Hayabusa v1.2.1 Release

Zach Mathis (@yamatosecurity)

May 13th

Agenda

- Self-Introduction
- Motivation for Hayabusa
- About Hayabusa
- Demos
- Q&A

Self-Introduction

- Zach Mathis (@yamatosecurity)
- Hacking since late 90’s.
- Infosec professional based in Kobe, Japan since 2006.
- Red team/blue team services for fortune 500 Japanese corporations.
- SANS Instructor 2016~.
- Founder of Yamato Security.

About Yamato Security

- Free hands-on security training in Japan since 2012.
- Topics wide ranging from penetration testing, reverse engineering, DFIR, CTF contests, etc... all around the country (Kobe, Fukuoka, Kagoshima, Sapporo, Tokyo).
- Largest hands-on security community in Japan with over 1600 registered members.
- https://yamatosecurity.connpass.com/

About Yamato Security

- Now releasing free OSS tools and resources in English as well:
- https://github.com/Yamato-Security

Hayabusa is...

- “a sigma-based threat hunting and fast forensics timeline generator for Windows event logs written in Rust.”
  - Fast, multi-threaded, cross-platform, memory safe, no dependencies (can do processing on endpoints without installing anything)!
- https://github.com/Yamato-Security/hayabusa

Hayabusa is brought to you by:

- Akira Nishikawa (@nishikawaakira): Previous lead developer, core hayabusa rule support, etc...
- DustInDark (@hitenkoku): Core developer (too many contributions to list up)
- Garigariganzy (@garigariganzy31): Developer, event ID statistics implementation, etc...
- ItiB (@itiB_S144) : Core developer, sigmac hayabusa backend, rule creation, etc...
- James Takai / hachiyone(@hach1yon): Current lead developer, tokio multi-threading, sigma aggregation logic, sigmac backend, rule creation, sigma count implementation etc…
- Kazuminn (@k47_um1n): Core Developer
- Tsubokku (@ytsuboi0322): Translations
- Yusuke Matsui (@apt773): AD hacking working group leader, rule testing, documentation, research, support, etc...
- Zach Mathis (@yamatosecurity, Yamato Security Founder): Project leader, tool and concept design, rule creation and tuning, etc…

Motivation for Windows event log analysis

- Traditionally a tedious and painful process:
  - Event Viewer is not ideal for checking many logs.
  - Hundreds of logs separated out.
  - Logs are often cryptic and hard to understand.
  - Levels assigned to events are not so helpful...
  - 90%+ is noise.
  - Default logging is terrible. (Default size: 20MB, no process creation information, PwSh, WMI, etc...)
    - Detection Capability = Data Source + Detection Rules

Motivation for Windows event log analysis

- However, if you properly configure event logging with Group Policy, security baselines, etc... and install and configure Sysmon on endpoints you can get great visibility and opportunities to detect attackers (for FREE!).
- Out of scope for this talk:
  Recommendations: JSCU-NL, ACSC, Malware Archaeology Cheat Sheets, Sysmon Modular.
  Links: https://github.com/Yamato-Security/hayabusa#windows-logging-recommendations

Motivation for Windows event log analysis

- Even without proper logging configuration and sysmon, at least some useful Windows event logs are present on all Windows machines making them a very important DFIR artifact.
- => If you do Windows DFIR, you need to spend the time to learn how to interpret event logs, understand which logs are important, etc...

Hayabusa's Main Goal

- To turn Windows event analysis into a fun and easy(ier) process.
- Audience:
  Novice system administrators to DFIR professionals.
- Two main features:
  1. Threat Hunting (with Sigma and Hayabusa rules)
     → quickly identify malicious activity. (seconds ~ minutes)
  2. Fast forensics timeline generator (CSV)
     → only provide useful information for investigations.
     (focus on noise reduction)

Threat Hunting with Hayabusa

- Majority of rules come from Sigma (2200+). https://github.com/SigmaHQ/sigma
  - Greatest sigma support out of OSS tools. (AFAIK)
- Hayabusa rules are Sigma-compatible with a focus on Windows event log detection. (※There are some extra features not found in Sigma. (details, |equalsfield, etc...)
- 125+ Hayabusa rules and steadily growing.
- Separate repo: https://github.com/Yamato-Security/hayabusa-rules

Threat Hunting with Hayabusa

- Detection rules are simple YML files:
- Hidden user account detection example:

```
detection:
    selection:
        Channel: Security
        EventID: 4720
        TargetUserName|endswith: "$"
condition: selection
```

Threat Hunting with Hayabusa

- Most rules are just searching for certain keywords.
- When there are many false positives, add a filter: section for your allowlist and add and not filter in the condition.
- Should allow most CSIRT teams to easily create rules for IoCs (Indicators of Compromise) in their incidents to perform scoping.

Fast Forensics Timeline with Hayabusa

- By default, results will be outputted to screen but outputting to CSV is recommended due to the size.
- Analyze the CSV results with:
  Excel (Not recommended, very slow, no grouping, etc...)
  Timeline Explorer (Recommended for easy setup)
  Elastic Stack (Recommended for advanced users)
- https://github.com/Yamato-Security/hayabusa/tree/main/doc

Sample .evtx files

- https://github.com/Yamato-Security/hayabusa-sample-evtx
  - Repository of repositories...
  - DeepBlueCLI (Eric Conrad)
  - EVTX-ATTACK-SAMPLES (Samir @sbousseaden)
  - EVTX-to-MITRE-Attack (Michel de CREVOISIER)
  - Yamato Security
- About 700 .evtx files (160MB) with various attack evidence.

Demo Time!

- Recommended:
  - Windows Terminal for Windows, iTerm2 for macOS
  - Extra-wide monitor!
- Download the latest release .zip (v1.2.1) from https://github.com/Yamato-Security/hayabusa/releases
- Includes needed rules and binaries for Windows (32 & 64 bit), macOS (Intel and Arm) and 64-bit Linux.

Most used options

- No options or -h (--help): Show help
- -u (--update-rules): Update rules.
- -d (--directory): Scan a directory with .evtx files
- -f (--filepath): Scan one .evtx file.
- -o (--output): Output results to CSV.
- -c (--color): Output in different colors accordingly to level. (True Color terminal (Windows Terminal, iTerm2, etc...) required.)

Hayabusa rules' "details" field

- Output format:
  Time | Computer | Channel | EID | Level | Alert Title | Alert Details
- When saving to CSV, MITRE ATT&CK tags, rule path and file path are also saved.
- By default, only fields defined in the details field of the Hayabusa rule will be displayed as there is usually too much information (and noise).
- Currently, sigma rules will not display field information in the details column but hopefully in the next release.

Details Abbreviations

Addr -> Address

Auth -> Authentication

Cmd -> Command

Dst -> Destination

LID -> Logon ID

Src -> Source

Svr -> Server

Svc -> Service

Tgt -> Target

PID -> Process ID

PGUID -> Process GUID (Global Unique ID)

Most used options

- -m (--min-level): Set minimal level to low, medium, high or critical. Default is informational (→ everything).
- -U (--utc): Display in UTC.
- -l (--live-analysis): Perform live analysis on Windows. Requires Administrator privileges.
  Same as -d C:\Windows\System32\winevt\Logs

Other options

- -r (--rules): Specify the rules directory or single rule file.
- -C (--config): Specify a custom rules config directory.
- -D (--enable-deprecated-rules)
- -n (--enable-noisy-rules): (./rules/config/noisy_rules.txt)
- -t (--thread-number): (Default is optimal number. Lower to reduce CPU usage.)
- -v (--verbose): Great for debugging!
- -q (--quiet): Do not display launch banner.
- -Q (--quiet-errors): Do no save error logs (./logs)

Other options

- --start-timeline 'Start time of the event logs to load. (Example: '2018/11/28 12:00:00 +09:00')'
- --end-timeline 'End time of the event logs to load. (Example: '2018/11/28 12:00:00 +09:00')'
- --rfc-2822 'Output date and time in RFC 2822 format. (Example: Mon, 07 Aug 2006 12:34:56 -0600)'
- --rfc-3339 'Output date and time in RFC 3339 format. (Example: 2006-08-07T12:34:56.485214 -06:00)'

New options in version 1.2.1 thanks to:

- DustInDark (@hitenkoku): Update rule output, -C / --config, bug fixes, code improvements, MITRE ATT&CK tags, etc...
- ItiB (@itiB_S144) : --level-tuning option, sigma converter update.
- James Takai / hachiyone(@hach1yon): |equalsfield aggregator, -F / --full-data option, performance tuning, etc...
- Kazuminn (@k47_um1n): -p (--pivot-keywords-list) option.
- Garigariganzy (@garigariganzy31): EID Statistics Update.

New options in version 1.2.1

- -F (--full-data): Output all field information.
- -s (--statistics): Not a new feature, but highly updated.
  Understand at a high layer what events are in your .evtx files.
- -p (--pivot-keywords-list)
  Will create a list of unique keywords for you to pivot on:
  LIDs(Logon IDs), Computer/User names, IP Addresses, Processes, etc
  (./config/pivot_keywords.txt)
  Combine with the -m option for the best results! -o to output to files.
  Example: -m high
  It will require manual checking afterwards! Use the pivot list to narrow down your timeline. (Example: grep -f keywords.txt timeline.csv)

New options in version 1.2.1

- --level-tuning <file> (default: ./rules/config/level_tuning.txt)
- We will provide our recommended tuning file so please run this option after updating rules if you like.
- All detection tools require tuning to your environment! Just need to specify the rule ID and level: fdb62a13-9a81-4e5c-a38f-ea93a16f6d7c,medium
- The level: portion of the rule file will be overwritten.

Future plans

- Hayabusa VQL artifact for Velociraptor for enterprise-wide threat hunting.
- Details information for majority of events.
- Alerting capability to Slack, Elastic, etc...
- Improve Elastic Stack field parsing.
- More rules, improved tuning/output/accuracy, etc...
- Daily sigma rule updates.

Contributions

- Contributions to the Rust code, Hayabusa rule creations, testing, feedback, etc... are highly appreciated!
- More documentation at:
  https://github.com/Yamato-Security/hayabusa/blob/main/README.md
  https://github.com/Yamato-Security/hayabusa-rules/blob/main/README.md
- Submit bugs to the Hayabusa repository:
  https://github.com/Yamato-Security/hayabusa/issues/new/choose
- Submit rules to the Hayabusa-rules repository:
  https://github.com/Yamato-Security/hayabusa-rules
  ※ Only about half of the evtx files on https://github.com/Yamato-Security/hayabusa-sample-evtx can be detected!

Advice on using Hayabusa

- Start with critical alerts (-m critical) and work your way down.
- Pivot on keywords you find.
- Learn how to filter and group with Timeline Explorer and Elastic Stack.
- You will most likely still get noise and false-positives but simple filtering can cut the noise out quickly!
- Studying the Hayabusa rules is free training on what logs and fields to look for when doing DFIR investigations!

Thank you!

- Any questions, comments, etc... ?

HAYABUSA