SIGMA

News from Sigma

Thomas Patzke

EU ATT&CK Workshop

2021-06-01

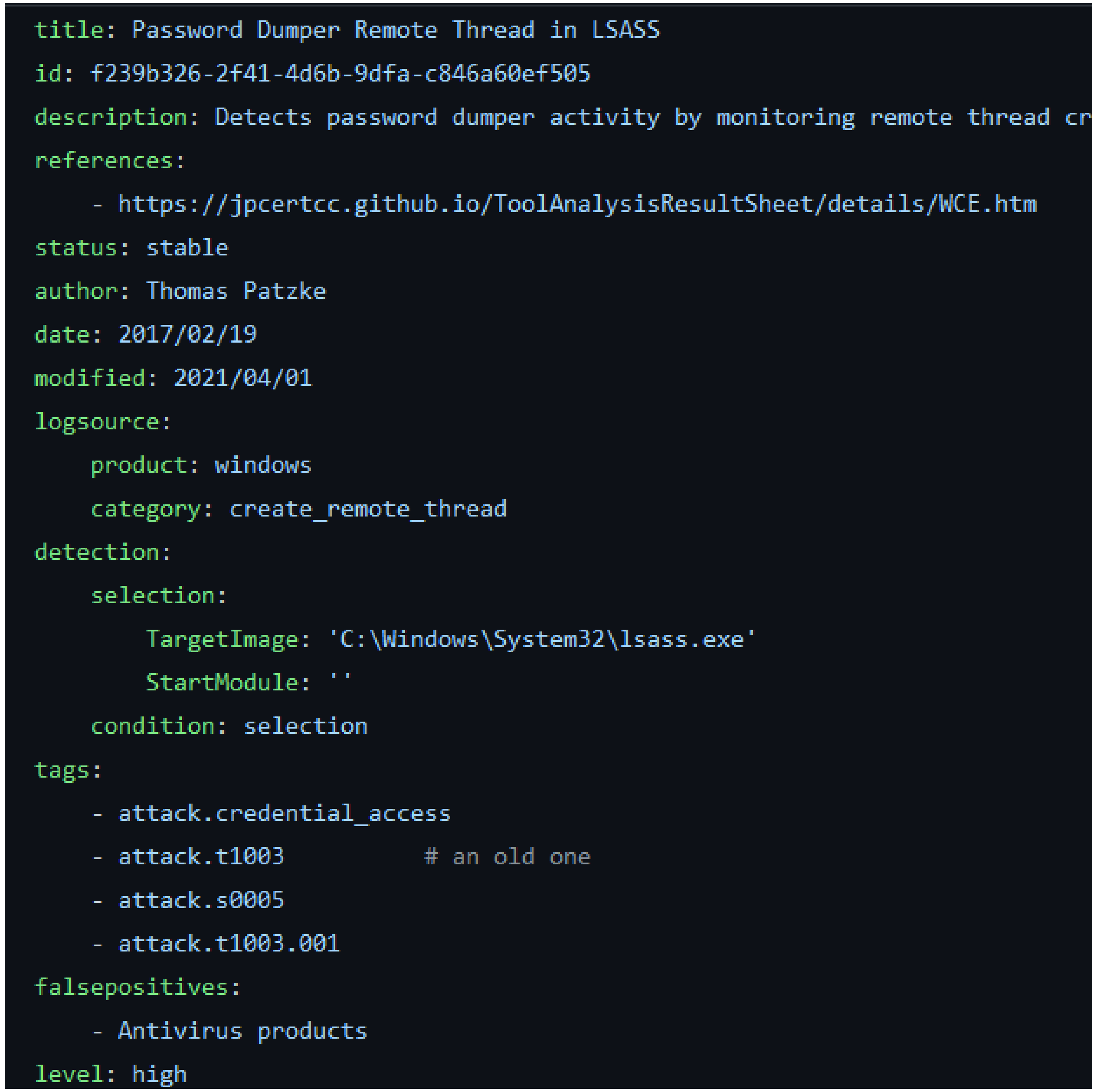
TLP:White

Agenda

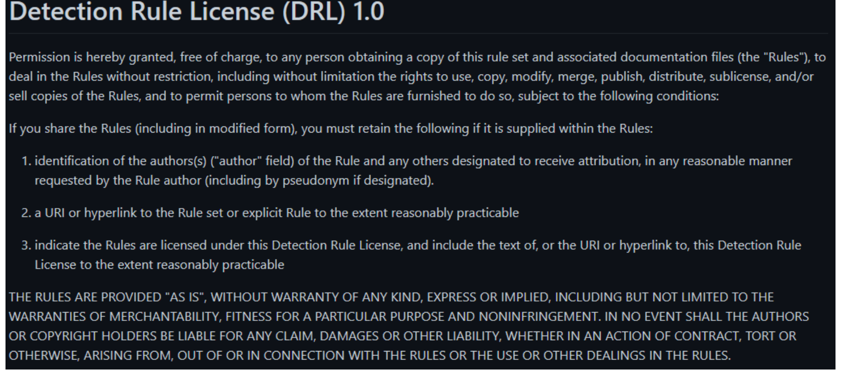
- (Very) Short intro to Sigma
- What was achieved in the last year
- What is currently developed

What is Sigma?

- Format specification for log event signatures
  - generic
  - vendor-agnostic
  - environment-agnostic
- Readable and writable by humans as well as software
- Huge (>950) open source rule set.
- Conversion tool: Sigma to SIEM/EDR query languages



```
title: Password Dumper Remote Thread in LSASS
id: f239b326-2f41-4d6b-9dfa-c846a60ef505
description: Detects password dumper activity by monitoring remote thread cr
references:
    - https://jpcertcc.github.io/ToolAnalysisResultSheet/details/WCE.htm
status: stable
author: Thomas Patzke
date: 2017/02/19
modified: 2021/04/01
logsource:
    product: windows
    category: create_remote_thread
detection:
    selection:
        TargetImage: 'C:\Windows\System32\lsass.exe'
        StartModule: ''
    condition: selection
tags:
    - attack.credential_access
    - attack.t1003          # an old one
    - attack.s0005
    - attack.t1003.001
falsepositives:
    - Antivirus products
level: high
```


Detection Rule License (DRL) 1.0

Permission is hereby granted, free of charge, to any person obtaining a copy of this rule set and associated documentation files (the "Rules"), to deal in the Rules without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Rules, and to permit persons to whom the Rules are furnished to do so, subject to the following conditions:

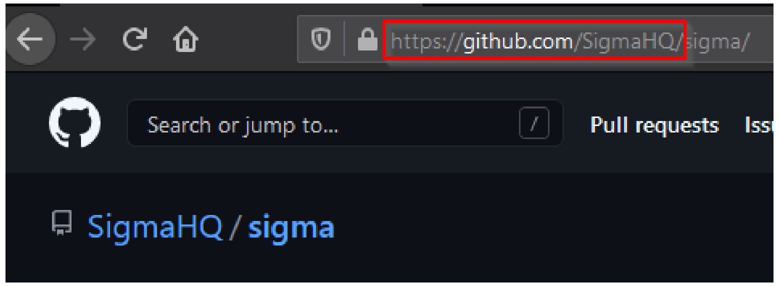
If you share the Rules (including in modified form), you must retain the following if it is supplied within the Rules:

1. identification of the authors(s) ("author" field) of the Rule and any others designated to receive attribution, in any reasonable manner requested by the Rule author (including by pseudonym if designated).
2. a URI or hyperlink to the Rule set or explicit Rule to the extent reasonably practicable
3. indicate the Rules are licensed under this Detection Rule License, and include the text of, or the URI or hyperlink to, this Detection Rule License to the extent reasonably practicable

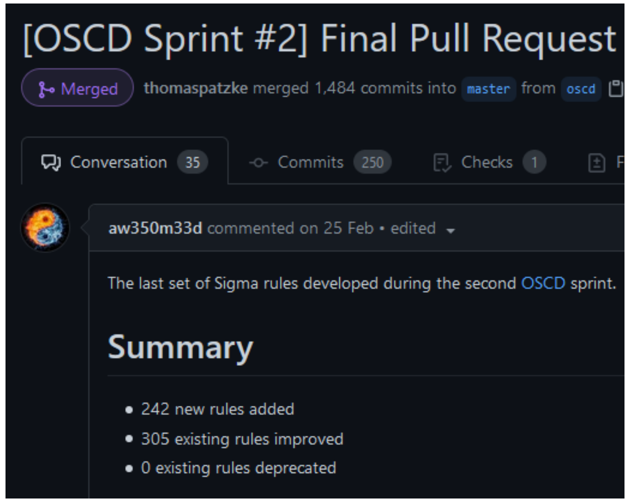
THE RULES ARE PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE RULES OR THE USE OR OTHER DEALINGS IN THE RULES.

Recent News

- Rules are now licensed under the Detection Rule License (DRL 1.0)
  - Custom developed license (yes, we considered lots of options)
  - DRL = MIT + Attribution + detection rule specific
- Severity level *informational* was introduced for rules which are intended for enrichment/tagging but not for detection.
- OSCD Initiative Sprint #2 was merged into master branch
- Sigma was moved from Florians personal account into the SigmaHQ organization.


https://github.com/SigmaHQ/sigma/

SigmaHQ / sigma


[OSCD Sprint #2] Final Pull Request

Merged thomaspatzke merged 1,484 commits into master from oscd

aw350m33d commented on 25 Feb • edited

The last set of Sigma rules developed during the second OSCD sprint.

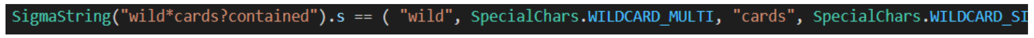
Summary

- 242 new rules added
- 305 existing rules improved
- 0 existing rules deprecated

A new Converter? Why?

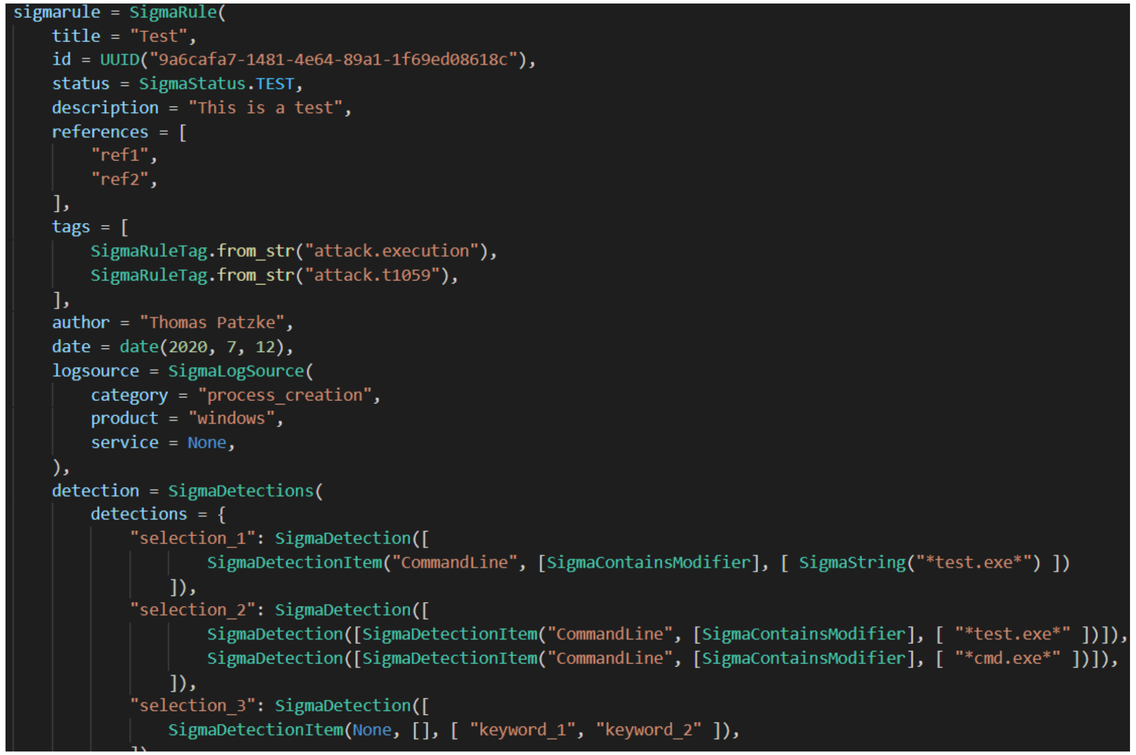
- The old one was written as PoC to prove that it’s possible.
- The initial version had only Splunk and Elasticsearch in mind, other query languages with different structures were added later.
- Lot of stuff is done at the wrong location.
- Code replication instead of code reuse.
- Bad design and missing abstraction makes it hard to implement new features.

Converter Rewrite

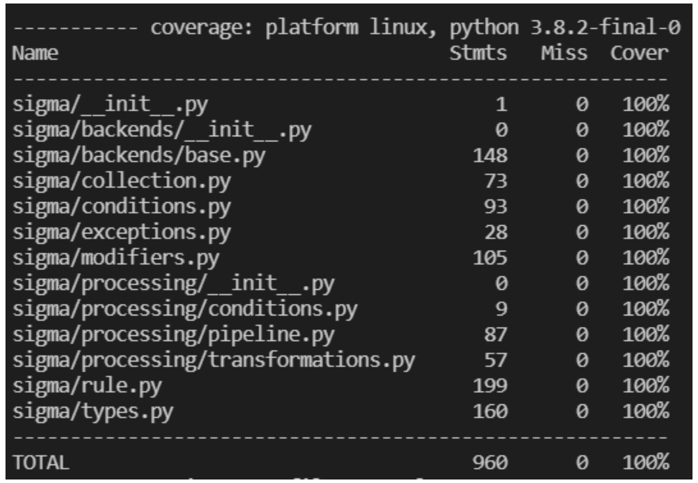


```
SigmaString("wild*cards?contained").s == ( "wild", SpecialChars.WILDCARD_MULTI, "cards", SpecialChars.WILDCARD_SI
```

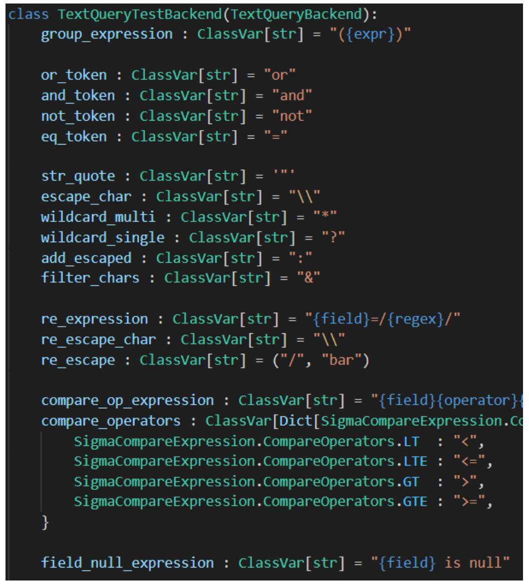
- Clean(er) design
- Sigma rule data model with typing
- More flexible backend templates
- Lots of small improvements, e.g.
  - Prefix/suffix queries
  - numeric comparisons: `response_code|gte: 500`
- Currently 99% test coverage by >200 tests, test run takes <15 seconds



```
sigmarule = SigmaRule(
    title = "Test",
    id = UUID("9a6cafa7-1481-4e64-89a1-1f69ed08618c"),
    status = SigmaStatus.TEST,
    description = "This is a test",
    references = [
        "ref1",
        "ref2",
    ],
    tags = [
        SigmaRuleTag.from_str("attack.execution"),
        SigmaRuleTag.from_str("attack.t1059"),
    ],
    author = "Thomas Patzke",
    date = date(2020, 7, 12),
    logsource = SigmaLogSource(
        category = "process_creation",
        product = "windows",
        service = None,
    ),
    detection = SigmaDetections(
        detections = {
            "selection_1": SigmaDetection([
                    SigmaDetectionItem("CommandLine", [SigmaContainsModifier], [ SigmaString("*test.exe*") ])
                ]),
            "selection_2": SigmaDetection([
                    SigmaDetection([SigmaDetectionItem("CommandLine", [SigmaContainsModifier], [ "*test.exe*" ])]),
                    SigmaDetection([SigmaDetectionItem("CommandLine", [SigmaContainsModifier], [ "*cmd.exe*" ])]),
                ]),
            "selection_3": SigmaDetection([
                SigmaDetectionItem(None, [], [ "keyword_1", "keyword_2" ]),
```



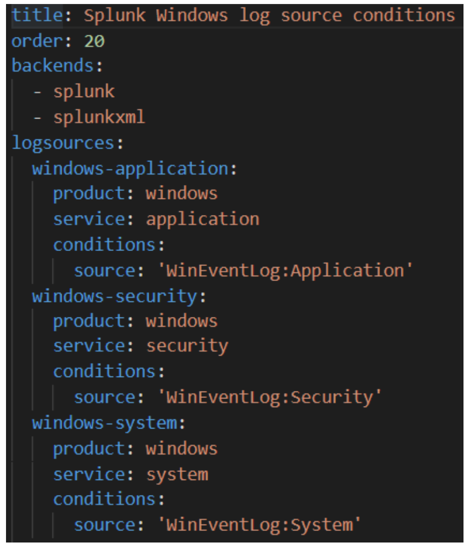
```
---------- coverage: platform linux, python 3.8.2-final-0
Name                                  Stmts   Miss  Cover
---------------------------------------------------------
sigma/__init__.py                         1      0   100%
sigma/backends/__init__.py                0      0   100%
sigma/backends/base.py                  148      0   100%
sigma/collection.py                      73      0   100%
sigma/conditions.py                      93      0   100%
sigma/exceptions.py                      28      0   100%
sigma/modifiers.py                      105      0   100%
sigma/processing/__init__.py              0      0   100%
sigma/processing/conditions.py            9      0   100%
sigma/processing/pipeline.py             87      0   100%
sigma/processing/transformations.py      57      0   100%
sigma/rule.py                           199      0   100%
sigma/types.py                          160      0   100%
---------------------------------------------------------
TOTAL                                   960      0   100%
```



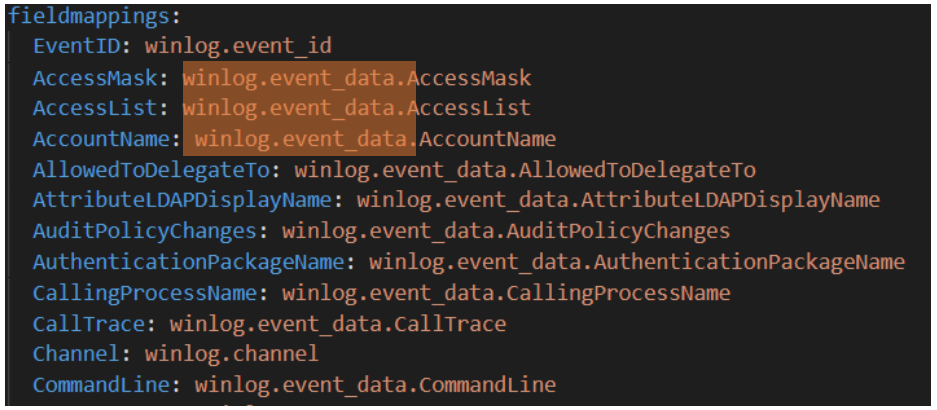
```
class TextQueryTestBackend(TextQueryBackend):
    group_expression : ClassVar[str] = "({expr})"

    or_token : ClassVar[str] = "or"
    and_token : ClassVar[str] = "and"
    not_token : ClassVar[str] = "not"
    eq_token : ClassVar[str] = "="

    str_quote : ClassVar[str] = '"'
    escape_char : ClassVar[str] = "\\"
    wildcard_multi : ClassVar[str] = "*"
    wildcard_single : ClassVar[str] = "?"
    add_escaped : ClassVar[str] = ":"
    filter_chars : ClassVar[str] = "&"

    re_expression : ClassVar[str] = "{field}=/{regex}/"
    re_escape_char : ClassVar[str] = "\\"
    re_escape : ClassVar[str] = ("/", "bar")

    compare_op_expression : ClassVar[str] = "{field}{operator}{
    compare_operators : ClassVar[Dict[SigmaCompareExpression.Co
        SigmaCompareExpression.CompareOperators.LT  : "<",
        SigmaCompareExpression.CompareOperators.LTE : "<=",
        SigmaCompareExpression.CompareOperators.GT  : ">",
        SigmaCompareExpression.CompareOperators.GTE : ">=",
    }

    field_null_expression : ClassVar[str] = "{field} is null"
```

Current State of Conversion Configuration

- It’s inflexible!
- It supports log source definitions…
- …and field name mappings…
- …that’s it!
- No “prefix field names”
- No operations on values (e.g. split, join)



```yaml
title: Splunk Windows log source conditions
order: 20
backends:
  - splunk
  - splunkxml
logsources:
  windows-application:
    product: windows
    service: application
    conditions:
      source: 'WinEventLog:Application'
  windows-security:
    product: windows
    service: security
    conditions:
      source: 'WinEventLog:Security'
  windows-system:
    product: windows
    service: system
    conditions:
      source: 'WinEventLog:System'
```



```yaml
fieldmappings:
  EventID: winlog.event_id
  AccessMask: winlog.event_data.AccessMask
  AccessList: winlog.event_data.AccessList
  AccountName: winlog.event_data.AccountName
  AllowedToDelegateTo: winlog.event_data.AllowedToDelegateTo
  AttributeLDAPDisplayName: winlog.event_data.AttributeLDAPDisplayName
  AuditPolicyChanges: winlog.event_data.AuditPolicyChanges
  AuthenticationPackageName: winlog.event_data.AuthenticationPackageName
  CallingProcessName: winlog.event_data.CallingProcessName
  CallTrace: winlog.event_data.CallTrace
  Channel: winlog.channel
  CommandLine: winlog.event_data.CommandLine
```

New: Rule Transformation Pipeline

- Rule Transformation pipeline = Sequence of transformation operations on Sigma rule
- Operation types like *add condition*, *field name mapping*, …
- Conditions can be attached to each operation which decide if it is applied to a given rule.
- Conditions can also define dependencies between operations.
- Variables

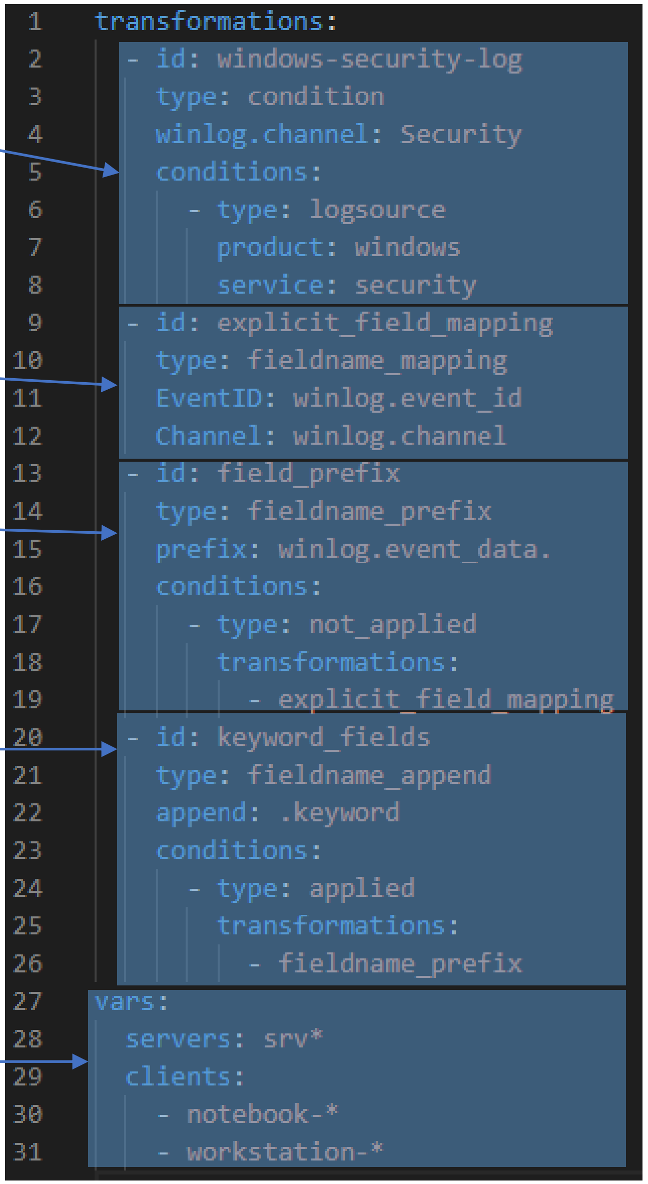
Add condition matching winlog.channel to “Security” if Sigma rule specifies Windows Security log as log source.

Field name mappings

Prefix field names not explicitly mapped by previous rule.

Append a suffix to all fields prefixed by previous rule.

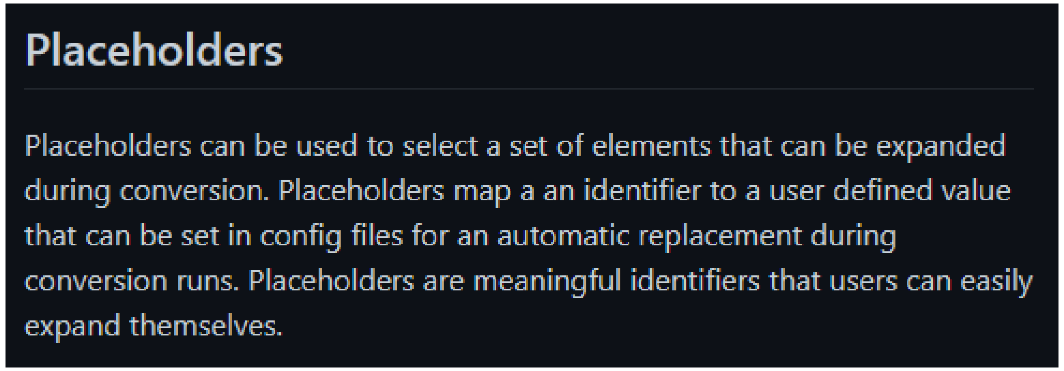
Variables which can be used in placeholders or conditions.



```
transformations:
  - id: windows-security-log
    type: condition
    winlog.channel: Security
    conditions:
      - type: logsource
        product: windows
        service: security
  - id: explicit_field_mapping
    type: fieldname_mapping
    EventID: winlog.event_id
    Channel: winlog.channel
  - id: field_prefix
    type: fieldname_prefix
    prefix: winlog.event_data.
    conditions:
      - type: not_applied
        transformations:
          - explicit_field_mapping
  - id: keyword_fields
    type: fieldname_append
    append: .keyword
    conditions:
      - type: applied
        transformations:
          - fieldname_prefix
vars:
  servers: srv*
  clients:
    - notebook-*
    - workstation-*
```

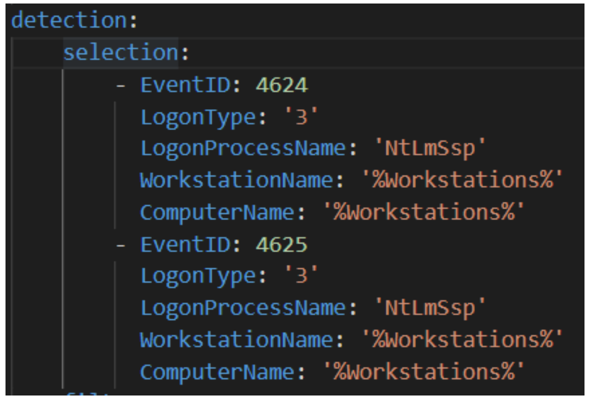
Placeholder History

- Specified since the beginning
- It is used in rules
- But it was never implemented in the converter and is often requested from the community.
- Now it’s coming! ☺



Placeholders

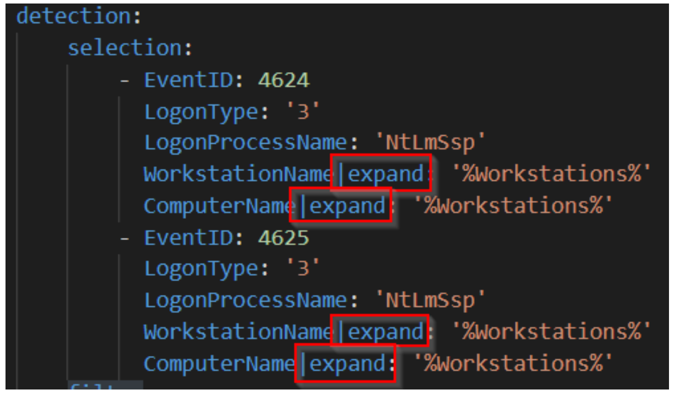
Placeholders can be used to select a set of elements that can be expanded during conversion. Placeholders map a an identifier to a user defined value that can be set in config files for an automatic replacement during conversion runs. Placeholders are meaningful identifiers that users can easily expand themselves.



```
detection:
    selection:
        - EventID: 4624
          LogonType: '3'
          LogonProcessName: 'NtLmSsp'
          WorkstationName: '%Workstations%'
          ComputerName: '%Workstations%'
        - EventID: 4625
          LogonType: '3'
          LogonProcessName: 'NtLmSsp'
          WorkstationName: '%Workstations%'
          ComputerName: '%Workstations%'
```

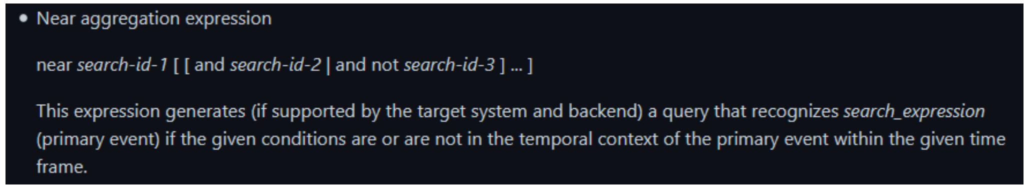
Placeholder Support in new Converter

- Value modifier *expand* to distinguish between intentional `%values%` and `%placeholders%`.
- % can also be escaped (\%) inside expanded value
- Processing pipeline defines expansion
  - Into plain values:
    … `src=notebook-* OR src=workstation-*` …
  - Into condition:
    … `tag=workstation` …
  - Into table lookup:
    … `[ | inputlookup workstations | rename name as src ]` …
  - Into wildcard (last resort fallback):
    … `src=*` …



```yaml
detection:
    selection:
        - EventID: 4624
          LogonType: '3'
          LogonProcessName: 'NtLmSsp'
          WorkstationName|expand: '%Workstations%'
          ComputerName|expand: '%Workstations%'
        - EventID: 4625
          LogonType: '3'
          LogonProcessName: 'NtLmSsp'
          WorkstationName|expand: '%Workstations%'
          ComputerName|expand: '%Workstations%'
```

Aggregations and Correlations



- Near aggregation expression

  near *search-id-1* [[and *search-id-2* | and not *search-id-3*] ...]

  This expression generates (if supported by the target system and backend) a query that recognizes *search_expression* (primary event) if the given conditions are or are not in the temporal context of the primary event within the given time frame.

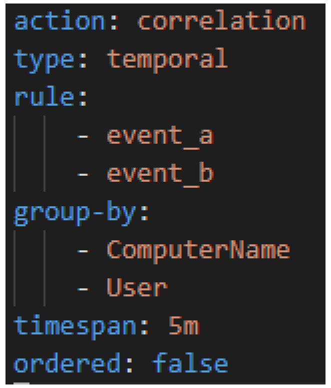
- Not well supported by Sigma converter
- Underspecified
  - Everything must fit within a single Sigma rule
  - This means: no relationship between events from different log sources
  - Overloaded *timeframe* parameter
  - Event Order?
  - Event originating from the same system? The same user? Both?

Sigma Correlations

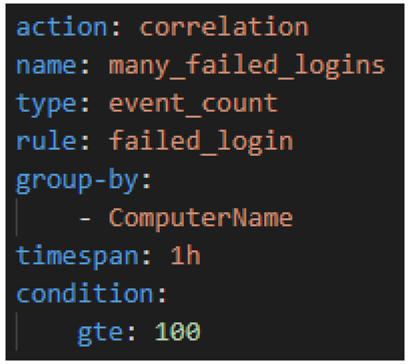
- Aggregations are dropped completely from conditions
- YAML-based, multiple YAML documents in one Sigma file
- Sigma rules describe single events
- Sigma correlation rules describe how the events must be combined to raise a detection

Examples: Sigma Correlations

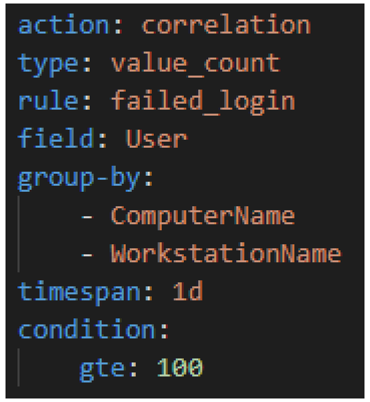
1. Event A and B must appear within five minutes on the same system by the same user.
2. 100 failed logins on a single system within 1 hour.
3. Failed logins to 100 different users counted for each source/target system pair within 1 day.



```yaml
action: correlation
type: temporal
rule:
    - event_a
    - event_b
group-by:
    - ComputerName
    - User
timespan: 5m
ordered: false
```



```yaml
action: correlation
name: many_failed_logins
type: event_count
rule: failed_login
group-by:
    - ComputerName
timespan: 1h
condition:
    gte: 100
```



```yaml
action: correlation
type: value_count
rule: failed_login
field: User
group-by:
    - ComputerName
    - WorkstationName
timespan: 1d
condition:
    gte: 100
```

Roadmap

- First preview release without correlations in the next weeks
  - Processing pipelines support
  - Placeholders
- First correlations support in the next months
- Porting/developing backends
- New Sigma converter will be the 1.x release, old one 0.x
  - Main development efforts go into 1.x
  - Maintenance and bug fixing in 0.x
  - 0.x will be discontinued when most relevant and still maintained backends are ported.
- Rules and converter will be separate projects below the SigmaHQ GitHub organization

Questions?

Answers!

- Now
- https://github.com/SigmaHQ/sigma
- https://siemexchange.slack.com/
- thomas@patzke.org
- Twitter:
  - @blubbfiction
  - @sigma_hq

Want to contribute?

Your code and rules are welcome!

Code:

- Develop or maintain a backend
- Documentation
- Fix an issue

Rules:

- Test existing rules and make a "state: stable" pull request
- Improve existing rules
- Write a missing detection