Suricata community style guide

A collaborative document to collect style guidelines from the community of rule writers

Rule format

Overall

- Keep rule direction and variables simple
  - No bidirectional rules, bidirectional rules can produce unexpected results. It is better to use 2 rules
    - Example: `$EXTERNAL_NET any -> any any` and `$HOME_NET any -> any any` **not** `alert http any any <-> any any`
  - Avoid using `any any -> any any`, it's better to create multiple rules if we expect multi direction, specifically stating INBOUND or OUTBOUND depending on the direction of the rule.
- Avoid using `packet_data;` if possible
  - `packet_data;`resets the inspection pointer, resulting in confiusing and disjointed logic
- Avoid creating `byte_test;` only rules, they perform very badly
- Avoid creating rules without `content:` keywords, they also perform poorly
- Avoid using the `priority` keyword
  - it overrides the operator's ability to tune priority for their specific environment via `classification.conf`
- Avoid sticky buffer naming convention
  - Pre suri5 buffer naming convention is complicated (sticky vs modifier)
  - Example: `http.header;` over `http_header;`
- Avoid inventing network variables, port variables or classtypes
  - suricata errors may surprise unsuspecting users
  - Example: avoid `classtype:newbadthing;`
  - Example: avoid `alert tcp $CLOUDFLARE_IP any -> $MY_NAS $SYNOLOGY_PORTS`
- Assert app-layer-protocol in `alert [app-layer-protocol]`, not in rule body
  - Example: `alert http ...` not `alert tcp ... app-layer-protocol:http;`
- Use lowercase `A-F` in hex
  - Example: `content:"User-Agent|3a 20|Patp|ca fe|py";` not `content:"User-Agent|3A 20|Patp|CA FE|py";`
- Use `fast_pattern;` where ambiguity exists, or to clarify intent
  - Suricata will choose the longest `content:"match";` not the most unique, uniqueness is **far** more important
- Rules for DNS queries should have a source host variable of `$HOME_NET` and a destination host variable of `any`
  - This ensures that queries in environments with and without local resolvers are covered

Whitespace & escaping

- Do not use spaces unless they are required
  - Allows for simpler text searching for example: `grep flowbits:set` vs `grep -P 'flowbits\s*:\s*set`
  - Example: `content:"User-Agent|3a 20|Patp|22 27|py";` or another way `"options:value;"` and not `"option: value;"` or `"option:value ;"`
- Escape `[\x3a\x3b\x20\x22\x27\x7b\x7c\x5c\x2f\x60\x24\x28\x29]` characters in content and PCRE
  - use `\x20` for literal space, `\s` for spaces, newlines, tabs within pcre
  - In content: use `|3b|` for `;`
  - Example: `content:"User-Agent|3a 20|Patp|22 27|py";`
- Use single space between entries
  - Example: `options:value; options:value;` not `options:value;options:value;`
- Use whitespace between bytes in content for easier eyeball parsing
  - Example: `content:"|c0 ff ee ba be|";` not `content:"|face10adbabe|";`
- Escape using hex encoding not \
  - Example: `content:"C|3a 5c|Windows|5c|system32|5c|";` not `content:"C|3a|\\Windows\\system32\\";`
- Escape using hex encoding for pcre
  - Example: `pcre:"/C\x3a\x5cWindows\x5csystem32\x5c/";` not `pcre:"/C:\/Windows\/system32\//";`

Keyword Order

- Rule order is `msg.*detection_logic.*reference.*classtype.*sid.*rev.*metadata`
- Port negations appear first, use brackets if number of items > 1
  - (e.g. `$EXTERNAL_NET [!8000:9000,9000:]`)
- Content keywords modified by `offset` & `depth` keywords appear first (& only once)
- `flow` follows `msg:"...";`if it is used
- Stream and flow keywords (`stream.size`, `flow.age`, etc) go after the `flow` keyword and before any buffers
- `flowbits` keywords should be after the `flow` keyword and before any content buffers/detection logic
- `urilen` should placed after the stream and flow keywords, but before any other buffer keywords
- Basic rule order is: `buffer`, `content`, `pointer movement`, `fast_pattern`, `nocase`, `isdataat/startswith/endswith`
  - Example: `http.header; content:"patpoopy"; depth:8; fast_pattern; nocase; isdataat:!2,relative;`
- Inline `threshold` keywords should be placed after all detection logic, before `reference/sid/rev/metadata`
- `bsize` occurs immediately after the buffer declaration and before any content matches

    - Example: `http.user_agent; bsize:6; content:"foobar"`
- Transformations occur immediately after the buffer declaration and before any content matches
    - Example: `dns.query; dotprefix; content:".google.com";`

Msg field

- Msg format is: `RULESET CATEGORY malware/product/protocol NAME [verbs] [date]`
    - For malware include the architecture/OS/platform in the signature message (ex. `Win32/malfamily`, `Win64/malfamily`, `ELF/malfamily`, `OSX/malfamily`, `PS/malfamily`)
- Avoid using the words `possible` and `unknown`, it's OK to make stuff up if need be
- Do not list author/team, use `metadata` instead
- Dates are ISO format
    - Example: `2017-11-03`
    - Use date sparingly for things that may change soon
- Use `CnC` for Command and Control/C2/etc
    - Example: `MSIL/Patpoopy CnC Check-in`
- Use filetype in malware name
    - Example: `Go/MSIL/ELF64/MSIL/JS/Win32/DOS/Amiga/C64/Plan9`
- Defang domain names by using a space *before* the label separator to avoid accidental information leaks
    - Example: `Observed Malicious Win32/Badhombre DNS Query (tromf .mx)`
        - `patpoo .py`
- Method (`M[0-9]`)
    - Use when detecting several behaviors of the same malware
    - Example: `Yowza Ransomware CnC Checkin M1`, `Yowza Ransomware CnC Checkin M2`
    - If there's another similar rule with no number already, give it a number
- Avoid Unicode graphemes, ASCII only. Unicode graphemes break import to srcfire

Flow, flowbits, xbits

- Write flow state before direction
    - Example: `flow:established,to_server;` not `flow:to_server,established;`
- Use flow (`to_server`|`to_client`) and not (`from_client`|`from_server`)
    - Example: `flow:established,to_server;`
- Use flowbits naming convention: `RULESET.description.flowbit`
    - Example: `ET.descriptive.flowbit` -> ET OPEN
    - Example: `ETPRO.descriptive.flowbit` -> ETPRO
    - Always use ET OPEN naming convention for noalert flowbits
    - Use noalert after flowbit (?:un)?set

    * Example: `flowbits:set,ET.descriptive.flowbit; flowbits:noalert;`
- Use xbits naming convention: `RULESET.description`
  - `ET.descriptive.flowbit ET.descriptive` -> ET OPEN
  - `ET.descriptive.flowbit ETPRO.descriptive` -> ETPRO
  - Explicitly set xbits expire value
    * Example: `xbits:isset,ET.badgum,track ip_src,expire 60;`

PCRE

- use non-capturing parens in pcre unless using the value later in the rule
  - Example: `pcre:"/unnamed(?:capture|group)/";` NOT `pcre:"/oops(capture|group)/";`
- use named variables instead of \1 \2 \3 in pcre
  - Example: `pcre:"/^(?P&lt;guid>[a-z0-9]{8}-[a-z0-9]{4}-[a-z0-9]{4}-[a-z0-9]{4}-[a-z0-`
- Anchor relative PCRE (^) when possible
  - performance wise it's often better to go out of the way to create an additional `content` keyword for the sake of anchoring and improved performance
  - Example: `http.request_body; content:"pat"; pcre:"/^\s*poopy/R";`
  - Example: `http.request_body; content:"pat"; startswith; pcre:"/^pat\s*poopy/";`
- Put pcre after content
  - Unless you need a second content to anchor more PCRE
- Do not use `.*` in pcre without considering the performance implication of unlimited inspection depth

References

- Reference keywords should be lowercase
  - Example: `cve,2017-21354` not `CVE,2017-21354`
  - Example: `arachnids:25` not `arachNIDS:25`
- no prefixes in signature reference url
  - `url,http://` and `url,https://` should not be present in the ruleset.
  - Example: `reference:url,https://google.com;` should instead be `reference:url,google.com;`
  - Backstory: older SIEM prepended an url based on settings in `reference.config`, so an additional `http://` created broken links

Nuance Corner

JEC - discussion points and stuff for others to agree/disagree with

- In the context of EXPLOIT signatures, use `any` as the source unless the signature message explicitly states directional behavior. Ex. you can use `$EXTERNAL_NET any -> any any` if the message states something such as 'Inbound from External Source'. Reasoning: EXPLOIT sigs can see fires when the source is either external or internal, consider lateral movement scenarios.

Performance nuances

- `http.response_body;` in Suricata 5.0 performs significantly worse than `file.data;` despite `file.data;` applying to many protocols (such as SMB)
- Use `base64_*` keywords sparingly, their performance can be less than ideal
- Do not apply `fast_pattern` to content in a `base64_data` buffer, it's often better to search the encoded string with various offsets (using a script such as this from Darien Huss - https://github.com/darienhuss/base_to_content) than it is to fast_pattern the raw string after base64 decoding.
- `Tls.fingerprint` in Suricata 4 appears to be bugged and causes drastic performance degradation for unknown reasons (the worst performing ET rule currently uses this buffer and is significantly worse than anything else).
- `Urilen` is currently much faster than applying bsize to the http.uri keyword.

Emerging Threats specific

Metadata fields that are sacrosanct and within our purview are always populated:

- attack target
- Severity
- Impact
- deployment