash Activity 2 || User: hack1 | Comp: att
acker | IP-Addr: 10.23.123.11 | Proc: - | LID: 0x308fabb0c
2021-12-14 23:42:48.690 +09:00 || rootdc1.offsec.lan || Sec || 4776 || info || 237294516 || NTLM Logon To Local Account || User: hack1 | Comp:
| Status: 0x0
2021-12-14 23:42:48.690 +09:00 || rootdc1.offsec.lan || Sec || 4624 || info || 237294517 || Logon (Type 3 Network) || User: hack1 | Comp: - | IP
-Addr: 10.23.123.11 | LID: 0x308fb82ad
2021-12-14 23:42:48.693 +09:00 || rootdc1.offsec.lan || Sec || 5140 || info || 237294519 || NetShare Access || User: hack1 | ShareName: \\*\IPC$ | SharePath: | IP-Addr: 10.23.123.11 | LID: 0x308fb82ad
2021-12-14 23:42:48.908 +09:00 || rootdc1.offsec.lan || Sec || 4781 || high || 237294532 || Suspicious Computer Account Name Change CVE-2021-42
287 || OldTgtUser: compnay-88$ | NewTgtUser: rootdc1 | TgtSID: S-1-5-21-4230534742-2542757381-3142984815-1296 | User: hack1 | SID: S-1-5-2
1-4230534742-2542757381-3142984815-1234 | Domain: OFFSEC | TgtDomain: OFFSEC | PrivList: - | LID: 0x308fabb0c
2021-12-14 23:42:48.908 +09:00 || rootdc1.offsec.lan || Sec || 4781 || high || 237294532 || Suspicious Computer Account Name Change CVE-2021-42
287 || OldTgtUser: compnay-88$ | NewTgtUser: rootdc1 | TgtSID: S-1-5-21-4230534742-2542757381-3142984815-1296 | User: hack1 | SID: S-1-5-2
1-4230534742-2542757381-3142984815-1234 | Domain: OFFSEC | TgtDomain: OFFSEC | PrivList: - | LID: 0x308fabb0c
2021-12-14 23:42:49.222 +09:00 || rootdc1.offsec.lan || Sec || 4768 || info || 237294534 || Kerberos TGT Requested || User: rootdc1 | Svc: krbtg
t | IP-Addr: ::ffff:10.23.123.11 | Status: 0x0 | PreAuthType: 2

```

イベント頻度タイムライン出力画面 (-Tオプション)



結果サマリ画面 (Results Summary)

Results Summary:

Events with hits / Total events: 19,545 / 76,967 (Data reduction: 57,422 events (74.61%))

Total | Unique detections: 32,684 | 554
 Total | Unique critical detections: 46 (0.14%) | 18 (3.25%)
 Total | Unique high detections: 6,141 (18.79%) | 250 (45.13%)
 Total | Unique medium detections: 1,472 (4.50%) | 156 (28.16%)
 Total | Unique low detections: 6,771 (20.72%) | 76 (13.72%)
 Total | Unique informational detections: 18,254 (55.85%) | 54 (9.75%)

Dates with most total detections:

critical: 2019-07-19 (15), **high:** 2016-09-20 (3,656), **medium:** 2019-05-19 (165), **low:** 2016-09-20 (3,780), **informational:** 2016-08-19 (2,105)

Top 5 computers with most unique detections:

critical: MSEdgeWIN10 (6), IEWIN7 (3), FS03.offsec.lan (2), rootdc1.offsec.lan (2), srvdefender01.offsec.lan (2)
high: MSEdgeWIN10 (109), IEWIN7 (70), FS03.offsec.lan (31), fs03vuln.offsec.lan (27), IE10Win7 (23)
medium: MSEdgeWIN10 (62), IEWIN7 (38), FS03.offsec.lan (16), IE10Win7 (15), PC01.example.corp (14)
low: MSEdgeWIN10 (35), IEWIN7 (18), FS03.offsec.lan (16), fs03vuln.offsec.lan (13), IE10Win7 (11)
informational: MSEdgeWIN10 (18), IEWIN7 (17), fs01.offsec.lan (16), PC01.example.corp (13), IE8Win7 (12)

Top critical alerts:

Sticky Key Like Backdoor Usage (10)
 Meterpreter or Cobalt Strike Getsystem Service Installation (6)
 Active Directory Replication from Non Machine Account (6)
 Windows Defender Alert (4)
 WannaCry Ransomware (4)

Top high alerts:

Metasploit SMB Authentication (3,562)
 Malicious Svc Possibly Installed (271)
 Susp Svc Installed (257)
 PowerShell Scripts Installed as Services (253)
 Suspicious Service Installation Script (250)

Top medium alerts:

Potentially Malicious PwSh (235)
 Proc Injection (104)
 Reg Key Value Set_Sysmon Alert (103)
 Suspicious Remote Thread Target (93)
 Cscript Visual Basic Script Execution (60)

Top low alerts:

Logon Failure_Wrong Password (3,564)
 Susp CmdLine (Possible LOLBIN) (1,418)
 Non Interactive PowerShell (325)
 Rare Service Installations (321)
 Windows Processes Suspicious Parent Directory (282)

Top informational alerts:

Proc Exec (11,173)
 NetShare File Access (2,564)
 PwSh Scriptblock (789)
 PwSh Pipeline Exec (680)
 NetShare Access (433)

Explicit Logon (342)
 Svc Installed (331)
 New Non-USB PnP Device (268)
 Logon (Type 3 Network) (228)
 File Created (210)

Elapsed Time: 00:00:28.827

HTMLの結果サマリ (-Hオプション)

General Overview

- Start time: 2022/10/02 03:16
- Excluded rules: 12
- Noisy rules: 5 (Disabled)
- Experimental rules: 2036 (67.24%)
- Stable rules: 213 (7.03%)
- Test rules: 779 (25.73%)
- Hayabusa rules: 138
- Sigma rules: 2890
- Total enabled detection rules: 3028
- Elapsed Time: 00:00:23.844

Results Summary

- Saved alerts and events: 19,545
- Total events analyzed: 76,967
- Data reduction: 57,422 events (74.61%)
- Dates with most total detections:
 - critical: 2019-07-19 (15)
 - high: 2016-09-20 (3,656)
 - medium: 2019-05-19 (165)
 - low: 2016-09-20 (3,780)
 - informational: 2016-08-19 (2,105)

Computers with most unique critical detections:

- MSEDGEWIN10 (6)
- IEWIN7 (3)
- srvdefender01.offsec.lan (2)
- FS03.offsec.lan (2)
- rootdc1.offsec.lan (2)
- alice.insecurebank.local (1)
- IE10Win7 (1)
- DC1.insecurebank.local (1)
- win10-02.offsec.lan (1)
- fs01.offsec.lan (1)
- fs03vuln.offsec.lan (1)
- DESKTOP-PIU87N6 (1)

Computers with most unique high detections:

- MSEDGEWIN10 (112)
- IEWIN7 (72)

Top critical alerts:

- [Sticky Key Like Backdoor Usage](#) (10)
- [Meterpreter or Cobalt Strike Getsystem Service Installation](#) (6)
- [Active Directory Replication from Non Machine Account](#) (6)
- [Windows Defender Alert](#) (4)
- [WannaCry Ransomware](#) (4)
- [Dumpert Process Dumper](#) (3)
- [SystemNightmare Exploitation Script Execution](#) (2)
- [Mimikatz MemSSP Default Log File Creation](#) (2)
- [Suspicious LSASS Process Clone](#) (2)
- [TrustedPath UAC Bypass Pattern](#) (2)
- [SMB Relay Attack Tools](#) (1)
- [smbexec.py Service Installation](#) (1)
- [Malicious Named Pipe](#) (1)
- [CobaltStrike Service Installations in Registry](#) (1)
- [Lsass Memory Dump via Comsvcs DLL](#) (1)

Top high alerts:

- [Metasploit SMB Authentication](#) (3,562)
- [Malicious Svc Possibly Installed](#) (271)

Excelでの解析

Time	Computername	EventID	Level	Alert	Details
2021-05-03 17:58:38.774 +09:00	webiis01.offsec.lan	4624	informational	Logon Type 3 - Network	User: adminmig : Workstation: - : IP Address: 10.23.23.9 : Port: 62234 : LogonID: 0x258b9ee5
2021-05-03 17:58:38.775 +09:00	webiis01.offsec.lan	4624	informational	Logon Type 3 - Network	User: adminmig : Workstation: - : IP Address: 10.23.23.9 : Port: 62235 : LogonID: 0x258b9ef8
2021-05-03 17:58:38.775 +09:00	webiis01.offsec.lan	4624	informational	Logon Type 3 - Network	User: adminmig : Workstation: - : IP Address: 10.23.23.9 : Port: 62236 : LogonID: 0x258b9efd
2021-05-03 21:06:57.954 +09:00	win10-02.offsec.lan	1	high	Process Creation Sysmon Rule Alert	Rule: technique_id=T1059,technique_name=Command-Line Interface : Command: C:\windows\sysmon.exe
2021-05-03 21:06:57.954 +09:00	win10-02.offsec.lan	1	critical	Sticky Key Like Backdoor Usage	
2021-05-15 05:39:33.214 +09:00	fs01.offsec.lan	1102	high	Security log was cleared	User: adminmig
2021-05-19 06:18:40.607 +09:00	rootdc1.offsec.lan	150	critical	DNS Server Error Failed Loading the ServerLevelPluginDLL	
2021-05-19 06:18:40.607 +09:00	rootdc1.offsec.lan	150	high	Possible CVE-2021-1675 Print Spooler Exploitation	
2021-05-19 06:18:40.607 +09:00	rootdc1.offsec.lan	150	critical	Mimikatz Use	
2021-05-19 06:23:27.038 +09:00	rootdc1.offsec.lan	150	critical	DNS Server Error Failed Loading the ServerLevelPluginDLL	
2021-05-19 06:23:27.038 +09:00	rootdc1.offsec.lan	150	high	Possible CVE-2021-1675 Print Spooler Exploitation	
2021-05-19 06:23:27.038 +09:00	rootdc1.offsec.lan	150	critical	Mimikatz Use	
2021-05-19 06:30:17.318 +09:00	rootdc1.offsec.lan	4688	high	Possible CVE-2021-1675 Print Spooler Exploitation	
2021-05-19 06:30:17.318 +09:00	rootdc1.offsec.lan	4688	critical	Mimikatz Use	
2021-05-19 06:30:17.318 +09:00	rootdc1.offsec.lan	4688	high	Relevant Anti-Virus Event	
2021-05-19 06:33:49.548 +09:00	rootdc1.offsec.lan	770	critical	DNS Server Error Failed Loading the ServerLevelPluginDLL	
2021-05-19 06:33:49.548 +09:00	rootdc1.offsec.lan	770	high	Possible CVE-2021-1675 Print Spooler Exploitation	
2021-05-19 06:33:49.548 +09:00	rootdc1.offsec.lan	770	high	Relevant Anti-Virus Event	
2021-05-19 06:33:49.548 +09:00	rootdc1.offsec.lan	770	critical	Mimikatz Use	
2021-05-20 21:49:31.863 +09:00	fs01.offsec.lan	1102	high	Security log was cleared	User: adminmig
2021-05-20 21:49:46.875 +09:00	fs01.offsec.lan	4648	informational	Explicit Logon	Source User: FS01\$: Target User: sshd_5848 : IP Address: - : Process: C:\Program Files\OpenSSH\OpenSSH-7.4p1-1.1.0.1-1~1\sshd.exe : Port: 22 : LogonID: 0x3c569ed
2021-05-20 21:49:46.876 +09:00	fs01.offsec.lan	4624	low	Logon Type 5 - Service	User: sshd_5848 : Workstation: - : IP Address: - : Port: - : LogonID: 0x3c569ed
2021-05-20 21:49:46.876 +09:00	fs01.offsec.lan	4672	informational	Admin Logon	User: sshd_5848 : LogonID: 0x3c569ed
2021-05-20 21:49:52.315 +09:00	fs01.offsec.lan	4776	informational	NTLM Logon to Local Account	User: NOUSER : Workstation FS01 : Status: 0xc0000064
2021-05-20 21:49:52.315 +09:00	fs01.offsec.lan	4625	informational	Logon Failure - Username does not exist	User: NOUSER : Type: 8 : Workstation: FS01 : IP Address: - : SubStatus: 0xc0000064 : AuthP

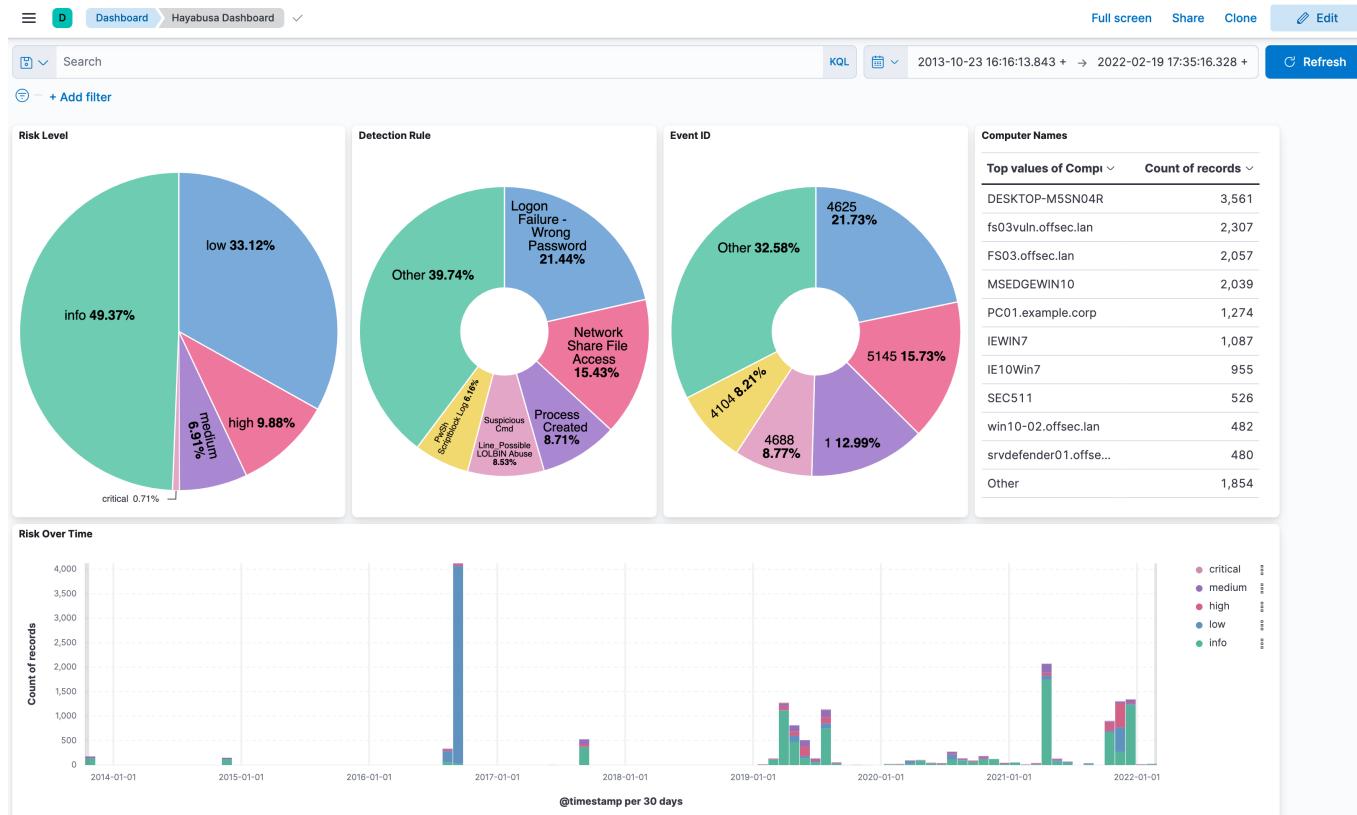
Timeline Explorerでの解析

Time	Computername	Eventid	Level	Alert	Details
2021-05-22 05:43:18.227 +09:00	fs01.offsec.lan	4648	informational	Explicit Logon	Source User: FS01\$: Target User: admmig
2021-05-22 05:43:22.562 +09:00	fs01.offsec.lan	4625	low	Logon Failure - Wrong Password	User: admmig@offsec.lan : Type: 8 : Wor
2021-05-22 05:43:49.345 +09:00	fs01.offsec.lan	4625	low	Logon Failure - Wrong Password	User: admmig@offsec.lan : Type: 8 : Wor
2021-05-22 05:43:50.131 +09:00	fs01.offsec.lan	4625	low	Logon Failure - Wrong Password	User: admmig@offsec.lan : Type: 8 : Wor
2021-05-22 05:43:50.607 +09:00	fs01.offsec.lan	4625	low	Logon Failure - Wrong Password	User: admmig@offsec.lan : Type: 8 : Wor
2021-05-22 05:43:50.866 +09:00	fs01.offsec.lan	4625	low	Logon Failure - Wrong Password	User: admmig@offsec.lan : Type: 8 : Wor
2021-05-23 06:56:57.685 +09:00	fs01.offsec.lan	1102	high	Security log was cleared	User: admmig
2021-05-23 06:57:11.842 +09:00	fs01.offsec.lan	4688	high	Relevant Anti-Virus Event	
2021-05-23 06:57:11.842 +09:00	fs01.offsec.lan	4688	critical	Mimikatz Use	
2021-05-26 22:02:27.149 +09:00	mssql01.offsec.lan	4624	informational	Logon Type 3 - Network	User: admmig : Workstation: - : IP Addr
2021-05-26 22:02:27.155 +09:00	mssql01.offsec.lan	5145	medium	DCERPC SMB Spoolss Named Pipe	
2021-05-26 22:02:27.155 +09:00	mssql01.offsec.lan	5145	critical	CVE-2021-1675 Print Spooler Exploitation IPC Access	
2021-05-26 22:02:29.726 +09:00	mssql01.offsec.lan	4624	informational	Logon Type 3 - Network	User: admmig : Workstation: - : IP Addr
2021-05-26 22:02:29.734 +09:00	mssql01.offsec.lan	5145	medium	DCERPC SMB Spoolss Named Pipe	
2021-05-26 22:02:29.734 +09:00	mssql01.offsec.lan	5145	critical	CVE-2021-1675 Print Spooler Exploitation IPC Access	
2021-05-26 22:02:34.373 +09:00	mssql01.offsec.lan	4624	informational	Logon Type 3 - Network	User: admmig : Workstation: - : IP Addr
2021-05-26 22:02:34.373 +09:00	mssql01.offsec.lan	5145	medium	DCERPC SMB Spoolss Named Pipe	
2021-05-26 22:02:34.379 +09:00	mssql01.offsec.lan	4624	informational	Logon Type 3 - Network	User: admmig : Workstation: - : IP Addr
2021-05-26 22:02:34.379 +09:00	mssql01.offsec.lan	4624	informational	Logon Type 3 - Network	User: admmig : Workstation: - : IP Addr
2021-05-26 22:02:34.380 +09:00	mssql01.offsec.lan	4624	informational	Logon Type 3 - Network	User: admmig : Workstation: - : IP Addr
2021-05-27 05:24:46.570 +09:00	rootdc1.offsec.lan	4768	medium	Possible AS-REP Roasting	Possible AS-REP Roasting
2021-05-27 05:24:46.570 +09:00	rootdc1.offsec.lan	4768	informational	Kerberos TGT was requested	User: admin-test : Service: krbtgt : IP
2021-06-01 23:06:34.542 +09:00	fs01.offsec.lan	4720	medium	Local user account created	User: WADGUtilityAccount : SID:S-1-5-21-1081258321-3780
2021-06-01 23:08:21.225 +09:00	fs01.offsec.lan	4720	medium	Local user account created	User: elie : SID:S-1-5-21-1081258321-3780
2021-06-03 21:17:56.988 +09:00	fs01.offsec.lan	1102	high	Security log was cleared	User: admmig
2021-06-03 21:18:12.941 +09:00	fs01.offsec.lan	4672	informational	Admin Logon	User: admmig : LogonID: 0x322e5b7
2021-06-03 21:18:12.942 +09:00	fs01.offsec.lan	4624	informational	Logon Type 3 - Network	User: admmig : Workstation: - : IP Addr
2021-06-04 03:34:12.672 +09:00	fs01.offsec.lan	4104	high	Windows Firewall Profile Disabled	
2021-06-04 04:17:44.873 +09:00	fs01.offsec.lan	1102	high	Security log was cleared	User: admmig

Criticalアラートのフィルタリングとコンピュータごとのグルーピング

Computername ▲					
Line	Tag	Time	Eventid	Level	Alert
▼ =	■	UTC	■	= critical	■
▶ Computername: 01566s-win16-ir.threebeesco.com (Count: 1)					
▶ Computername: alice.insecurebank.local (Count: 3)					
△ Computername: DC1.insecurebank.local (Count: 18)					
5540	■	2019-03-26 06:28:45.026 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5539	■	2019-03-26 06:28:45.026 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5538	■	2019-03-26 06:28:45.026 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5537	■	2019-03-26 06:28:45.026 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5536	■	2019-03-26 06:28:45.025 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5535	■	2019-03-26 06:28:45.025 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5534	■	2019-03-26 06:28:45.025 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5533	■	2019-03-26 06:28:45.025 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5532	■	2019-03-26 06:28:45.025 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5531	■	2019-03-26 06:28:45.024 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5530	■	2019-03-26 06:28:45.024 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5529	■	2019-03-26 06:28:45.024 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5528	■	2019-03-26 06:28:45.023 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5527	■	2019-03-26 06:28:45.023 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5526	■	2019-03-26 06:28:45.023 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5525	■	2019-03-26 06:28:45.023 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5524	■	2019-03-26 06:28:45.022 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
5523	■	2019-03-26 06:28:45.022 +09:00	5136	critical	Powerview Add-DomainObjectAcl DCSync AD Extend Right
▶ Computername: DESKTOP-PIU87N6 (Count: 1)					

Elastic Stackダッシュボードでの解析



Top 10 Alerts							Top 10 Critical Alerts							Top 10 High Alerts						
Top values of RuleTitle		info	Cour	low	Coun	high	Cou	medium	Cou	critical	Cou	Count of records	Top values of RuleTitle	Count of records	Top values of RuleTitle	Count of records				
Network Share File Access		2,564	-	-	-	-	-	-	-	-	-	Mimikatz Use	33	Malicious Service Possibly Inst...	271					
Process Created		1,447	-	-	-	-	-	-	-	-	-	Powerview Add-DomainObjectAcl...	22	Suspicious Service Installed	257					
PwSh Scriptblock Log		1,024	-	-	-	-	-	-	-	-	-	Sticky Key Like Backdoor Usage	8	System Log File Cleared	97					
PwSh Pipeline Execution		680	-	-	-	-	-	-	-	-	-	Active Directory Replication from ...	6	Suspicious Remote Thread Crea...	94					
Network Share Access		433	-	-	-	-	-	-	-	-	-	EfsPotato Named Pipe	6	Accessing WinAPI in PowerShe...	93					
Other		2,058	223	831	594	41	3,564	1,418	154	108	39	WannaCry Ransomware	4	Relevant Anti-Virus Event	71					
Logon Failure - Wrong Password		-	3,564	-	-	-	-	-	-	-	-	CobaltStrike Service Installations	3	Security Log Cleared	66					
Suspicious Cmd Line_Possible LOLBIN ...		-	1,418	-	-	-	-	-	-	-	-	DNS Server Error Failed Loading t...	3	Process Created_Sysmon Alert	60					
Process Access		-	154	-	-	-	-	-	-	-	-	Dumpert Process Dumper	3	Disabling Windows Event Audit...	42					
Image Loaded_Sysmon Alert		-	108	-	-	-	-	-	-	-	-	LSASS Access from Non System ...	3	Malicious PowerShell Keywords	30					
Process Start From Suspicious Folder		-	39	-	-	-	-	-	-	-	-	Other	27	Other	562					

Hayabusa Discover (16622 documents)

Time	Computer	EventID	Level	MitreAttack	RuleTitle	Details
> 2022-02-19 17:35:16.328 +00:00	DESKTOP-TTEQ6PR	7	info	Persis Evas Pr ivEsc	Windows Spooler Service Suspicious Binary Load	-
> 2022-02-19 17:35:16.301 +00:00	DESKTOP-TTEQ6PR	11	info	-	File Created	Path: C:\Windows\System32\spool\drivers\x64\4\Test.dll Process: C:\Users\win10\Desktop\SpoolPool-main\SpoolPool.exe PID: 1232 PUUID: 0BDAA6306-2A54-6211-0B01-000000001000
> 2022-02-19 17:35:16.301 +00:00	DESKTOP-TTEQ6PR	11	medium	-	Rename Common File to DLL	-
> 2022-02-19 17:35:16.207 +00:00	DESKTOP-TTEQ6PR	1	info	-	Process Created	Cmd: "C:\Users\win10\Desktop\SpoolPool-main\SpoolPool.exe" -d11 C:\ProgramData\Test.dll Process: C:\Users\win10\Desktop\SpoolPool-main\SpoolPool.exe User: DESKTOP-TTEQ6PR\win10 Parent Cmd: "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -noexit -command Set-Location -LiteralPath C:\Users\win10\Desktop\SpoolPool-main L ID: 0x277ef PID: 1232 PUUID: 0BDAA6306-2A54-6211-0B01-000000001000
> 2022-02-19 17:35:16.207 +00:00	DESKTOP-TTEQ6PR	1	low	Exec	Process Start From Suspicious Folder	-
> 2022-02-16 19:37:20.934 +00:00	01566s-win16-ir.t hreebeesco.com	5145	info	Collect	Network Share File Access	User: samir Share Name: *\C\$ Share Path: \?\C:\ Path: Users\PSECURITY IP Addr: 172.16.66.36 LID: 0x567758

Timesketchでの解析

2019-05-08T02:10:43	<input type="checkbox"/> Active Directory Replication from Non Machine Account	User: Administrator ObjSvr: DS ObjName: %{c6faf700-bfe4-452a-a766-424f84c29583} OpType: Object Access HID: 0x0 LID: 0x40c6511	4662	DC1.insecurebank.local	Sec	T1003.006
2019-05-08T02:10:43	<input type="checkbox"/> Active Directory Replication from Non Machine Account	User: Administrator ObjSvr: DS ObjName: %{c6faf700-bfe4-452a-a766-424f84c29583} OpType: Object Access HID: 0x0 LID: 0x40c6511	4662	DC1.insecurebank.local	Sec	T1003.006
2019-05-08T02:10:43	<input type="checkbox"/> Active Directory Replication from Non Machine Account	User: Administrator ObjSvr: DS ObjName: %{c6faf700-bfe4-452a-a766-424f84c29583} OpType: Object Access HID: 0x0 LID: 0x40c6511	4662	DC1.insecurebank.local	Sec	T1003.006
4 days						
2019-05-12T12:52:43	<input type="checkbox"/> Meterpreter or Cobalt Strike Getsystem Service Installation	Svc: WinPwnage Path: %COMSPEC% /c ping -n 1 127.0.0.1 >nul && echo 'WinPwnage' > \\.\pipe\WinPwnagePipe Acct: LocalSystem StartType: demand start	7045	IEWIN7	Sys	T1134.001 : T1134.002
39 days						
2019-06-21T07:35:37	<input type="checkbox"/> Dumpert Process Dumper	Path: C:\Windows\Temp\dumpert.dmp Process: C:\Users\administrator\Desktop\x64\O utfleck-Dumpert.exe PID: 3572 PGUID: ECAD0485-88C9-5D0C-0000-0010348C1D00	11	alice.insecurebank.local	Sysmon	T1003.001

タイムライン結果のインポートと解析について

CSVのタイムラインをExcelやTimeline Explorerで分析する方法は[こちら](#)で紹介しています。

CSVのタイムラインをElastic Stackにインポートする方法は[こちら](#)で紹介しています。

CSVのタイムラインをTimesketchにインポートする方法は[こちら](#)で紹介しています。

jqによるJSON形式の結果の解析

JSON形式の結果をjqで解析する方法については、[こちら](#)を参照してください。

特徴＆機能

- クロスプラットフォーム対応: Windows, Linux, macOS。
- Rustで開発され、メモリセーフでハヤブサよりも高速です！
- マルチスレッド対応により、最大5倍のスピードアップを実現。
- フォレンジック調査やインシデントレスポンスのために、分析しやすいCSVタイムラインを作成します。
- 読みやすい/作成/編集可能なYMLベースのHayabusaルールで作成されたIoCシグネチャに基づくスレット。
- SigmaルールをHayabusaルールに変換するためのSigmaルールのサポートがされています。
- 現在、他の類似ツールに比べ最も多くのSigmaルールをサポートしており、カウントルール、新しい機能の|equalsfieldや|endswithfield等にも対応しています。
- イベントログの統計。(どのような種類のイベントがあるのかを把握し、ログ設定のチューニングに有効です。)

- 不良ルールやノイズの多いルールを除外するルールチューニング設定が可能です。
- MITRE ATT&CKとのマッピング (CSVの出力ファイルのみ)。
- ルールレベルのチューニング。
- イベントログから不審なユーザやファイルを素早く特定するためのピボットキーワードの一覧作成。
- 詳細な調査のために全フィールド情報の出力。
- 成功と失敗したユーザログオンの要約。
- [Velociraptor](#)と組み合わせた企業向けの広範囲なすべてのエンドポイントに対するスレットハンティングとDFIR。
- CSV、JSON、JSONL形式とHTML結果サマリの出力。
- 毎日のSigmaルール更新。
- JSON形式のログ入力にも対応。
- ログフィールドの正規化
- IPアドレスにGeolP (ASN、都市、国) 情報を付加することによるログエンリッチメント。

ダウンロード

[Releases](#)ページからHayabusaの安定したバージョンでコンパイルされたバイナリが含まれている最新版もしくはソースコードをダウンロードできます。

Gitクローン

以下のgit cloneコマンドでレポジトリをダウンロードし、ソースコードからコンパイルして使用することも可能です：

```
git clone https://github.com/Yamato-Security/hayabusa.git --recursive
```

注意： mainブランチは開発中のバージョンです。まだ正式にリリースされていない新機能が使えるかもしれないが、バグがある可能性もあるので、テスト版だと思って下さい。

*--recursiveをつけ忘れた場合、サブモジュールとして管理されているrulesフォルダ内のファイルはダウンロードされません。

git pull --recurse-submodulesコマンド、もしくは以下のコマンドでrulesフォルダを同期し、Hayabusaの最新のルールを更新することができます:

```
hayabusa.exe update-rules
```

アップデートが失敗した場合は、rulesフォルダの名前を変更してから、もう一回アップデートしてみて下さい。

注意： アップデートを実行する際に rules フォルダは hayabusa-rules レポジトリの最新のルールとコンフィグファイルに置き換えられます 既存ファイルへの修正はすべて上書きされますので、アップデート実行前に編集したファイルのバックアップをおすすめします。もし、level-tuningを行っ

ているのであれば、アップデート後にルールファイルの再調整をしてください **rules** フォルダ内に新しく追加したルールは、アップデート時に上書きもしくは削除は行われません。

アドバンス: ソースコードからのコンパイル（任意）

Rustがインストールされている場合、以下のコマンドでソースコードからコンパイルすることができます：

注意: `hayabusa`をコンパイルするためにはRust(`rustc`)が**1.66.0**以上であることが必要です。

```
cargo build --release
```

最新のunstable版はmainブランチから、最新の安定版は[Releases](#)ページからダウンロードできます。

以下のコマンドで定期的にRustをアップデートしてください：

```
rustup update stable
```

コンパイルされたバイナリは**target/release** フォルダ配下で作成されます。

Rustパッケージの更新

コンパイル前に最新のRust crateにアップデートすることで、最新のライブラリを利用することができます：

```
cargo update
```

アップデート後、何か不具合がありましたらお知らせください。

32ビットWindowsバイナリのクロスコンパイル

以下のコマンドで64ビットのWindows端末で32ビットのバイナリをクロスコンパイルできます：

```
rustup install stable-i686-pc-windows-msvc
rustup target add i686-pc-windows-msvc
rustup run stable-i686-pc-windows-msvc cargo build --release
```

注意: Rust の新しい安定版が出たときには必ず `rustup install stable-i686-pc-windows-msvc` を実行してください。`rustup update stable` はクロスコンパイル用のコンパイラを更新しないので、ビルドエラーが発生することがあります。

macOSでのコンパイルの注意点

`openssl`についてのコンパイルエラーが表示される場合は、[Homebrew](#)をインストールしてから、以下のパッケージをインストールする必要があります：

```
brew install pkg-config  
brew install openssl
```

Linuxでのコンパイルの注意点

opensslについてのコンパイルエラーが表示される場合は、以下のパッケージをインストールする必要があります。

Ubuntu系のディストロ:

```
sudo apt install libssl-dev
```

Fedora系のディストロ:

```
sudo yum install openssl-devel
```

LinuxのMUSLバイナリのクロスコンパイル

まず、Linux OSでターゲットをインストールします。

```
rustup install stable-x86_64-unknown-linux-musl  
rustup target add x86_64-unknown-linux-musl
```

以下のようにコンパイルします:

```
cargo build --release --target=x86_64-unknown-linux-musl
```

注意: Rust の新しい安定版が出たときには必ず `rustup install stable-x86_64-unknown-linux-musl` を実行してください。 `rustup update stable` はクロスコンパイル用のコンパイラを更新しないので、ビルドエラーが発生することがあります。

MUSLバイナリは `./target/x86_64-unknown-linux-musl/release/` ディレクトリ配下に作成されます。
MUSLバイナリはGNUバイナリより約15%遅いですが、より多くのLinuxバージョンとディストロで実行できます。

Hayabusaの実行

注意: アンチウィルス/EDRの誤検知と遅い初回実行

Hayabusa実行する際や、`.yml` ルールのダウンロードや実行時にルール内で `detection` に不審な PowerShell コマンドや `mimikatz` のようなキーワードが書かれている際に、アンチウィルスや EDR にブロックされる可能性があります。

誤検知のため、セキュリティ対策の製品がHayabusaを許可するように設定する必要があります。マルウェア感染が心配であれば、ソースコードを確認した上で、自分でバイナリをコンパイルして下さい。

Windows PC起動後の初回実行時に時間がかかる場合があります。これはWindows Defenderのリアルタイムスキャンが行われていることが原因です。リアルタイムスキャンを無効にするかHayabusaのディレクトリをアンチウィルススキャンから除外することでこの現象は解消しますが、設定を変える前にセキュリティリスクを十分ご考慮ください。

Windows

コマンドプロンプトやWindows Terminalから32ビットもしくは64ビットのWindowsバイナリをHayabusaのルートディレクトリから実行します。

Linux

まず、バイナリに実行権限を与える必要があります。

```
chmod +x ./hayabusa
```

次に、Hayabusaのルートディレクトリから実行します：

```
./hayabusa
```

macOS

まず、ターミナルやiTerm2からバイナリに実行権限を与える必要があります。

```
chmod +x ./hayabusa
```

次に、Hayabusaのルートディレクトリから実行してみてください：

```
./hayabusa
```

macOSの最新版では、以下のセキュリティ警告が出る可能性があります：



macOSの環境設定から「セキュリティとプライバシー」を開き、「一般」タブから「このまま許可」ボタンをクリックしてください。

その後、ターミナルからもう一回実行してみてください：

```
./hayabusa
```

以下の警告が出るので、「開く」をクリックしてください。



これで実行できるようになります。

主なコマンド

- `csv-timeline`: CSV形式のタイムラインを出力する。
- `json-timeline`: JSON/JSONL形式のタイムラインを出力する。
- `logon-summary`: ログオンイベントのサマリを出力する。
- `metrics`: イベントIDに基づくイベントの合計と割合の集計を出力する。
- `pivot-keywords-list`: ピボットする不審なキーワードのリストを作成する。
- `update-rules`: GitHubの[hayabusa-rules](#)リポジトリにある最新のルールに同期させる。
- `level-tuning`: アラート `level` のカスタムチューニング。
- `list-profiles`: 出力プロファイルの一覧表示。
- `set-default-profile`: デフォルトプロファイルを変更する。

使用方法

デフォルトのヘルプメニュー

Usage:

```
hayabusa.exe help <COMMAND>
hayabusa.exe <COMMAND> [OPTIONS]
```

Commands:

csv-timeline	CSV形式のタイムラインを出力
json-timeline	JSON/JSONL形式のタイムラインを出力
logon-summary	ログオンイベントのサマリを出力
metrics	イベントIDに基づくイベントの合計と割合の集計を出力
pivot-keywords-list	ピボットキーワードの一覧作成
update-rules	rulesフォルダをhayabusa-rulesのgithubリポジトリの最新版に更新する
level-tuning	ルールlevelのチューニング（デフォルト：./rules/config/level_tuning.txt）
set-default-profile	デフォルトの出力コンフィグを設定する
list-contributors	コントリビュータの一覧表示
list-profiles	出力プロファイルの一覧表示
help	コマンドに付随するオプションのヘルプを表示する

Options:

--no-color	カラーで出力しない
-q, --quiet	Quietモード：起動バナーを表示しない

csv-timelineコマンド

csv-timelineコマンドはイベントのフォレンジックタイムラインをCSV形式で作成します。

Usage: csv-timeline <INPUT> [OPTIONS]

Options:

-G, --GeoIP <MAXMIND-DB-DIR>	IPアドレスのGeoIP(ASN、都市、国)情報を追加する
-J, --JSON-input	.evtxファイルの代わりにJSON形式のログファイル(.jsonまたは.jsonl)をスキャンする
-Q, --quiet-errors	Quiet errorsモード：エラーログを保存しない
-c, --rules-config <DIR> ト: ./rules/config	ルールフォルダのコンフィグディレクトリ（デフォルト: ./rules/config）
-t, --threads <NUMBER> 値)	スレッド数（デフォルト：パフォーマンスに最適な数値）
-v, --verbose	詳細な情報を出力する

Output:

-H, --HTML-report <FILE>	HTML形式で詳細な結果を出力する（例: results.html）
-o, --output <FILE>	タイムラインを保存する（例: results.csv）
-p, --profile <PROFILE>	利用する出力プロファイル名を指定する

Input:

-d, --directory <DIR>	.evtxファイルを持つディレクトリのパス
-f, --file <FILE>	1つの.evtxファイルに対して解析を行う
-l, --live-analysis	ローカル端末のC:\Windows\System32\winevt\Logsフォルダを解析する

Advanced:

-r, --rules <DIR/FILE>	ルールファイルまたはルールファイルを持つディレクトリ (デフォルト: ./rules)
--target-file-ext <EVTX_FILE_EXT>	evtx以外の拡張子を解析対象に追加する。 (例1: evtx_data 例2: evtx1, evtx2)

Filtering:

-E, --EID-filter	速度を上げるために主なEIDだけスキャンする (コンフィグファイル: ./rules/config/target_event_IDs.txt)
--enable-deprecated-rules	Deprecatedルールを有効にする
-n, --enable-noisy-rules	Noisyルールを有効にする
-e, --exact-level <LEVEL>	特定のレベルだけスキャンする (informational, low, medium, high, critical)
--exclude-status <STATUS>	読み込み対象外とするルール内でのステータス (ex: experimental) (ex: stable, test)
-m, --min-level <LEVEL>	結果出力をするルールの最低レベル (デフォルト: informational)
--timeline-end <DATE>	解析対象とするイベントログの終了時刻 (例: "2022-02-22 23:59:59 +09:00")
--timeline-start <DATE>	解析対象とするイベントログの開始時刻 (例: "2020-02-22 00:00:00 +09:00")

Time Format:

--European-time	ヨーロッパ形式で日付と時刻を出力する (例: 22-02-2022 22:00:00.123 +02:00)
--ISO-8601	ISO-8601形式で日付と時刻を出力する (ex: 2022-02-22T10:10:10.1234567Z) (いつもUTC)
--RFC-2822	RFC 2822形式で日付と時刻を出力する (例: Fri, 22 Feb 2022 22:00:00 -0600)
--RFC-3339	RFC 3339形式で日付と時刻を出力する (例: 2022-02-22 22:00:00.123456-06:00)
--US-military-time	24時間制(ミリタリータイム)のアメリカ形式で日付と時刻を出力する (例: 02-22-2022 22:00:00.123 -06:00)
--US-time	アメリカ形式で日付と時刻を出力する (例: 02-22-2022 10:00:00.123 PM -06:00)
-U, --UTC	UTC形式で日付と時刻を出力する (デフォルト: 現地時間)

Display Settings:

--no-summary	結果概要を出力しない (多少速くなる)
-T, --visualize-timeline	イベント頻度タイムラインを出力する (ターミナルはUnicodeに対応する必要がある)

csv-timelineコマンドの使用例

- デフォルトのstandardプロファイルで1つのWindowsイベントログファイルに対してHayabusaを実行する:

```
hayabusa.exe csv-timeline -f eventlog.evtx
```

- **verbose**プロファイルで複数のWindowsイベントログファイルのあるsample-evtxディレクトリに対して、Hayabusaを実行する:

```
hayabusa.exe csv-timeline -d .\hayabusa-sample-evtx -p verbose
```

- 全てのフィールド情報も含めて1つのCSVファイルにエクスポートして、Excel、Timeline Explorer、Elastic Stack等でさらに分析することができる(注意: **super-verbose**プロファイルを使すると、出力するファイルのサイズがとても大きくなる!):

```
hayabusa.exe csv-timeline -d .\hayabusa-sample-evtx -o results.csv -p super-verbose
```

- EID(イベントID)フィルタを有効にし、タイムラインをJSON形式で保存する:

注意: EIDフィルタを有効にすると、私達のテストでは処理時間が約10~15%速くなりますが、アラートを見逃す可能性があります。

```
hayabusa.exe json-timeline -E -d .\hayabusa-sample-evtx -o results.json
```

- Hayabusaルールのみを実行する(デフォルトでは-r .\rulesにあるすべてのルールが利用される):

```
hayabusa.exe csv-timeline -d .\hayabusa-sample-evtx -r .\rules\hayabusa -o results.csv
```

- Windowsでデフォルトで有効になっているログに対してのみ、Hayabusaルールを実行する:

```
hayabusa.exe csv-timeline -d .\hayabusa-sample-evtx -r .\rules\hayabusa\builtin -o results.csv
```

- Sysmonログに対してのみHayabusaルールを実行する:

```
hayabusa.exe csv-timeline -d .\hayabusa-sample-evtx -r .\rules\hayabusa\sysmon -o results.csv
```

- Sigmaルールのみを実行する:

```
hayabusa.exe csv-timeline -d .\hayabusa-sample-evtx -r .\rules\sigma -o results.csv
```

- 廃棄(deprecated)されたルール(statusがdeprecatedになっているルール)とノイジールール(.\\rules\\config\\noisy_rules.txtにルールIDが書かれているルール)を有効にする:

注意: 最近、廃止されたルールはSigmaリポジトリで別のディレクトリに置かれるようになり、Hayabusaではもうデフォルトでは含まれないようになりました。従って、廃止されたルールを有効にする必要はないでしょう。

```
hayabusa.exe csv-timeline -d .\hayabusa-sample-evtx --enable-noisy-rules --enable-deprecated-rules -o results.csv
```

- ログオン情報を分析するルールのみを実行し、UTCタイムゾーンで出力する:

```
hayabusa.exe csv-timeline -d .\hayabusa-sample-evtx -r .\rules\hayabusa\builtin\Security\LogonLogoff\Logon -U -o results.csv
```

- 起動中のWindows端末上で実行し（Administrator権限が必要）、アラート（悪意のある可能性のある動作）のみを検知する:

```
hayabusa.exe csv-timeline -l -m low
```

- 詳細なメッセージを出力する(処理に時間がかかるファイル、パースエラー等を特定するのに便利):

```
hayabusa.exe csv-timeline -d .\hayabusa-sample-evtx -v
```

- Verbose出力の例:

```
Checking target evtx FilePath: "./hayabusa-sample-evtx/YamatoSecurity/T1027.004_0bfuscated Files or Information\u{a0}Compile After Delivery/sysmon.evtx"
1 / 509 [>-----] 0.20 % 1s
Checking target evtx FilePath: "./hayabusa-sample-evtx/YamatoSecurity/T1558.004_Steal or Forge Kerberos Tickets AS-REP Roasting/Security.evtx"
2 / 509 [>-----] 0.39 % 1s
Checking target evtx FilePath: "./hayabusa-sample-evtx/YamatoSecurity/T1558.003_Steal or Forge Kerberos Tickets\u{a0}Kerberoasting/Security.evtx"
```

```

3 / 509 [>-----] 0.59 % 1s
-----] 0.59 % 1s
Checking target evtx FilePath: "./hayabusa-sample-
evtx/YamatoSecurity/T1197_BITS_Jobs/Windows-BitsClient.evtx"
4 / 509 [=>-----] 0.79 % 1s
-----] 0.79 % 1s
Checking target evtx FilePath: "./hayabusa-sample-
evtx/YamatoSecurity/T1218.004_Signed Binary Proxy
Execution\{a0}InstallUtil/sysmon.evtx"
5 / 509 [=>-----] 0.98 % 1s
-----] 0.98 % 1s

```

- 結果を[Timesketch](#)にインポートできるCSV形式に保存する:

```
hayabusa.exe csv-timeline -d ./hayabusa-sample-evtx --RFC-3339 -o
timesketch-import.csv -p timesketch -U
```

- エラーログの出力をさせないようにする: デフォルトでは、Hayabusaはエラーメッセージをエラーログに保存します。エラーメッセージを保存したくない場合は、[-Q](#)を追加してください。

csv-timelineコマンドの設定ファイル

[./rules/config/channel_abbreviations.txt](#): チャンネル名とその略称のマッピング。

[./rules/config/default_details.txt](#): ルールに[details](#): 行が指定されていない場合に、どのようなデフォルトのフィールド情報 (%Details%フィールド)を出力するかを設定するファイルです。プロバイダー名とイベントIDを元に作成されます。

[./rules/config/eventkey_alias.txt](#): このファイルには、フィールドの短い名前のエイリアスと、元の長いフィールド名のマッピングがあります。

例:

```
InstanceId,Event.UserData.UMDFHostDeviceArrivalBegin.InstanceId
IntegrityLevel,Event.EventData.IntegrityLevel
IpAddress,Event.EventData.IpAddress
```

ここでフィールドが定義されていない場合、Hayabusaは自動的に[Event.EventData](#)にあるフィールドを使用します。

[./rules/config/exclude_rules.txt](#): このファイルには、使用から除外されるルールIDのリストがあります。通常は、あるルールが別のルールに置き換わったか、そもそもそのルールが使用できないことが原因です。ファイアウォールやIDSと同様に、シグネチャベースのツールは、自身の環境に合わせてチューニングする必要があるため、特定のルールを恒久的または一時的に除外する必要があるかもしれません。

[./rules/config/exclude_rules.txt](#)にルールID (例:[4fe151c2-ecf9-4fae-95ae-b88ec9c2fcfa6](#))を追加すると、不要なルールや使用できないルールを無視できます。

`./rules/config/noisy_rules.txt`: このファイルには、デフォルトでは無効になっているルールのIDが入っています。`-n, --enable-noisy-rules`オプションでノイジールールを有効にできます。これらのルールは通常、性質上ノイズが多いか、誤検出があるためです。

`./rules/config/target_event_IDs.txt`: EIDフィルターが有効な場合、このファイルで指定されたイベントIDのみがスキャンされます。デフォルトでは、Hayabusaはすべてのイベントをスキャンしますが、パフォーマンスを向上させたい場合は、`-E, --EID-filter`オプションを使用してください。これにより、通常10~25%の速度向上があります。

json-timelineコマンド

`json-timeline`コマンドは、JSONまたはJSONL形式でイベントのフォレンジックタイムラインを作成します。JSONLへの出力は、JSONよりも高速でファイルサイズも小さいので、結果をElastic Stack等の他のツールにインポートするだけなら、JSONLが理想です。テキストエディタで手動で解析する場合は、JSONの方が良いでしょう。CSV出力は小さいタイムライン(通常2GB以下)をExcelやTimeline Explorerのようなツールにインポートするのに適しています。JSONは、`jq`等のツールでデータ(大きな結果ファイルを含む)をより詳細に分析する場合に最適です。`Details`フィールドが分離されているので、分析が容易になるからです。(CSV出力では、すべてのイベントログのフィールドが1つの大きな`Details`カラムに入っています、データのソートなどが難しくなっています。)

Usage: `json-timeline <INPUT> [OPTIONS]`

Options:

<code>-G, --GeoIP <MAXMIND-DB-DIR></code>	IPアドレスのGeoIP(ASN、都市、国)情報を追加する
<code>-J, --JSON-input</code>	.evtxファイルの代わりにJSON形式のログファイル
(.jsonまたは.jsonl)をスキャンする	
<code>-Q, --quiet-errors</code>	Quiet errorsモード: エラーログを保存しない
<code>-c, --rules-config <DIR></code> ト: <code>./rules/config</code>	ルールフォルダのコンフィグディレクトリ (デフォル
<code>-t, --threads <NUMBER></code> 値)	スレッド数 (デフォルト: パフォーマンスに最適な数
<code>-v, --verbose</code>	詳細な情報を出力する

Output:

<code>-H, --HTML-report <FILE></code>	HTML形式で詳細な結果を出力する (例: <code>results.html</code>)
<code>-L, --JSONL-output</code>	タイムラインをJSONL形式で保存する (例: <code>-L -o results.jsonl</code>)
<code>-o, --output <FILE></code>	タイムラインを保存する (例: <code>results.json</code>)
<code>-p, --profile <PROFILE></code>	利用する出力プロファイル名を指定する

Input:

<code>-d, --directory <DIR></code>	.evtxファイルを持つディレクトリのパス
<code>-f, --file <FILE></code>	1つの.evtxファイルに対して解析を行う
<code>-l, --live-analysis</code> ダを解析する	ローカル端末のC:\Windows\System32\winevt\Logsフォル

Advanced:

<code>-r, --rules <DIR/FILE></code> ディレクトリ (デフォルト: <code>./rules</code>)	ルールファイルまたはルールファイルを持つ
<code>--target-file-ext <EVTX_FILE_EXT></code> (例1: evtx_data 例2: evtx1, evtx2)	evtix以外の拡張子を解析対象に追加する。

Filtering:

<code>-E, --EID-filter</code>	速度を上げるために主なEIDだけスキャンする（コンフィグファイル: <code>./rules/config/target_event_IDs.txt</code> ）
<code>--enable-deprecated-rules</code>	Deprecatedルールを有効にする
<code>-n, --enable-noisy-rules</code>	Noisyルールを有効にする
<code>-e, --exact-level <LEVEL></code>	特定のレベルだけスキャンする（ <code>informational, low, medium, high, critical</code> ）
<code>--exclude-status <STATUS></code>	読み込み対象外とするルール内でのステータス（ex: <code>experimental</code> ）（ex: <code>stable, test</code> ）
<code>-m, --min-level <LEVEL></code>	結果出力をするルールの最低レベル（デフォルト: <code>informational</code> ）
<code>--timeline-end <DATE></code>	解析対象とするイベントログの終了時刻（例: "2022-02-22 23:59:59 +09:00"）
<code>--timeline-start <DATE></code>	解析対象とするイベントログの開始時刻（例: "2020-02-22 00:00:00 +09:00"）

Time Format:

<code>--European-time</code>	ヨーロッパ形式で日付と時刻を出力する（例: 22-02-2022 22:00:00.123 +02:00）
<code>--ISO-8601</code>	ISO-8601形式で日付と時刻を出力する（ex: 2022-02-22T10:10:10.1234567Z）（いつもUTC）
<code>--RFC-2822</code>	RFC 2822形式で日付と時刻を出力する（例: Fri, 22 Feb 2022 22:00:00 -0600）
<code>--RFC-3339</code>	RFC 3339形式で日付と時刻を出力する（例: 2022-02-22 22:00:00.123456-06:00）
<code>--US-military-time</code>	24時間制（ミリタリータイム）のアメリカ形式で日付と時刻を出力する（例: 02-22-2022 22:00:00.123 -06:00）
<code>--US-time</code>	アメリカ形式で日付と時刻を出力する（例: 02-22-2022 10:00:00.123 PM -06:00）
<code>-U, --UTC</code>	UTC形式で日付と時刻を出力する（デフォルト: 現地時間）

Display Settings:

<code>--no-summary</code>	結果概要を出力しない（多少速くなる）
<code>-T, --visualize-timeline</code>	イベント頻度タイムラインを出力する（ターミナルはUnicodeに対応する必要がある）

json-timelineコマンドの使用例と設定ファイル

`json-timeline`のオプションと設定ファイルは、`csv-timeline`と同じですが、JSON形式で出力するための`-L, --JSONL-output`オプションが1つ追加されています。

logon-summaryコマンド

`logon-summary`コマンドを使うことでログオン情報の要約（ユーザ名、ログイン成功数、ログイン失敗数）の画面出力ができます。単体のevtxファイルを解析したい場合は`-f`オプションを利用してください。複数のevtxファイルを対象としたい場合は`-d`オプションを合わせて使うことでevtxファイルごとのログイン情報の要約を出力できます。

```
Usage: logon-summary <INPUT> [OPTIONS]
```

Options:

<code>-J, --JSON-input</code>	.evtxファイルの代わりにJSON形式のログファイル
-------------------------------	-----------------------------

```
(.jsonまたは.jsonl)をスキャンする
-Q, --quiet-errors
-c, --rules-config <DIR>
ト: ./rules/config)
-t, --threads <NUMBER>
値)
-v, --verbose

Input:
-d, --directory <DIR>
-f, --file <FILE>
-l, --live-analysis
フォルダを解析する

Advanced:
--target-file-ext <EVTX_FILE_EXT> evtx以外の拡張子を解析対象に追加する
(例1: evtx_data 例2: evtx1,evttx2)

Output:
-o, --output <FILE> ログオンサマリをCSV形式で保存する (例: logon-summary.csv)
```

logon-summaryコマンドの使用例

- ログオンサマリの出力: `hayabusa.exe logon-summary -f Security.evtx`
- ログオンサマリ結果を保存する: `hayabusa.exe logon-summary -d ..\logs -o logon-summary.csv`

metricsコマンド

`metrics`コマンドを使用すると、イベントIDの総数や割合をチャンネルごとに分けて表示することができます。

```
Usage: metrics <INPUT> [OPTIONS]

Options:
-J, --JSON-input
(.jsonまたは.jsonl)をスキャンする
-Q, --quiet-errors
-c, --rules-config <DIR>
ト: ./rules/config)
-t, --threads <NUMBER>
値)
-v, --verbose

Input:
-d, --directory <DIR>
-f, --file <FILE>
-l, --live-analysis
フォルダを解析する

Advanced:
--target-file-ext <EVTX_FILE_EXT> evtx以外の拡張子を解析対象に追加する。
```

(例1: evtx_data 例2: evtx1, evtx2)

Output:

-o, --output <FILE> イベントIDに基づくイベントの合計と割合の集計を出力する
(例: metrics.csv)

metricsコマンドの使用例

- 一つのファイルに対してイベントIDの統計情報を出力する: `hayabusa.exe metrics -f Security.evtx`
- ディレクトリに対してイベントIDの統計情報を出力する: `hayabusa.exe metrics -d ..\logs`
- 結果をCSVファイルに保存する: `hayabusa.exe metrics -f metrics.csv`

metricsコマンドの設定ファイル

チャンネル名、イベントID、イベントのタイトルは、`rules/config/channel_eid_info.txt`で定義されています。

例:

```
Channel,EventID,EventTitle
Microsoft-Windows-Sysmon/Operational,1,Process Creation.
Microsoft-Windows-Sysmon/Operational,2,File Creation Timestamp Changed.
(Possible Timestamping)
Microsoft-Windows-Sysmon/Operational,3,Network Connection.
Microsoft-Windows-Sysmon/Operational,4,Sysmon Service State Changed.
```

pivot-keywords-listコマンド

`pivot-keywords-list`コマンドを使用すると、異常なユーザ、ホスト名、プロセスなどを迅速に特定し、イベントを関連付けるための固有のピボットキーワードのリストを作成することができます。

重要: デフォルトでは、Hayabusaはすべてのイベント (informationalおよびそれ以上) から結果を返すので、`pivot-keywords-list`コマンドと`-m, --min-level`オプションを組み合わせることを強くお勧めします。 例えば、まず`-m critical`でcriticalアラートのみのキーワードを作成し、次に`-m high, -m medium`等々と続けていきます。 検索結果には、多くの通常のイベントと一致する共通のキーワードが含まれている可能性が高いので、検索結果を手動でチェックし、固有のキーワードのリストを1つのファイルに作成した後、`grep -f keywords.txt timeline.csv`といったコマンドで疑わしい活動のタイムラインを絞り込み作成することができます。

Usage: `pivot-keywords-list <INPUT> [OPTIONS]`

Options:

<code>-J, --JSON-input</code>	.evtxファイルの代わりにJSON形式のログファイル
(.jsonまたは.jsonl)をスキャンする	
<code>-Q, --quiet-errors</code>	Quiet errorsモード: エラーログを保存しない

-c, --rules-config <DIR>	ルールフォルダのコンフィグディレクトリ（デフォルト: ./rules/config）
-t, --threads <NUMBER>	スレッド数（デフォルト: パフォーマンスに最適な数値）
-v, --verbose	詳細な情報を出力する
Input:	
-d, --directory <DIR>	.evtxファイルを持つディレクトリのパス
-f, --file <FILE>	1つの.evtxファイルに対して解析を行う
-l, --live-analysis	ローカル端末のC:\Windows\System32\winevt\Logs
フォルダを解析する	
Advanced:	
--target-file-ext <EVTX_FILE_EXT>	evtx以外の拡張子を解析対象に追加する。 (例1: evtx_data 例2: evtx1,evtx2)
Output:	
-o, --output <FILE>	ピボットキーワードの一覧を複数ファイルに出力する（例: pivot-keywords.txt）
Filtering:	
-E, --EID-filter	速度を上げるために主なEIDだけスキャンする（コンフィグファイル: ./rules/config/target_event_IDs.txt）
--enable-deprecated-rules	Deprecatedルールを有効にする
-n, --enable-noisy-rules	Noisyルールを有効にする
-e, --exact-level <LEVEL>	特定のレベルだけスキャンする（informational, low, medium, high, critical）
--exclude-status <STATUS>	読み込み対象外とするルール内でのステータス（ex: experimental）（ex: stable, test）
-m, --min-level <LEVEL>	結果出力をするルールの最低レベル（デフォルト: informational）
--timeline-end <DATE>	解析対象とするイベントログの終了時刻（例: "2022-02-22 23:59:59 +09:00"）
--timeline-start <DATE>	解析対象とするイベントログの開始時刻（例: "2020-02-22 00:00:00 +09:00"）

pivot-keywords-listコマンドの使用例

- 重要なアラートからピボットキーワードのリストを作成し、その結果を保存します。（結果は、**keywords-Ip Addresses.txt**、**keywords-Users.txt**等に保存されます）：

```
hayabusa.exe pivot-keywords-list -d ./logs -m critical -o keywords
```

pivot-keywords-listの設定ファイル

検索キーワードは、**./config/pivot_keywords.txt**を編集することでカスタマイズすることができます。デフォルト設定は以下の通りです：

```
Users.SubjectUserName  
Users.TargetUserName  
Users.User  
Logon IDs.SubjectLogonId  
Logon IDs.TargetLogonId  
Workstation Names.WorkstationName  
Ip Addresses.IpAddress  
Processes.Image
```

フォーマットは、**キーワード名.フィールド名**です。例えば、**Users**のリストを作成する場合、Hayabusaは、**SubjectUserName**、**TargetUserName**、**User**フィールドにあるすべての値をリストアップします。

update-rulesコマンド

update-rulesコマンドは、**rules**フォルダをHayabusaルールのGitHubリポジトリと同期し、ルールと設定ファイルを更新します。

```
Usage: update-rules [OPTIONS]
```

Options:

```
--no-color  カラーで出力しない  
-q, --quiet    Quietモード: 起動バナーを表示しない
```

Advanced:

```
-r, --rules <DIR/FILE> ルールファイルまたはルールファイルを持つディレクトリ (デフォルト: ./rules)
```

update-rulesコマンドの使用例

普段は次のように実行します: **hayabusa.exe update-rules**

level-tuningコマンド

level-tuningコマンドを使用すると、環境に応じてリスクレベルを上げたり下げたりして、ルールのアラートレベルを調整できます。

```
Usage: level-tuning [OPTIONS]
```

Options:

```
-f, --file <FILE> ルールlevelのチューニング (デフォルト: ./rules/config/level_tuning.txt)  
--no-color  カラーで出力しない  
-q, --quiet    Quietモード: 起動バナーを表示しない
```

level-tuningコマンドの使用例

- 通常使用: `hayabusa.exe level-tuning`
- カスタム設定ファイルに基づくルールのアラートレベルの調整: `hayabusa.exe level-tuning -f my_level_tuning.txt`

level-tuningの設定ファイル

HayabubsaとSigmaのルール作成者は、アラートのリスクレベルを判定してルールを作成します。しかし、実際のリスクレベルは環境に応じて異なる場合があります。`./rules/config/level_tuning.txt`にルールを追加して `hayabusa.exe level-tuning` を実行すると、ルールファイル内の `level` 行が更新され、リスクレベルを調整することができます。ルールファイルが直接更新されますので、ご注意ください。

注意: `update-rules` を実行するたびに、アラートレベルが元の設定に上書きされるので、レベルを変更したい場合は、`update-rules` を実行した後に、`level-tuning` コマンドも実行する必要があります。

`./rules/config/level_tuning.txt` の一例:

```
id,new_level
00000000-0000-0000-0000-000000000000,informational # レベルチューニングのサンプル
```

この場合、ルールディレクトリ内の `id` が `00000000-0000-0000-0000-000000000000` のルールのアラート `level` が、`informational` に書き換えられます。設定可能なレベルは、`critical`、`high`、`medium`、`low`、`informational` です。

set-default-profile コマンド

```
Usage: set-default-profile [OPTIONS]
```

Options:

<code>-p, --profile <PROFILE></code>	利用する出力プロファイル名を指定する
<code>--no-color</code>	カラーで出力しない
<code>-q, --quiet</code>	Quietモード: 起動バナーを表示しない

list-profiles コマンド

```
Usage: list-profiles [OPTIONS]
```

Options:

<code>--no-color</code>	カラーで出力しない
<code>-q, --quiet</code>	Quietモード: 起動バナーを表示しない

アドバンス

GeolPのログエンリッチメント

無償のGeoLite2のジオロケーションデータで、SrcIP（ソースIPアドレス）フィールドとTgtIP（ターゲットIPアドレス）フィールドにGeoIP（ASN組織、都市、国）情報を追加することができます。

手順:

1. まずMaxMindのアカウントを[こちら](#)で登録してください。
 2. [ダウンロードページ](#)から3つの.mmdbファイルをダウンロードし、ディレクトリに保存してください。ファイル名は、GeoLite2-ASN.mmdb、GeoLite2-City.mmdb、GeoLite2-Country.mmdbであることをご確認ください。
 3. csv-timelineまたはjson-timelineコマンドを実行する際には、-Gオプションの後にMaxMindデータベースのあるディレクトリを追加してください。
- csv-timelineを使用すると、次の6つのカラムが追加で出力されます: SrcASN、SrcCity、SrcCountry、TgtASN、TgtCity、TgtCountry
 - json-timelineを使用すると、同じSrcASN、SrcCity、SrcCountry、TgtASN、TgtCity、TgtCountryフィールドがDetailsオブジェクトに追加されますが、情報を含む場合のみとなります。
 - SrcIPまたはTgtIPがlocalhost (127.0.0.1、::1等々)の場合、SrcASNまたはTgtASNは、Localとして出力されます。
 - SrcIPまたはTgtIPがプライベートIPアドレス (10.0.0.0/8、fe80::/10等々)の場合、SrcASNまたはTgtASNは、Privateとして出力されます。

GeoIPの設定ファイル

GeoIPデータベースで検索される送信元と送信先のIPアドレスを含むフィールド名は、rules/config/geoip_field_mapping.yamlで定義されています。必要であれば、このリストに追加することができます。また、このファイルには、どのイベントからIPアドレス情報を抽出するかを決定するフィルタセクションもあります。

GeoIPデータベースの自動アップデート

MaxMind GeoIP データベースは、2週間ごとに更新されます。これらのデータベースを自動的に更新するために、[こちら](#)からMaxMindのgeoipupdateのツールをインストールすることができます。

macOSでの手順:

1. brew install geoipupdate
2. /usr/local/etc/GeoIP.confを編集する: MaxMindのウェブサイトにログインした後に作成したAccountIDとLicenseKeyを入れる。EditionIDsの行に、EditionIDs GeoLite2-ASN GeoLite2-City GeoLite2-Countryとあることを確認する。
3. geoipupdateを実行する。
4. GeoIP情報を追加する場合は、-G /usr/local/var/GeoIPを追加する。

Windowsでの手順:

1. ReleasesページからWindowsバイナリの最新版(例: geoipupdate_4.10.0_windows_amd64.zip)をダウンロードする。
2. \ProgramData\MaxMind\GeoIPUpdate\GeoIP.confを編集する: MaxMindのウェブサイトにログインした後に作成したAccountIDとLicenseKeyを入れる。EditionIDsの行に、EditionIDs

`GeoLite2-ASN` `GeoLite2-City` `GeoLite2-Country`とあることを確認する。

3. `geoipupdate`を実行する。

サンプルevtxファイルでHayabusaをテストする

Hayabusaをテストしたり、新しいルールを作成したりするためのサンプルevtxファイルをいくつか提供しています:
<https://github.com/Yamato-Security/Hayabusa-sample-evtx>

以下のコマンドで、サンプルのevtxファイルを新しいサブディレクトリ `hayabusa-sample-evtx` にダウンロードすることができます:

```
git clone https://github.com/Yamato-Security/hayabusa-sample-evtx.git
```

HayabusaのCSVとJSON/L出力

出力プロファイル

Hayabusaの`config/profiles.yaml`設定ファイルでは、5つのプロファイルが定義されています:

1. `minimal`
2. `standard` (デフォルト)
3. `verbose`
4. `all-field-info`
5. `all-field-info-verbose`
6. `super-verbose`
7. `timesketch-minimal`
8. `timesketch-verbose`

このファイルを編集することで、簡単に独自のプロファイルをカスタマイズしたり、追加したりすることができます。`set-default-profile -P <profile>`オプションでデフォルトのプロファイルを変更することもできます。利用可能なプロファイルとそのフィールド情報を表示するには、`csv-timeline --list-profiles`オプションを使用してください。

1. `minimal`プロファイルの出力

`%Timestamp%, %Computer%, %Channel%, %EventID%, %Level%, %RuleTitle%, %Details%`

2. `standard`プロファイルの出力

`%Timestamp%, %Computer%, %Channel%, %EventID%, %Level%, %RecordID%, %RuleTitle%, %Details%`

3. `verbose`プロファイルの出力

`%Timestamp%, %Computer%, %Channel%, %EventID%, %Level%, %MitreTactics%, %MitreTags%, %OtherTags%, %RecordID%, %RuleTitle%, %Details%, %RuleFile%, %EvtxFile%`

4. `all-field-info`プロファイルの出力

最小限の`details`情報を出力する代わりに、イベントにあるすべてのEventDataフィールド情報(`%AllFieldInfo%`)が出力されます。

```
%Timestamp%, %Computer%, %Channel%, %EventID%, %Level%, %RecordID%, %RuleTitle%,
%AllFieldInfo%, %RuleFile%, %EvtxFile%
```

5. `all-field-info-verbose`プロファイルの出力

`all-field-info`とタグ情報が出力されます。

```
%Timestamp%, %Computer%, %Channel%, %EventID%, %Level%, %MitreTactics%, %MitreTags%,
%OtherTags%, %RecordID%, %RuleTitle%, %AllFieldInfo%, %RuleFile%, %EvtxFile%
```

6. `super-verbose`プロファイルの出力

`verbose`プロファイルで出力される情報とイベントにあるすべてのEventDataフィールド情報(`%AllFieldInfo%`)の両方が出力されます。 (注意: 出力ファイルサイズは約2倍になります!)

```
%Timestamp%, %Computer%, %Channel%, %Provider%, %EventID%, %Level%, %MitreTactics%,
%MitreTags%, %OtherTags%, %RecordID%, %RuleTitle%, %RuleAuthor%, %RuleCreationDate%,
%RuleModifiedDate%, %Status%, %Details%, %RuleFile%, %EvtxFile%, %AllFieldInfo%
```

7. `timesketch-minimal`プロファイルの出力

Timesketchにインポートできる`verbose`プロファイル。

```
%Timestamp%, hayabusa, %RuleTitle%, %Computer%, %Channel%, %EventID%, %Level%,
%MitreTactics%, %MitreTags%, %OtherTags%, %RecordID%, %Details%, %RuleFile%, %EvtxFile%
```

8. `timesketch-verbose`プロファイルの出力

Timesketchにインポートできる`verbose`プロファイル。 (注意: 出力ファイルサイズは約2倍になります!)

```
%Timestamp%, hayabusa, %RuleTitle%, %Computer%, %Channel%, %EventID%, %Level%,
%MitreTactics%, %MitreTags%, %OtherTags%, %RecordID%, %Details%, %RuleFile%,
%EvtxFile%, %AllFieldInfo%
```

プロファイルの比較

以下のベンチマークは、2018年製のマックブックプロ上で7.5GBのEVTXデータに対して実施されました。

プロファイル	処理時間	結果のファイルサイズ
minimal	16分18秒	690 MB
standard	16分23秒	710 MB
verbose	17分	990 MB
timesketch-minimal	17分	1015 MB

プロファイル	処理時間	結果のファイルサイズ
all-field-info-verbose	16分50秒	1.6 GB
super-verbose	17分12秒	2.1 GB

Profile Field Aliases

エイリアス名	Hayabusaの出力情報
%Timestamp%	デフォルトではYYYY-MM-DD HH:mm:ss.sss +hh:mm形式になっている。イベントログの<Event><System><TimeCreated SystemTime>フィールドから来ている。デフォルトのタイムゾーンはローカルのタイムゾーンになるが、--UTCオプションでUTCに変更することができる。
%Computer%	イベントログの<Event><System><Computer>フィールド。
%Channel%	ログ名。イベントログの<Event><System><EventID>フィールド。
%EventID%	イベントログの<Event><System><EventID>フィールド。
%Level%	YML検知ルールのlevelフィールド。(例: informational、low、medium、high、critical)
%MitreTactics%	MITRE ATT&CKの戦術 (例: Initial Access、Lateral Movement等々)
%MitreTags%	MITRE ATT&CKの戦術以外の情報。attack.g(グループ)、attack.t(技術)、attack.s(ソフトウェア)の情報を出力する。
%OtherTags%	YML検知ルールのtagsフィールドからMitreTactics、MitreTags以外のキーワードを出力する。
%RecordID%	<Event><System><EventRecordID>フィールドのイベントレコードID。
%RuleTitle%	YML検知ルールのtitleフィールド。
%Details%	YML検知ルールのdetailsフィールドから来ていますが、このフィールドはHayabusaルールにしかありません。このフィールドはアラートとイベントに関する追加情報を提供し、ログのフィールドから有用なデータを抽出することができます。イベントキーのマッピングが間違っている場合、もしくはフィールドが存在しない場合で抽出ができなかった箇所はn/a (not available)と記載されます。YML検知ルールにdetailsフィールドが存在しない時のdetailsのメッセージを./rules/config/default_details.txtで設定できます。default_details.txtではProvider Name、EventID、detailsの組み合わせで設定することができます。default_details.txt`やYML検知ルールに対応するルールが記載されていない場合はすべてのフィールド情報を出力します。
%AllFieldInfo%	すべてのフィールド情報。
%RuleFile%	アラートまたはイベントを生成した検知ルールのファイル名。
%EvtxFile%	アラートまたはイベントを起こしたevttxファイルへのパス。
%RuleAuthor%	YML検知ルールのauthorフィールド。

エイリアス名	Hayabusaの出力情報
%RuleCreationDate%	YML検知ルールの <code>date</code> フィールド。
%RuleModifiedDate%	YML検知ルールの <code>modified</code> フィールド。
%Status%	YML検知ルールの <code>status</code> フィールド。
%RuleID%	YML検知ルールの <code>id</code> フィールド。
%Provider%	<code><Event><System><Provider></code> フィールド内の <code>Name</code> 属性。
%RenderedMessage%	WEC機能で転送されたイベントログの <code><Event><RenderingInfo><Message></code> フィールド。

これらのエイリアスは、出力プロファイルで使用することができます。また、他の[イベントキーエイリアス](#)を定義し、他のフィールドを出力することもできます。

Levelの省略

簡潔に出力するために `level` を以下のように省略し出力しています。

- `crit: critical`
- `high: high`
- `med : medium`
- `low : low`
- `info: informational`

MITRE ATT&CK戦術の省略

簡潔に出力するために MITRE ATT&CK の 戦術を以下のように省略しています。

`./config/mitre_tactics.txt` の 設定ファイルで自由に編集できます。

- `Recon` : Reconnaissance (偵察)
- `ResDev` : Resource Development (リソース開発)
- `InitAccess` : Initial Access (初期アクセス)
- `Exec` : Execution (実行)
- `Persis` : Persistence (永続化)
- `PrivEsc` : Privilege Escalation (権限昇格)
- `Evas` : Defense Evasion (防御回避)
- `CredAccess` : Credential Access (認証情報アクセス)
- `Disc` : Discovery (探索)
- `LatMov` : Lateral Movement (横展開)
- `Collect` : Collection (収集)
- `C2` : Command and Control (遠隔操作)
- `Exfil` : Exfiltration (持ち出し)
- `Impact` : Impact (影響)

Channel情報の省略

簡潔に表示するためにChannelの表示を以下のように省略しています。

./rules/config/channel_abbreviations.txt の設定ファイルで自由に編集できます。

- App : Application
- AppLocker : Microsoft-Windows-AppLocker/*
- BitsCli : Microsoft-Windows-Bits-Client/Operational
- CodeInteg : Microsoft-Windows-CodeIntegrity/Operational
- Defender : Microsoft-Windows-Windows Defender/Operational
- DHCP-Svr : Microsoft-Windows-DHCP-Server/Operational
- DNS-Svr : DNS Server
- DvrFmwk : Microsoft-Windows-DriverFrameworks-UserMode/Operational
- Exchange : MSExchange Management
- Firewall : Microsoft-Windows-Windows Firewall With Advanced Security/Firewall
- KeyMgtSvc : Key Management Service
- LDAP-Cli : Microsoft-Windows-LDAP-Client/Debug
- NTLM : Microsoft-Windows-NTLM/Operational
- OpenSSH : OpenSSH/Operational
- PrintAdm : Microsoft-Windows-PrintService/Admin
- PrintOp : Microsoft-Windows-PrintService/Operational
- PwSh : Microsoft-Windows-PowerShell/Operational
- PwShClassic : Windows PowerShell
- RDP-Client : Microsoft-Windows-TerminalServices-RDPClient/Operational
- Sec : Security
- SecMitig : Microsoft-Windows-Security-Mitigations/*
- SmbCliSec : Microsoft-Windows-SmbClient/Security
- SvcBusCli : Microsoft-ServiceBus-Client
- Sys : System
- Sysmon : Microsoft-Windows-Sysmon/Operational
- TaskSch : Microsoft-Windows-TaskScheduler/Operational
- WinRM : Microsoft-Windows-WinRM/Operational
- WMI : Microsoft-Windows-WMI-Activity/Operational

その他の省略

できるだけ簡潔にするために、以下の略語を使用しています:

- Acct -> Account
- Addr -> Address
- Auth -> Authentication
- Cli -> Client
- Chan -> Channel
- Cmd -> Command
- Cnt -> Count
- Comp -> Computer
- Conn -> Connection/Connected
- Creds -> Credentials

- **Crit** -> Critical
- **Disconnect** -> Disconnection/Disconnected
- **Dir** -> Directory
- **Drv** -> Driver
- **Dst** -> Destination
- **EID** -> Event ID
- **Err** -> Error
- **Exec** -> Execution
- **FW** -> Firewall
- **Grp** -> Group
- **Img** -> Image
- **Inj** -> Injection
- **Krb** -> Kerberos
- **LID** -> Logon ID
- **Med** -> Medium
- **Net** -> Network
- **Obj** -> Object
- **Op** -> Operational/Operation
- **Proto** -> Protocol
- **PW** -> Password
- **Reconn** -> Reconnection
- **Req** -> Request
- **Rsp** -> Response
- **Sess** -> Session
- **Sig** -> Signature
- **Susp** -> Suspicious
- **Src** -> Source
- **Svc** -> Service
- **Svr** -> Server
- **Temp** -> Temporary
- **Term** -> Termination/Terminated
- **Tkt** -> Ticket
- **Tgt** -> Target
- **Unkwn** -> Unknown
- **Usr** -> User
- **Perm** -> Permanent
- **Pkg** -> Package
- **Priv** -> Privilege
- **Proc** -> Process
- **PID** -> Process ID
- **PGUID** -> Process GUID (Global Unique ID)
- **Ver** -> Version

プログレスバー

プログレス・バーは、複数のevtxファイルに対してのみ機能します。 解析したevtxファイルの数と割合をリアルタイムで表示します。

カラー出力

Hayabusaの結果はlevel毎に文字色が変わります。`./config/level_color.txt`の値を変更することで文字色を変えることができます。形式はlevel名,(6桁のRGBのカラーhex)です。カラー出力をしないようにしたい場合は`--no-color`オプションをご利用ください。

結果のサマリ (Results Summary)

元々のイベント数、検知したイベント数、データ削減の統計、検知数情報、最多検知日、最多検知端末名、最多アラート等の情報がスキャン後に出力されます。

イベント頻度タイムライン

`-T, --visualize-timeline`オプションを使うことで、検知したイベントの数が5以上の時、頻度のタイムライン(スパークライン)を画面に出力します。マーカーの数は最大10個です。デフォルトのCommand PromptとPowerShell Promptでは文字化けがあるので、Windows TerminalやiTerm2等のターミナルをご利用ください。

Hayabusaルール

Hayabusa検知ルールはSigmaのようなYML形式で記述され、rulesディレクトリに入っています。

<https://github.com/Yamato-Security/hayabusa-rules>のレポジトリで管理しているので、ルールのissueやpull requestはhayabusaのレポジトリではなく、ルールレポジトリへお願いします。

ルールの作成方法については、[hayabusa-rulesレポジトリのREADME](#)をお読みください。

hayabusa-rulesレポジトリにあるすべてのルールは、rulesフォルダに配置する必要があります。levelがinformationのルールはイベントとみなされ、low以上はアラートとみなされます。

Hayabusaルールのディレクトリ構造は、2つのディレクトリに分かれています:

- `builtin`: Windowsの組み込み機能で生成できるログ。
- `sysmon`: `sysmon`によって生成されるログ。

ルールはさらにログタイプ（例：Security、Systemなど）によってディレクトリに分けられ、次の形式で名前が付けられます。

現在のルールをご確認いただき、新規作成時のテンプレートとして、また検知ロジックの確認用としてご利用ください。

Hayabusa v.s. 変換されたSigmaルール

Sigmaルールは、最初にHayabusaルール形式に変換する必要があります。変換のやり方は[ここで説明されています](#)。Hayabusaルールは`|contains|all, 1 of selection*, all of selection*`、Rust正規表現クレートでは機能しない正規表現を使用するルールをデフォルトで対応していないため、コンバータが必要です。殆どのルールはSigmaルールと互換性があるので、Sigmaルールのようにその他のSIEM形式に変換できます。Hayabusaルールは、Windowsのイベントログ解析専用に設計されており、以下のようない点があります:

1. ログの有用なフィールドのみから抽出された追加情報を表示するための`details`フィールドを追加しています。
2. Hayabusaルールはすべてサンプルログに対してテストされ、検知することが確認されています。

変換処理のバグ、サポートされていない機能、実装の違い(正規表現など)により、一部のSigmaルールは意図したとおりに動作しない可能性があります。

3. Sigmaルール仕様にない集計式(例：[|equalsfield](#)、[|endswithfield](#))の利用。

制限事項: 私たちの知る限り、Hayabusa はオープンソースの Windows イベントログ解析ツールの中でSigmaルールを最も多くサポートしていますが、まだサポートされていないルールもあります。

1. Sigmaルール仕様の[count](#)以外の集計式。
2. [|near](#)、[|base64offset|contains](#)を使用するルール。

その他のWindowsイベントログ解析ツールおよび関連リソース

「すべてを統治する1つのツール」というものではなく、それぞれにメリットがあるため、これらの他の優れたツールやプロジェクトをチェックして、どれが気に入ったかを確認することをお勧めします。

- [APT-Hunter](#) - Pythonで開発された攻撃検知ツール。
- [Awesome Event IDs](#) - フォレンジック調査とインシデント対応に役立つイベントIDのリソース。
- [Chainsaw](#) - Rustで開発されたSigmaベースの攻撃検知ツール。
- [DeepBlueCLI](#) - [Eric Conrad](#) によってPowershellで開発された攻撃検知ツール。
- [Epagneul](#) - Windowsイベントログの可視化ツール。
- [EventList](#) - [Miriam Wiesner](#)によるセキュリティベースラインの有効なイベントIDをMITRE ATT&CKにマッピングするPowerShellツール。
- [MITRE ATT&CKとWindowイベントログIDのマッピング](#) - 作者：[Michel de CREVOISIER](#)
- [EvtxECmd](#) - [Eric Zimmerman](#)によるEvtxパーサー。
- [EVTXtract](#) - 未使用領域やメモリダンプからEVTXファイルを復元するツール。
- [EvtxToElk](#) - Elastic StackにEvtxデータを送信するPythonツール。
- [EVTX ATTACK Samples](#) - [SBousseaden](#) によるEVTX攻撃サンプルイベントログファイル。
- [EVTX-to-MITRE-Attack](#) - [Michel de CREVOISIER](#)によるATT&CKにマッピングされたEVTX攻撃サンプルログのレポジトリ。
- [EVTX parser](#) - [@OBenamram](#) によって書かれた、Hayabusaが使用しているRustライブラリ。
- [Grafiki](#) - SysmonとPowerShellログの可視化ツール。
- [LogonTracer](#) - [JPCERTCC](#) による、横方向の動きを検知するためにログオンを視覚化するグラフィカルなインターフェース。
- [NSA Windows Event Monitoring Guidance](#) - NSAのWindowsイベントログ監視ガイド。
- [RustyBlue](#) - 大和セキュリティによるDeepBlueCLIのRust版。
- [Sigma](#) - コミュニティベースの汎用SIEMルール。
- [SOF-ELK](#) - [Phil Hagen](#) によるDFIR解析用のElastic Stack VM。
- [so-import-evtx](#) - evtxファイルをSecurityOnionにインポートするツール。
- [SysmonTools](#) - Sysmonの設定とオフライン可視化ツール。
- [Timeline Explorer](#) - [Eric Zimmerman](#) による最高のCSVタイムラインアナライザ。
- [Windows Event Log Analysis - Analyst Reference](#) - Forward DefenseのSteve AnsonによるWindowsイベントログ解析の参考資料。
- [WELA \(Windows Event Log Analyzer\)](#) - [Yamato Security](#)によるWindowsイベントログ解析のマルチツール。
- [Zircolite](#) - Pythonで書かれたSigmaベースの攻撃検知ツール。

Windowsイベントログ設定のススメ

Windows機での悪性な活動を検知する為には、デフォルトのログ設定を改善することが必要です。どのようなログ設定を有効にする必要があるのか、また、自動的に適切な設定を有効にするためのスクリプトを、別のプロジェクトとして作成しました: <https://github.com/Yamato-Security/EnableWindowsLogSettings>

以下のサイトを閲覧することもおすすめします。:

- JSCU-NL (Joint Sigt Cyber Unit Netherlands) Logging Essentials
- ACSC (Australian Cyber Security Centre) Logging and Fowarding Guide
- Malware Archaeology Cheat Sheets

Sysmon関係のプロジェクト

フォレンジックに有用な証拠を作り、高い精度で検知をさせるためには、sysmonをインストールする必要があります。以下のサイトを参考に設定することをおすすめします。:

- Sysmon Modular
- TrustedSec Sysmon Community Guide
- SwiftOnSecurityのSysmon設定ファイル
- Neo23x0によるSwiftOnSecurityのSysmon設定ファイルのフォーク
- ion-stormによるSwiftOnSecurityのSysmon設定ファイルのフォーク

コミュニティによるドキュメンテーション

英語

- 2022/06/19 [VelociraptorチュートリアルとHayabusaの統合方法](#) by Eric Capuano
- 2022/01/24 [Hayabusa結果をneo4jで可視化する方法](#) by Matthew Seyer (@forensic_matt)

日本語

- 2022/01/22 [Hayabusa結果をElastic Stackで可視化する方法](#) by @kzzzo2
- 2021/12/31 [Windowsイベントログ解析ツール「Hayabusa」を使ってみる](#) by itiB (@itiB_S144)
- 2021/12/27 [Hayabusaの中身](#) by Kazuminn (@k47_um1n)

貢献

どのような形でも構いませんので、ご協力を願いします。 プルリクエスト、ルール作成、evtxログのサンプルなどがベストですが、機能リクエスト、バグの通知なども大歓迎です。

少なくとも、私たちのツールを気に入っていただけなら、GitHubで星を付けて、あなたのサポートを表明してください。

バグの報告

見つけたバグを[こちら](#)でご連絡ください。 報告されたバグを喜んで修正します！

Hayabusaルールの問題点（誤検出、バグ等々）を発見された方は、hayabusa-rulesのGitHubの[Issues](#)ページにご報告ください。

Sigmaルールの問題点（誤検出、バグ等々）を発見された方は、上流のSigmaHQ GitHubの[Issues](#)ページにご報告ください。

ライセンス

Hayabusaは[GPLv3](#)で公開され、すべてのルールは[Detection Rule License \(DRL\) 1.1](#)で公開されています。

Twitter

@SecurityYamatoでHayabusa、ルール更新、その他の大和セキュリティツール等々について情報を提供しています。