

# Detection Engineer v1.6



Daniel Taylor & Matthew Iverson

## Daily Tasks

- Add IOCs from Intel
- Develop 1 T-Code Rule
  - APT + Data Source = Rule
- Trim alerts
  - **Suricata, Sigma, Yara** to less than 50 each per on SecO
  - **Spl reports** less than 50 each on Splunk
  - IOC Dashboard
    - Run against virustotal for **IPs, Domains, Urls, and Sha256** that populate
- Note CVEs on network
  - Run `github CVE-XXXX-XXXX POC` on google
    - Generate SPL T-Code Rule if you see something populate

## 1. Threat Intel

Know your industry, recent attacks, your partners and who is attacking you so you can better predict which threat will attack you. This way you can develop better monitoring mechanisms to defend your monitor your network.

- a. Know the APTs that Affects you
- b. Affects your partners
- c. Affects your industry
- d. Recent Vulnerabilities

## 2. Threat Modeling for Detection Engineering

- a. Understand how adversaries will attack your environment and prioritize detections accordingly.
- b. Key Components
  - i. Attack Surface Analysis
    - 1. Identify all possible entry points attackers may use (Endpoints, Networks, Applications, Cloud, OT, ICS).
  - ii. Threat Prioritization
    - 1. Rank threats based on likelihood and impact to focus detection efforts on the most critical attack paths.
  - iii. Adversary Emulation
    - 1. Use frameworks like MITRE ATT&CK to map attacker behavior to real-world scenarios.
- c. Tools to Use
  - i. MITRE ATT&CK Navigator
  - ii. Crowdstrike
  - iii. Malpedia
  - iv. Threat Intelligence Reports (Mandiant, Microsoft, CISA, etc.)
  - v. CTI Feeds (VirusTotal, AlienVault, Abuse.ch)
  - vi. MITRE Engage for Active Defense

## 3. Creating IOC Doc

- a. Upload all iocs found to ioc doc
  - i. Md5
  - ii. Sha256
  - iii. Domain
  - iv. Ip

- b. Deduplicate all duplicated by going in the top of google sheets
  - i. Highlight the whole chart for each sheet in worksheet
  - ii. Data > data cleanup > remove duplicates
    1. Remove from column a
- c. Upload to virustotal collection

## 4: Detection Sources & IOC Tooling (Practical Workflow)

- Pull detections from CTI sources (e.g., **Mandiant**, **Microsoft**, **CISA**) and/or OSINT sources
- If malware or network activity matches a **TTP of an APT**, create **YARA** or **SIGMA** rules using **YarGen** or manually

### Relevant Tools:

- **Zeek**: `intel.dat` for IOC ingestion
- **YARA Scanners**: **LOKI**, **THOR**, **Dissect's YARA scanner** (Includes dead disk analysis)
- **Hayabusa** and **Chainsaw**: for Windows event log parsing and analysis

## 4. Virustotal

- a. Finding what are current t-codes
- b. Entering current CVEs into virustotal
- c. Grabbing any IOCs that are more than 1/3 of anti-virus vendors
- d. Look for in the wild exploits and over 20 detections

```
<NAME> have:itw p:20+
```

A different approach would be to instead look for malicious files that are not detected by existing Sigma rules, since they can uncover novel methodologies and provide new opportunities for detection creation.

```
p:5+ have:behavior fs:30d+ not have:sigma
```

we'd want to focus on files uploaded in the last month that haven't triggered any Sigma rules. This gives us a good starting point for building new detection rules.

```
p:5+ (sandbox_name:"CAPE Sandbox" or sandbox_name:"Zenbox") fs:30d+ not have:sigma
```

Specifically, we will look for samples with zero critical, high or medium alerts - and no more than two low severity ones.

```
p:5+ have:behavior fs:30d+ sigma_critical:0 sigma_high:0 sigma_medium:0  
sigma_low:2-
```

- e. Look for correlated iocs

## 5. Verifying IOCs

- a. Verify against <https://any.run/malware-trends/><NAME>
- b. Watch 3 different malware on anyrun

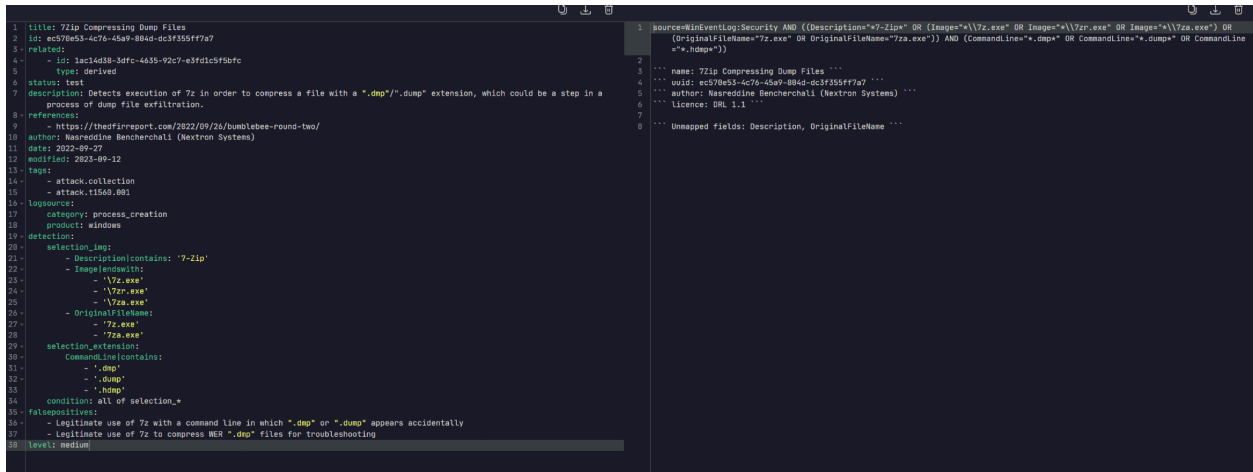
## 6. Rules

- a. Yara
  - i. File log detection
    - 1. <https://github.com/InQuest/awesome-yara>
    - 2. <https://yaraify.abuse.ch/scan/>
    - 3. <https://yarahq.github.io/>
    - 4. [Malpedia](#)
- b. Splunk
  - i. SPL host or network log detection
    - 1. [Splunk detections](#)
      - a. Excellent resource to grab splunk rules for T-codes
    - 2. [Mitre analytics](#)
      - a. Prewritten rules from MITRE to detect the threats
    - 3. [Litmus test](#)
    - 4. [Lolbas](#)
      - a. Create rules to tools that are used by APTs affecting you
- c. Elastic
  - i. [Elastic detections](#)
- d. Sysmon
  - i. [https://www.youtube.com/watch?v=kESndPO5Fig&list=PLqyUgadpThTKCbCY\\_4Kjp2P0p915yb1iN](https://www.youtube.com/watch?v=kESndPO5Fig&list=PLqyUgadpThTKCbCY_4Kjp2P0p915yb1iN)
- e. Suricata
  - i. Network log detection
    - 1. <https://github.com/satta/awesome-suricata>
- f. [Sigma](#)
  - i. Mostly host or network log detection

## 7. Create Rules

### a. Convert rules from sigma to SPL or KQL

#### i. [Uncoder](#)



```
1 title: 7Zip Compressing Dump Files
2 id: ec578e53-4c76-45a9-884d-dc3f355ff7a7
3 related:
4   - id: Inc16d38-3afc-4435-92c7-e3fduc5f5bfc
5     type: derived
6 status: test
7 description: Detects execution of 7z in order to compress a file with a ".dump"/".dump" extension, which could be a step in a
8               process of dump file exfiltration.
9 references:
10  - https://threatreport.com/2022/09/26/bumblebee-round-two/
11 author: Nearedline Benchercall (Neutron Systems)
12 date: 2022-09-27
13 modified: 2023-09-12
14 tags:
15   - attack.collection
16   - attack.t1563.081
17 logsource:
18   category: process_creation
19   product: windows
20 detection:
21   selection_log:
22     - Description|contains: '7z-Zip'
23     - Image|endswith:
24       - '\7z.exe'
25       - '\7zr.exe'
26     - OriginalFileName:
27       - '7z.exe'
28       - '7za.exe'
29   selection_extensions:
30     CommandLine|contains:
31       - '.dump'
32       - '.dump'
33       - '.dump'
34   condition: all of selection,*
35 falsepositives:
36   - Legitimate use of 7z with a command line in which ".dump" or ".dump" appears accidentally
37   - Legitimate use of 7z to compress NER ".dump" files for troubleshooting
38 level: medium
```

```
1 source=WinEventLog:Security AND ((Description="7z-Zip" OR (Image="*\\7z.exe" OR Image="*\\7zr.exe" OR Image="*\\7za.exe") OR
2   (OriginalFileName="7z.exe" OR OriginalFileName="7za.exe"))) AND (CommandLine=".dump" OR CommandLine=".dump" OR CommandLine
3   =".dump")
4
5 ''' name: 7Zip Compressing Dump Files '''
6 ''' uid: ec578e53-4c76-45a9-884d-dc3f355ff7a7 '''
7 ''' author: Nearedline Benchercall (Neutron Systems) '''
8 ''' license: OMA 1.1 '''
9
10 ''' Unmapped fields: Description, OriginalFileName '''
```

This will allow you convert SIGMA rules to elastic or splunk rules to easily convert to your SIEM.

#### ii. [Sigma2Splunk](#)

#### iii. [Custom IOC Converter](#)

1. This allows Detection engineers to grab and sort many Threat Intel IOC documents into one mass import for splunk

## 8. Rule Uniformity

- a. Add all fields needed

```
`indextime` `sysmon` <SEARCH>
| eval hash_sha256= lower(hash_sha256),
  hunting_trigger="",
  mitre_category="Defense_Evasion",
  mitre_technique="Obfuscated Files or Information",
  mitre_technique_id="T####",
  mitre_subtechnique="",
  mitre_subtechnique_id="T####.###",
  apt="",
  mitre_link="https://attack.mitre.org/techniques/T####/",
  creator="Cpl Iverson",
  last_tested="",
  upload_date="FIRSTDATE",
  last_modify_date="CURRENTDATE",
  mitre_version="v16",
  priority=""
| `process_create_whitelist`
| eval indextime = _indextime
| convert ctime(indextime)
| table _time indextime event_description hash_sha256 host_fqdn user_name
  original_file_name process_path process_guid process_parent_path process_id
  process_parent_id process_command_line process_parent_command_line
  process_parent_guid mitre_category mitre_technique mitre_technique_id hunting_trigger
  mitre_subtechnique mitre_subtechnique_id apt mitre_link creator upload_date
  last_modify_date mitre_version priority
| collect `jarvis_index`
```

### Permissions

- Display For: App
- Read: Everyone

- Write: admin

## Schedule

- Schedule: Run on Cron Schedule
- \*/15 \* \* \* \* (every 15 minutes)
- Time Range: Since date time
- Schedule Priority: Default
- Schedule Window: Auto

## Macros

- jarvis\_index: index=jarvis
- indextime: \_index\_earliest=-15m@m AND \_index\_latest=now
- sysmon: index=windows (source=WinEventLog:Microsoft-Windows-Sysmon/Operational)
- windows: index=windows

## 9. Refine Rules/ Remove Rules

- Check whether rule is hitting over 500 - if hitting over 500 remove and reassess
- If under 500 refine by seeing what processes, parent processes and items it is hitting on

## 10. Test Rules

- Have TNE say what codes they will hit
- Track what mitre t-codes you currently have
- Test
- See what alerts tracked
- Add the rules you missed
- Work with TNE to find areas where logs are detecting what they are doing
  - Create Rules
    - Use multiple sourcetypes
    - Use AI to help create rules
    - Specify based on timeframe
    - Data enrich
- repeat

Testing your rules is important to verify they are picking up the problems you are running into. Otherwise you do not know if your rules are finding the enemy.

- [Atomic red team](#)



- i. [Caldera](#)
- j. [APT Simulator](#)
- k. [Dumpster Fire](#)
- l. [Fire Drill](#)
- m. [Splunk Attack Range](#)
- n. [Sysmon Simulator](#)
- o. [Test my NIDs](#)

## 11. Update Rules

- a. Check [Att&ck Sync](#)
- b. Check t-codes that changed
- c. Change mitre\_technique="TECHNIQUE NAME", mitre\_technique\_id="T####", mitre\_subtechnique="", mitre\_subtechnique\_id="T####.###", apt="", mitre\_link="", last\_modify\_date="CURRENTDATE",
- d. Check all fields are the same

## 12. Back ups

- a. Yara - SecO - /nsm/so/rules/yara/
- b. Suricata - SecO - /opt/so/rules/nids/
- c. Reports for Jarvis - SPLUNK -  
/opt/splunk/etc/local/Arc\_Reactor\_app/jarvis.dashboard/savedsearches
- d. Whitelists - SPLUNK - /opt/splunk/etc/local/Arc\_Reactor\_app/lookups  
\*whitelist\*
- e. IOCs
  - i. Ip.csv - SPLUNK - /opt/splunk/etc/local/Arc\_Reactor\_app/lookup
  - ii. Md5.csv - SPLUNK - /opt/splunk/etc/local/Arc\_Reactor\_app/lookup
  - iii. Sha256.csv - SPLUNK - /opt/splunk/etc/local/Arc\_Reactor\_app/lookup
  - iv. Domain.csv - SPLUNK - /opt/splunk/etc/local/Arc\_Reactor\_app/lookup
  - v. ipv6.csv - SPLUNK - /opt/splunk/etc/local/Arc\_Reactor\_app/lookup

# References

1. <https://github.com/SigmaHQ/sigma>
2. <https://lolbas-project.github.io/>
3. <https://attack.mitre.org/datasources/>
4. <https://research.splunk.com/detections/>
5. [https://github.com/Kirtar22/Litmus\\_Test](https://github.com/Kirtar22/Litmus_Test)
6. <https://github.com/elastic/detection-rules>
7. <https://github.com/elastic/detection-rules>
8. <https://uncoder.io/>
9. <https://github.com/P4T12ICK/Sigma2SplunkAlert>
10. <https://github.com/redcanaryco/atomic-red-team>
11. <https://caldera.mitre.org/>
12. <https://github.com/NextronSystems/APTSimulator>
13. <https://github.com/TryCatchHCF/DumpsterFire>
14. <https://github.com/TryCatchHCF/DumpsterFire>
15. <https://github.com/FourCoreLabs/firedrill>
16. [https://github.com/splunk/attack\\_range](https://github.com/splunk/attack_range)
17. <https://github.com/ScarredMonk/SysmonSimulator>
18. <https://github.com/3CORESec/testmynids.org>
19. <https://cyb3rops.medium.com/about-detection-engineering-44d39e0755f0>
20. <https://www.sans.org/blog/purple-teaming-threat-informed-detection-engineering/>
21. <https://medium.com/@dylanhwilliams/what-makes-a-good-detection-dd6a3b373860>
22. <https://www.mitre.org/sites/default/files/2021-11/16-3713-finding-cyber-threats-with-attack-based-analytics.pdf>
23. [https://www.dragos.com/wp-content/uploads/The\\_Four\\_Types-of\\_Threat\\_Detection.pdf](https://www.dragos.com/wp-content/uploads/The_Four_Types-of_Threat_Detection.pdf)
24. <https://socprime.com/blog/sigma-rules-the-beginners-guide>
25. [https://sigmahq.streamlit.app/Create\\_A\\_New\\_Sigma\\_Rule](https://sigmahq.streamlit.app/Create_A_New_Sigma_Rule)
26. [https://code.europa.eu/groups/ec-digit-s2/opentide/-/wikis/uploads/e66f0f311d758449b350ff4f96105898/OpenTIDE\\_White\\_Paper.pdf](https://code.europa.eu/groups/ec-digit-s2/opentide/-/wikis/uploads/e66f0f311d758449b350ff4f96105898/OpenTIDE_White_Paper.pdf)
27. <https://www.tidalcyber.com/blog/detecting-and-simulating-recent-apt-persistence-methods-with-community-resources>

28. <https://medium.com/anton-on-security/detection-engineering-is-painful-and-it-should-nt-be-part-1-3641d8740458>
29. <https://medium.com/anton-on-security/build-for-detection-engineering-and-alerting-will-improve-part-3-dbd433516f95>
30. <https://posts.specterops.io/on-detection/home>
31. <https://medium.com/snowflake/threat-detection-maturity-framework-23bbb74db2bc>
32. <https://d4rkci3h3r.medium.com/establishing-a-detection-engineering-program-from-the-ground-up-b170811166c>
33. <https://detect.fyi/>
34. <https://github.com/infosecB/awesome-detection-engineering>
35. <https://github.com/0x4D31/awesome-threat-detection>
36. <https://sigmahq.io/>
37. <https://strontic.github.io/xcyclopedia/>

## Version Changes

- 1.0 create or intel, rules and testing
- 1.1 added rule refinement and rule deletion
- 1.2 update where to store backups and how alerts are created
- 1.3 add splunk and more references
- 1.4 add virustotal and CTI sections
- 1.5 update to include any.run