



YARA-L Cheat Sheet

Google SecOps language for
Search/Rules/Dashboard

UDM search [🔗](#)

```
metadata.event_type = "USER_LOGIN"
security_result.action = "FAIL"
```

Use UDM Lookup tool to find fields (and values) or Gemini to create searches. For searches, events section is implied, not for rules/dashboards.

UDM grouped fields search [🔗](#)

```
user = "root"
domain = "www.gmail.com" nocase
```

Hint: Use the `⌘+↵` / `alt+↵` shortcut to launch search.

Statistics (aggregation) [🔗](#)

`match`: domain, user

- Hop windows (overlapping, rule default): `match`: user over 1d
- Tumbling windows (non-overlapping): `match`: user by DAY
- Sliding: `match`: over 10m after \$login

Windowing and Scheduling information is available here: [🔗](#)

Outcome [🔗](#)

```
outcome: $data_sent = sum(network.sent_bytes)
outcome: $score = max(
  if(principal.hostname = /win-adfs/, 5,
  if(principal.hostname = /server/, 3,
  0)))
```

Aggregate functions [🔗](#)

- | | | |
|------------------|------------------|----------|
| ○ array | ○ count | ○ min |
| ○ array_distinct | ○ count_distinct | ○ sum |
| ○ avg | ○ max | ○ stddev |

Others functions are here: [🔗](#)

Conditions [🔗](#)

```
condition: #e > 5
condition: ANY of [
  $vt_first_seen_time = 1,
  $vt_last_analysis_time = 1 ]
```

Using REGEX

```
re.regex(network.email.from, `.*goggle\.com`)
network.email.from = /.goggle\.com/
```

SecOps uses RE2 library [🔗](#)

Raw search [🔗](#)

```
raw = "root"
raw = /. AND parsed = false
```

Variables [🔗](#)

```
$destination = target.ip
```

Using variable is available in events & outcome sections. Can be reversed.

Modifiers

```
limit: 42
order: $count desc
select: principal.ip
unselect: namespace, $destination
dedup: target.hostname
```

Search in Entity Graph [🔗](#)

```
graph.metadata.entity_type = "FILE"
graph.metadata.entity_type = "ASSET" AND
net.ip_in_range_cidr(graph.entity.ip, "192.168.0.0/16")
```

Entity Graph stores contextual data (assets, users, IOCs, prevalence...)

Link with data tables [🔗](#)

- Row based (join):
target.hostname = %very_suspicious.hostname
- Column-based comparison:
`not` security_result.rule_name `IN` regex %white_rules.regex

Join (left / right) [🔗](#)

```
$e.metadata.event_type = "NETWORK_CONNECTION"
$g.graph.metadata.entity_type = "ASSET"
left join $e.principal.asset.hostname = $g.graph.entity.asset.hostname
```

Since two different events are used, variables are associated for each.

Export to data tables (rules only) [🔗](#)

```
export:
  %mydatatable.write_row(host: $hostname, port: $port_nb)
```

Multistage: Composite rules (rules only) [🔗](#)

```
stage absolute_deviations { ...
  outcome: $host = ... }
$host = $absolute_deviations.host
```