RawSec

WHIDS: update

an open source EDR

GitHub / Twitter: 0xrawsec

Project: https://github.com/0xrawsec/whids

Introduction

Who is talking ?

First Name: Quentin **Last Name**: JEROME **Age**: 33

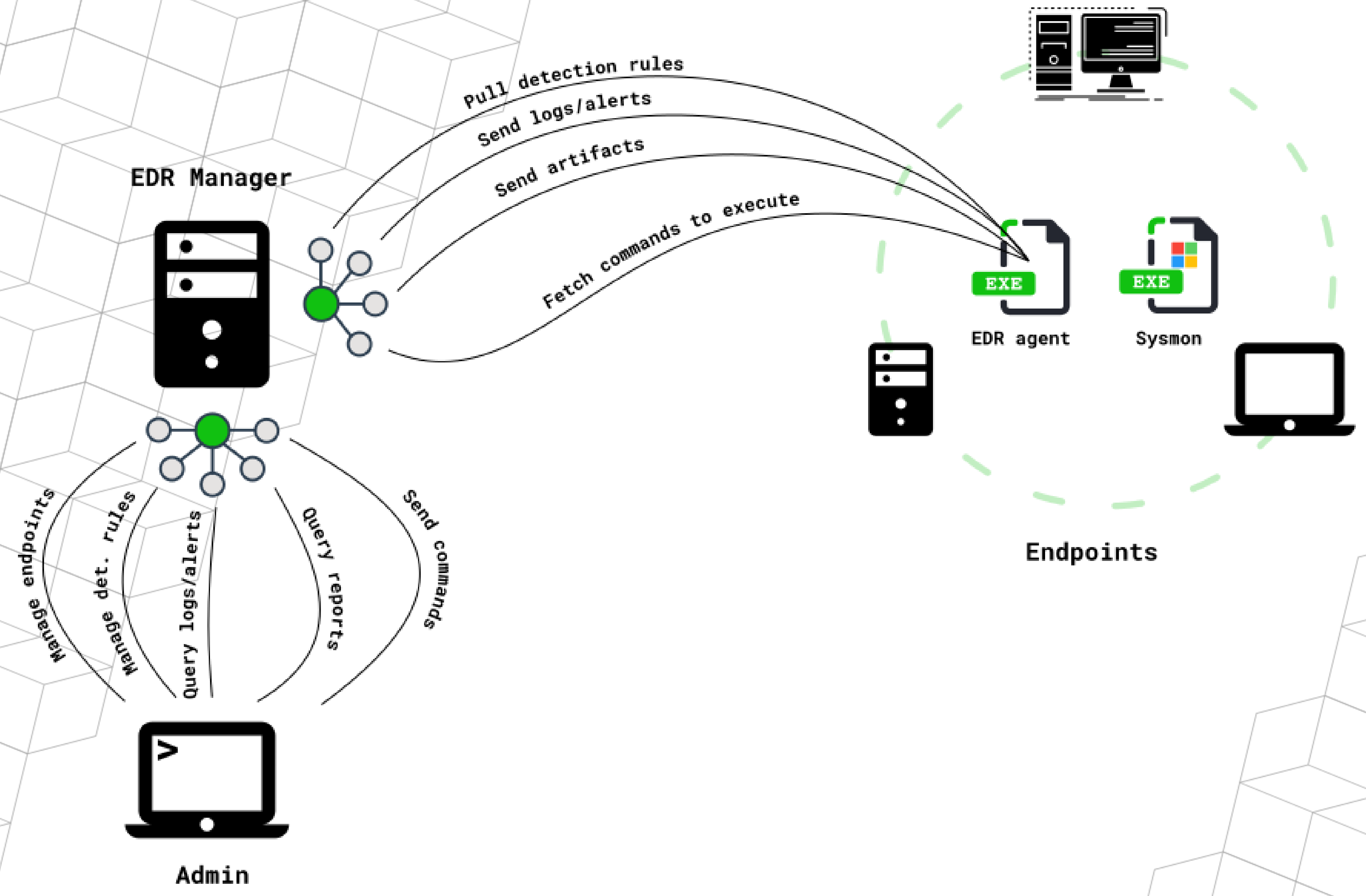
Job: Freelance Security Consultant working in Luxembourg

- **Background**: doing Incident Response, digital forensics, endpoint’s based Threat Hunting …
- **Now:** Open-Source developer mainly in Go, C, Python. At the origin of several projects: Gene, WHIDS, golang-evtx, golang-misp, golang-etw …

Why this project ?

- to bring open source alternative
- to help people
- to stimulate my brain

What it is, in image



More details !

RawSec

Agent

- **Correlate & enrich** events on host
- **Detect** in real time suspicious events (raw/correlated)
  - **100%** based on custom rules -> no rules -> no detection -> ☹
- **React** to detection in real-time
  - dump artifacts (files, process memory, registries)
  - dump a nice JSON reports (ref. as IR reports)
    - directly usable by incident handlers
    - for automation
  - blacklist process
  - kill process

Manager

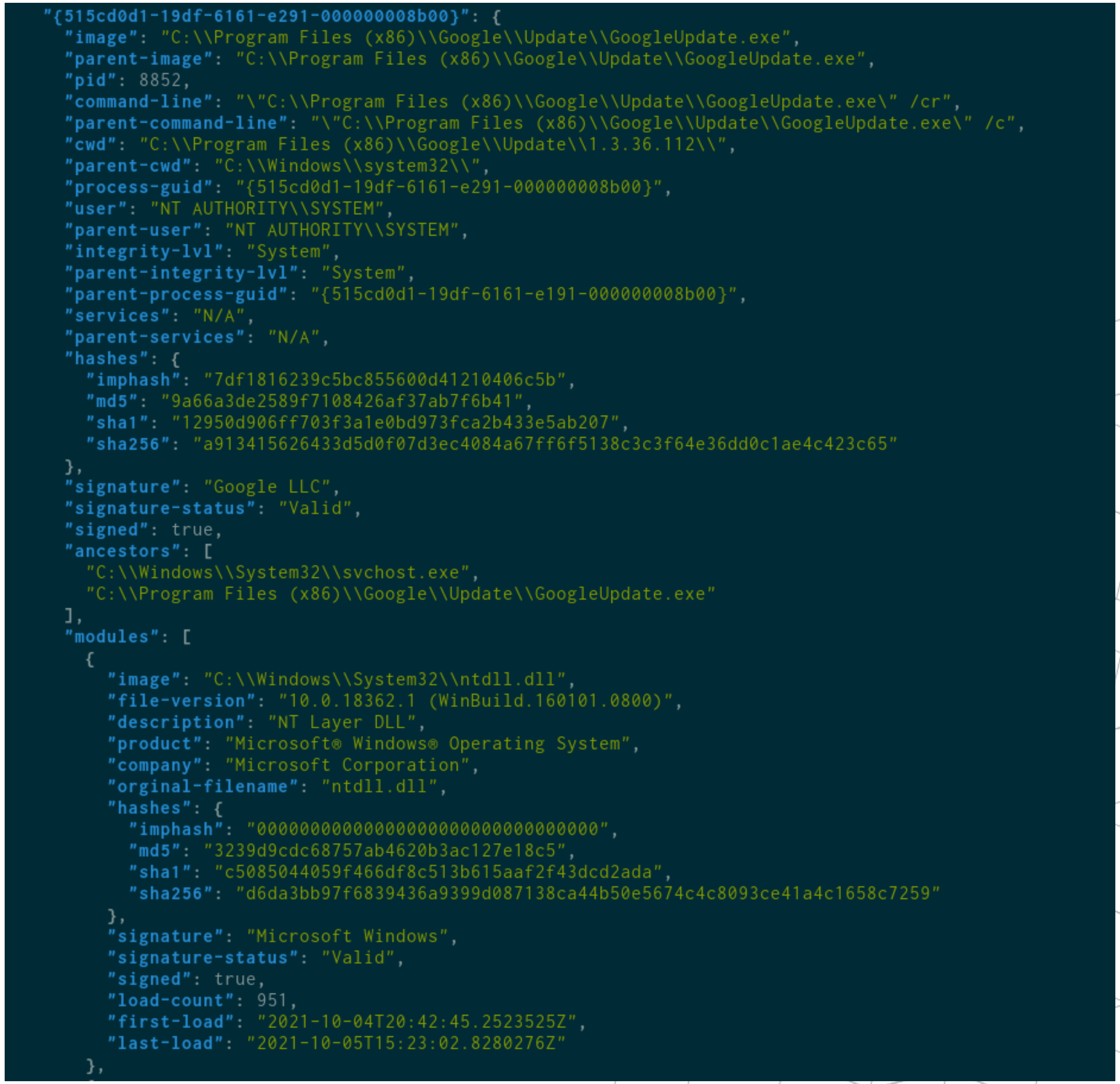
- **Central** manager to administrate endpoints
- **Collect** alerts, logs, and artifacts
- **HTTP API** for administrators and plugins

You said reports ?

On-demand or automatic

- running processes
  - DLLs loaded
  - network connections
  - DNS resolutions
  - last files opened
- all drivers/DLLs ever loaded since boot
- specific commands configured
  - OSQuery
  - …

Basically all you the data you need to understand what is going on



```
  "{515cd0d1-19df-6161-e291-000000008b00}": {
    "image": "C:\\Program Files (x86)\\Google\\Update\\GoogleUpdate.exe",
    "parent-image": "C:\\Program Files (x86)\\Google\\Update\\GoogleUpdate.exe",
    "pid": 8852,
    "command-line": "\"C:\\Program Files (x86)\\Google\\Update\\GoogleUpdate.exe\" /cr",
    "parent-command-line": "\"C:\\Program Files (x86)\\Google\\Update\\GoogleUpdate.exe\" /c",
    "cwd": "C:\\Program Files (x86)\\Google\\Update\\1.3.36.112\\",
    "parent-cwd": "C:\\Windows\\system32\\",
    "process-guid": "{515cd0d1-19df-6161-e291-000000008b00}",
    "user": "NT AUTHORITY\\SYSTEM",
    "parent-user": "NT AUTHORITY\\SYSTEM",
    "integrity-lvl": "System",
    "parent-integrity-lvl": "System",
    "parent-process-guid": "{515cd0d1-19df-6161-e191-000000008b00}",
    "services": "N/A",
    "parent-services": "N/A",
    "hashes": {
      "imphash": "7df1816239c5bc855600d41210406c5b",
      "md5": "9a66a3de2589f7108426af37ab7f6b41",
      "sha1": "12950d906ff703f3a1e0bd973fca2b433e5ab207",
      "sha256": "a913415626433d5d0f07d3ec4084a67ff6f5138c3c3f64e36dd0c1ae4c423c65"
    },
    "signature": "Google LLC",
    "signature-status": "Valid",
    "signed": true,
    "ancestors": [
      "C:\\Windows\\System32\\svchost.exe",
      "C:\\Program Files (x86)\\Google\\Update\\GoogleUpdate.exe"
    ],
    "modules": [
      {
        "image": "C:\\Windows\\System32\\ntdll.dll",
        "file-version": "10.0.18362.1 (WinBuild.160101.0800)",
        "description": "NT Layer DLL",
        "product": "Microsoft® Windows® Operating System",
        "company": "Microsoft Corporation",
        "orginal-filename": "ntdll.dll",
        "hashes": {
          "imphash": "00000000000000000000000000000000",
          "md5": "3239d9cdc68757ab4620b3ac127e18c5",
          "sha1": "c5085044059f466df8c513b615aaf2f43dcd2ada",
          "sha256": "d6da3bb97f6839436a9399d087138ca44b50e5674c4c8093ce41a4c1658c7259"
        },
        "signature": "Microsoft Windows",
        "signature-status": "Valid",
        "signed": true,
        "load-count": 951,
        "first-load": "2021-10-04T20:42:45.2523525Z",
        "last-load": "2021-10-05T15:23:02.8280276Z"
      },
```

Other Features

RawSec

On endpoints

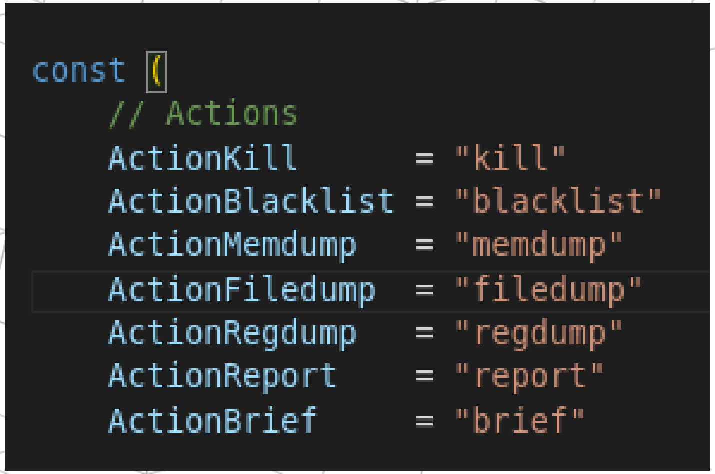
- Canary files management
  - Creates dummy canary files and alerts when they are accessed
  - Use existing files and consider them as canaries
- Configure Audit Policies, specific log channels to monitor …

On Manager

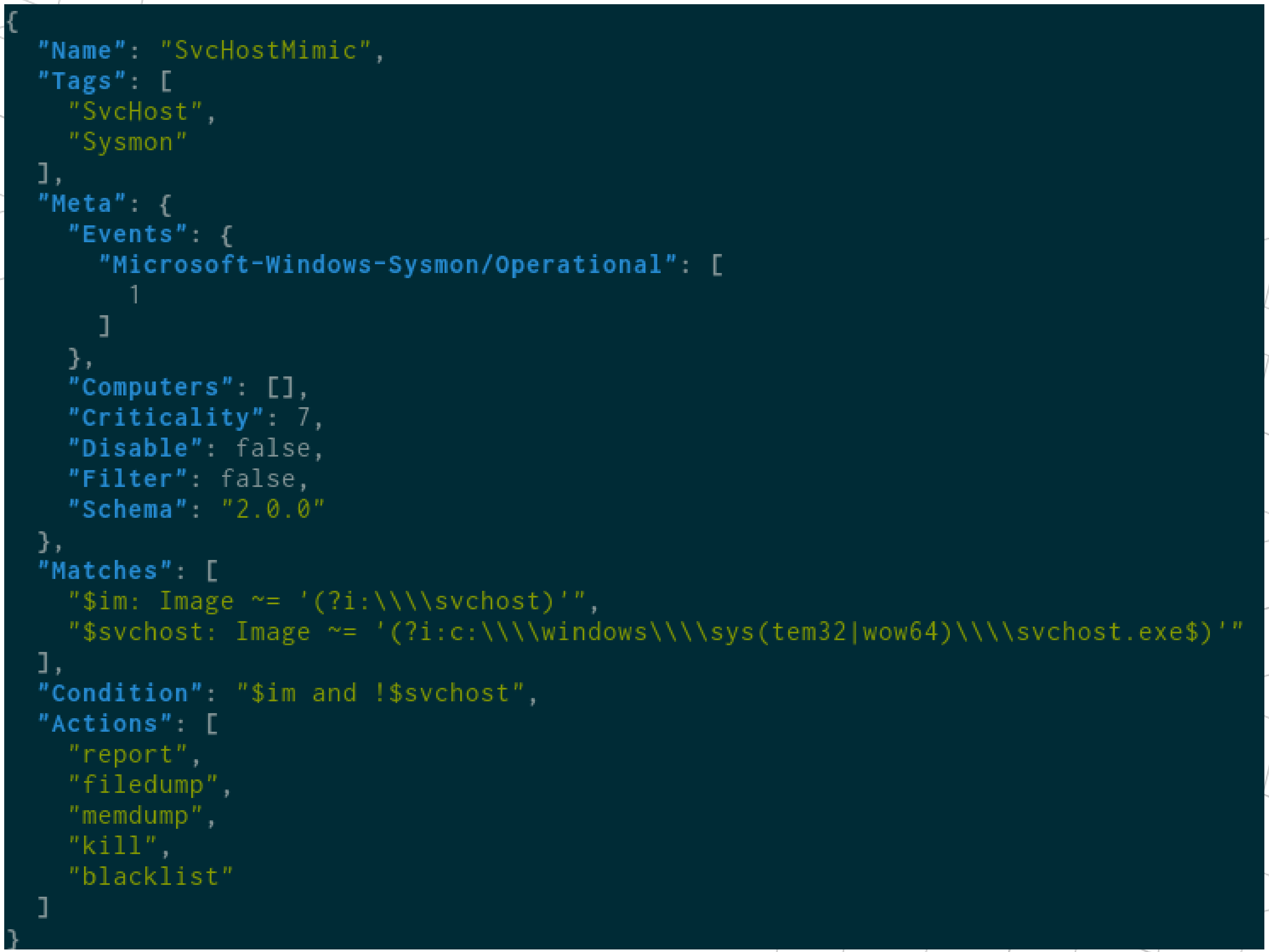
- Query logs and alerts
  - Pivot on timestamps
- Detection reports (aggregates alerts on a given time frame) to rank endpoints and prioritize analysis
- Artifacts storage -> memdumps, files, registries, reports
  - Can be used by analysts or for automation
- Event streaming through Websocket -> consume events in realtime
- Plugins development/usage
  - **reporting.py** -> push reports to MISP
  - **sightings.py** -> update MISP attributes sightings
  - **sync_iocs.py** -> update IoCs from MISP (manages attribute deletion)

What’s new ?

Better Action Granularity



```
const (
    // Actions
    ActionKill      = "kill"
    ActionBlacklist = "blacklist"
    ActionMemdump   = "memdump"
    ActionFiledump  = "filedump"
    ActionRegdump   = "regdump"
    ActionReport    = "report"
    ActionBrief     = "brief"
```

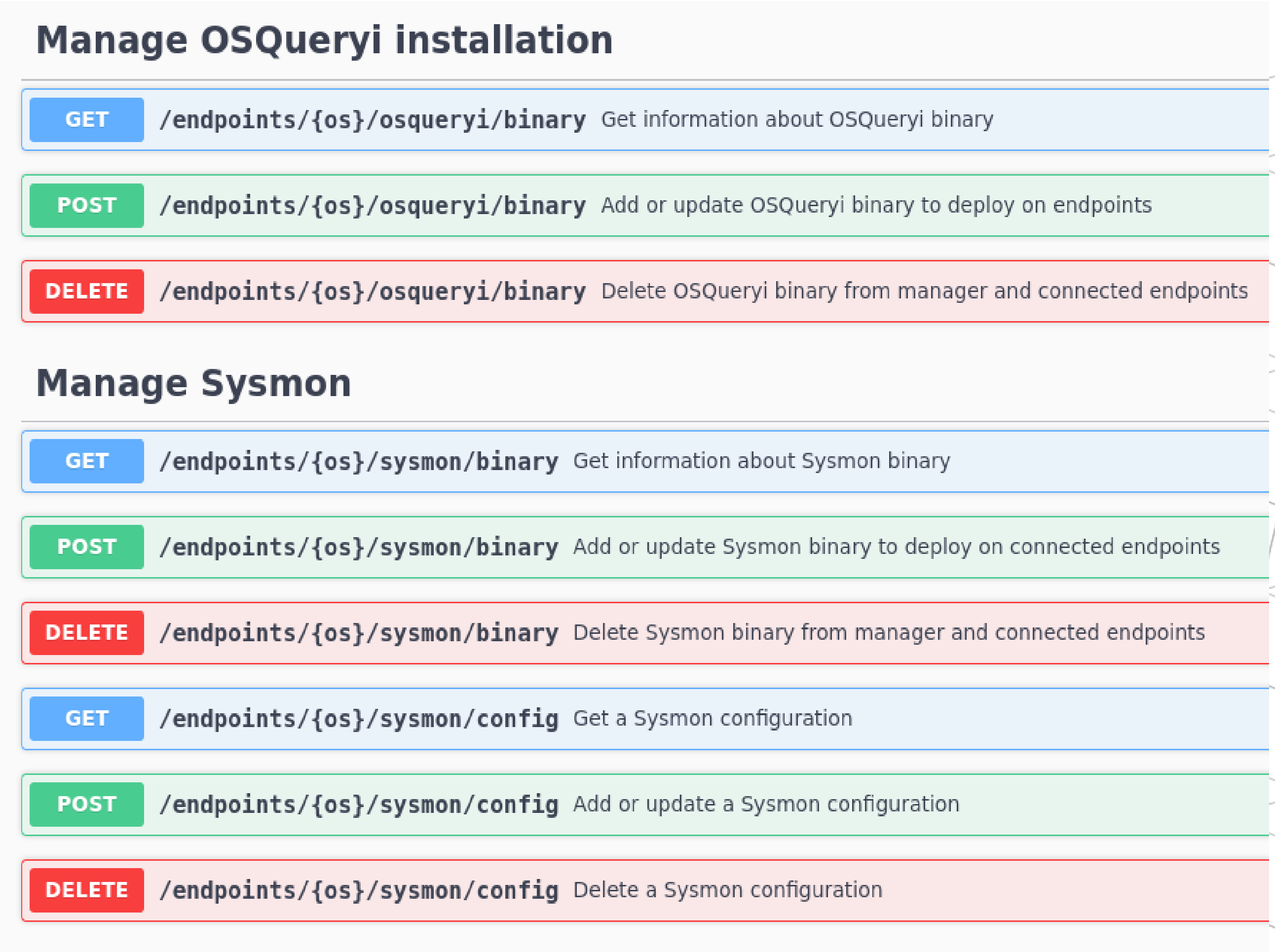


```
{
  "Name": "SvcHostMimic",
  "Tags": [
    "SvcHost",
    "Sysmon"
  ],
  "Meta": {
    "Events": {
      "Microsoft-Windows-Sysmon/Operational": [
        1
      ]
    },
    "Computers": [],
    "Criticality": 7,
    "Disable": false,
    "Filter": false,
    "Schema": "2.0.0"
  },
  "Matches": [
    "$im: Image ~= '(?i:\\\\svchost)'",
    "$svchost: Image ~= '(?i:c:\\\\windows\\\\sys(tem32|wow64)\\\\svchost.exe$)'"
  ],
  "Condition": "$im and !$svchost",
  "Actions": [
    "report",
    "filedump",
    "memdump",
    "kill",
    "blacklist"
  ]
}
```

Brand new tools management

Manage external tools

- Deploy OSQuery
  - Can be used to gather forensic data via EDR commands
- Deploy Sysmon
- Manage Sysmon config
  - Update your Sysmon configuration from a central place
  - If no configuration is present, agent is using a default config



Manage OSQueryi installation

GET `/endpoints/{os}/osqueryi/binary` Get information about OSQueryi binary

POST `/endpoints/{os}/osqueryi/binary` Add or update OSQueryi binary to deploy on endpoints

DELETE `/endpoints/{os}/osqueryi/binary` Delete OSQueryi binary from manager and connected endpoints

Manage Sysmon

GET `/endpoints/{os}/sysmon/binary` Get information about Sysmon binary

POST `/endpoints/{os}/sysmon/binary` Add or update Sysmon binary to deploy on connected endpoints

DELETE `/endpoints/{os}/sysmon/binary` Delete Sysmon binary from manager and connected endpoints

GET `/endpoints/{os}/sysmon/config` Get a Sysmon configuration

POST `/endpoints/{os}/sysmon/config` Add or update a Sysmon configuration

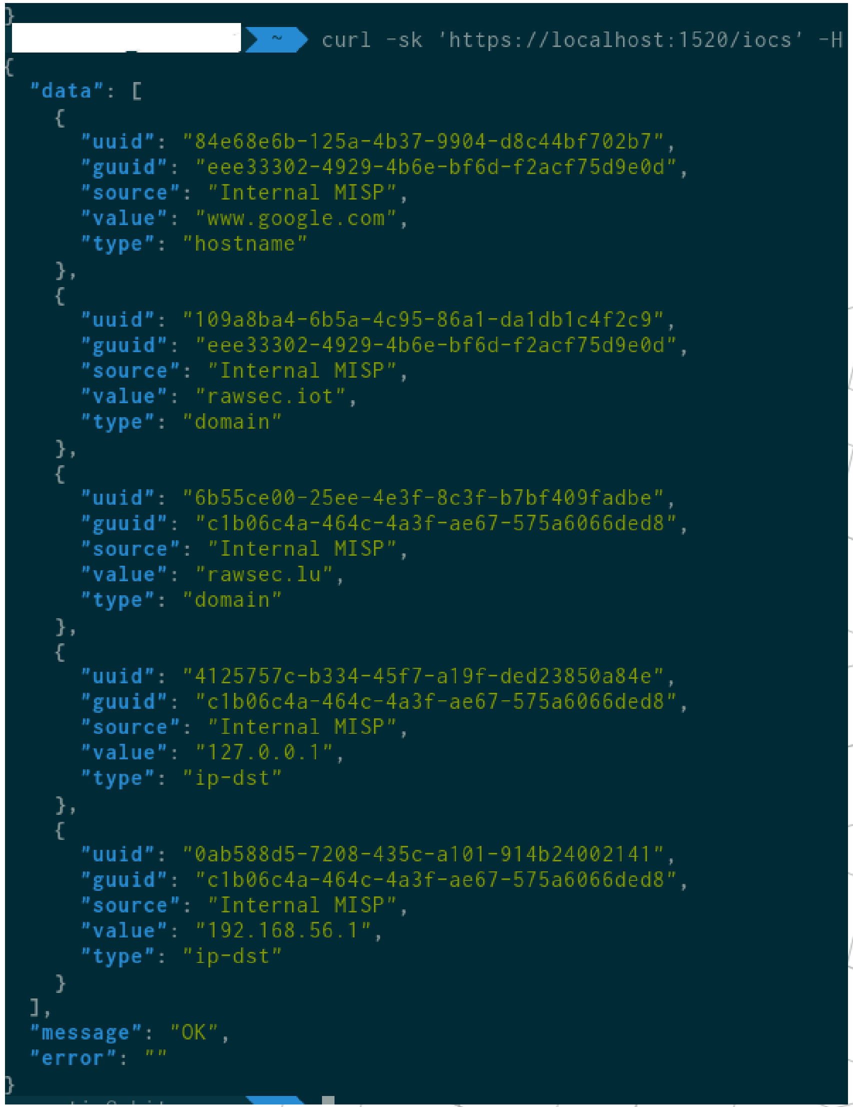
DELETE `/endpoints/{os}/sysmon/config` Delete a Sysmon configuration

New IoC management

Previously, the only way to push IoCs on agents (used for detection) was to connect the manager to a **unique** MISP instance.

Now, an HTTP API is available to manage IoCs (add/update/delete)

- more flexibility
- not strongly tight to MISP -> easy to push IoCs from any source
- what's up with MISP ?
  - You can use IoCs from **several** MISP instances
  - A specific EDR plugin has been developed to synchronize IoCs from a **MISP instance** (**sync_iocs.py**) -> simply configure it and run it



```
}
[redacted] ~  curl -sk 'https://localhost:1520/iocs' -H
{
  "data": [
    {
      "uuid": "84e68e6b-125a-4b37-9904-d8c44bf702b7",
      "guuid": "eee33302-4929-4b6e-bf6d-f2acf75d9e0d",
      "source": "Internal MISP",
      "value": "www.google.com",
      "type": "hostname"
    },
    {
      "uuid": "109a8ba4-6b5a-4c95-86a1-da1db1c4f2c9",
      "guuid": "eee33302-4929-4b6e-bf6d-f2acf75d9e0d",
      "source": "Internal MISP",
      "value": "rawsec.iot",
      "type": "domain"
    },
    {
      "uuid": "6b55ce00-25ee-4e3f-8c3f-b7bf409fadbe",
      "guuid": "c1b06c4a-464c-4a3f-ae67-575a6066ded8",
      "source": "Internal MISP",
      "value": "rawsec.lu",
      "type": "domain"
    },
    {
      "uuid": "4125757c-b334-45f7-a19f-ded23850a84e",
      "guuid": "c1b06c4a-464c-4a3f-ae67-575a6066ded8",
      "source": "Internal MISP",
      "value": "127.0.0.1",
      "type": "ip-dst"
    },
    {
      "uuid": "0ab588d5-7208-435c-a101-914b24002141",
      "guuid": "c1b06c4a-464c-4a3f-ae67-575a6066ded8",
      "source": "Internal MISP",
      "value": "192.168.56.1",
      "type": "ip-dst"
    }
  ],
  "message": "OK",
  "error": ""
}
```

Central agent configuration

So far, we had to edit agent’s configuration file on the endpoint

Now, agent configuration is managed from an API

- Easier / quicker management
- Centralized view and management of all configurations
- If agent configuration is not known to the manager, the manager takes the configuration sent by the agent
- Agent pulls configuration updates and restart if needed

This feature was the last remaining step to have a full centralized management of endpoints

ETW support (agent side)

ETW (Event Tracing Windows): Windows technology to trace and log events.

Previously, WHIDS was subscribing to Windows log channels using the Windows EvtXXX API family. However this approach has several drawbacks, compared to ETW.

Why using ETW is better ?

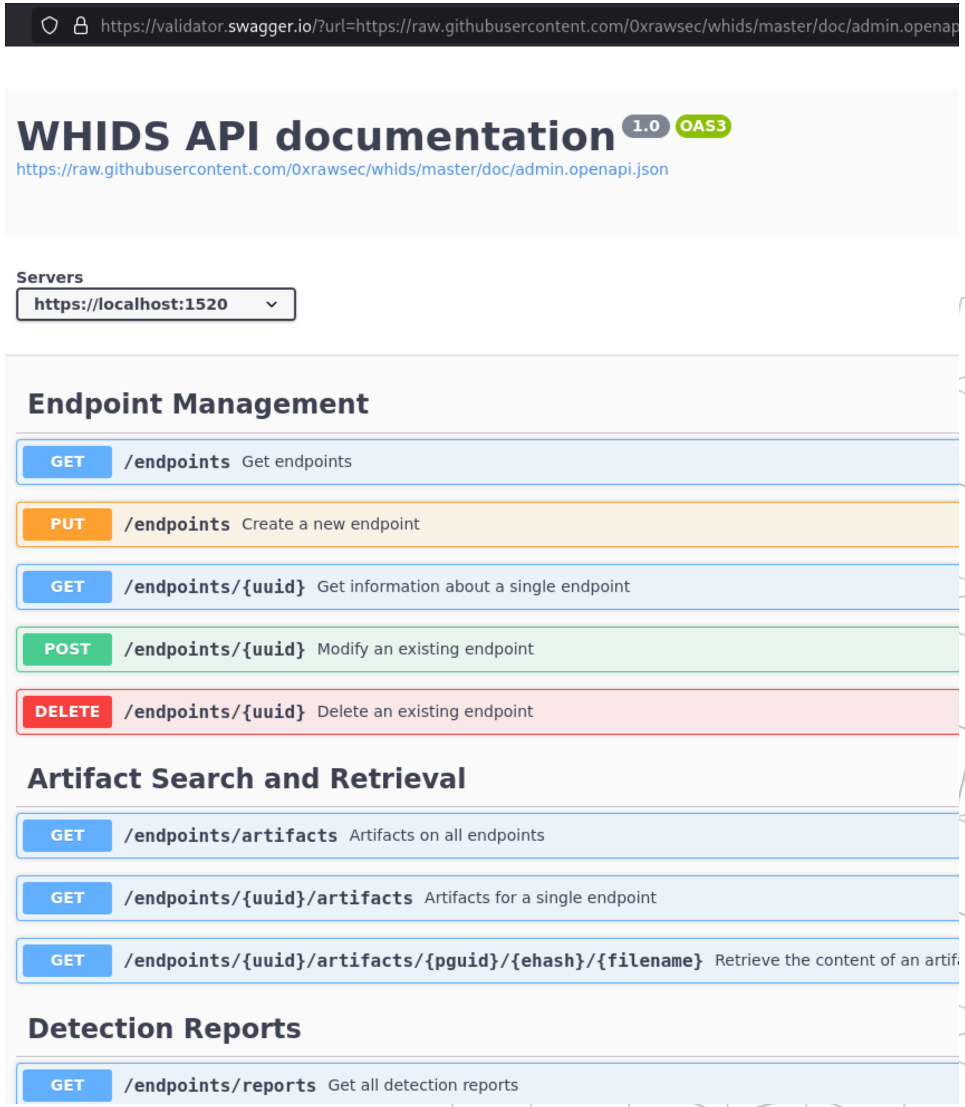
- Using EvtXXX API family is vulnerable to Invoke-Phant0m (suspending Windows Event Log service)
- Best speed (as it is closer to the kernel)
- Better flexibility
  - we can filter ETW events from APIs, so no useless resources are allocated if not needed
  - we control event parsing so we might decide not to parse all events
- Better visibility -> more ETW providers than Windows Log Channels
- You get some nice features to buffer events even if you don’t consume them, to protect your ETW session from being stopped …

All this work led to the development of golang-etw module

Improved documentation (1)

Auto-generated OpenAPI documentation (for Admin API)

- › Built on top of golang tests
  - Documentation is always in line with a commit/project version
  - Allow to get HTTP API code coverage in the same time as building documentation
- › Ran by default at every commit (using git hook)
  - If tests are failing, changes are not committed



https://validator.swagger.io/?url=https://raw.githubusercontent.com/0xrawsec/whids/master/doc/admin.openap

WHIDS API documentation 1.0 OAS3

https://raw.githubusercontent.com/0xrawsec/whids/master/doc/admin.openapi.json

Servers

https://localhost:1520

Endpoint Management

GET /endpoints Get endpoints

PUT /endpoints Create a new endpoint

GET /endpoints/{uuid} Get information about a single endpoint

POST /endpoints/{uuid} Modify an existing endpoint

DELETE /endpoints/{uuid} Delete an existing endpoint

Artifact Search and Retrieval

GET /endpoints/artifacts Artifacts on all endpoints

GET /endpoints/{uuid}/artifacts Artifacts for a single endpoint

GET /endpoints/{uuid}/artifacts/{pguid}/{ehash}/{filename} Retrieve the content of an artifa

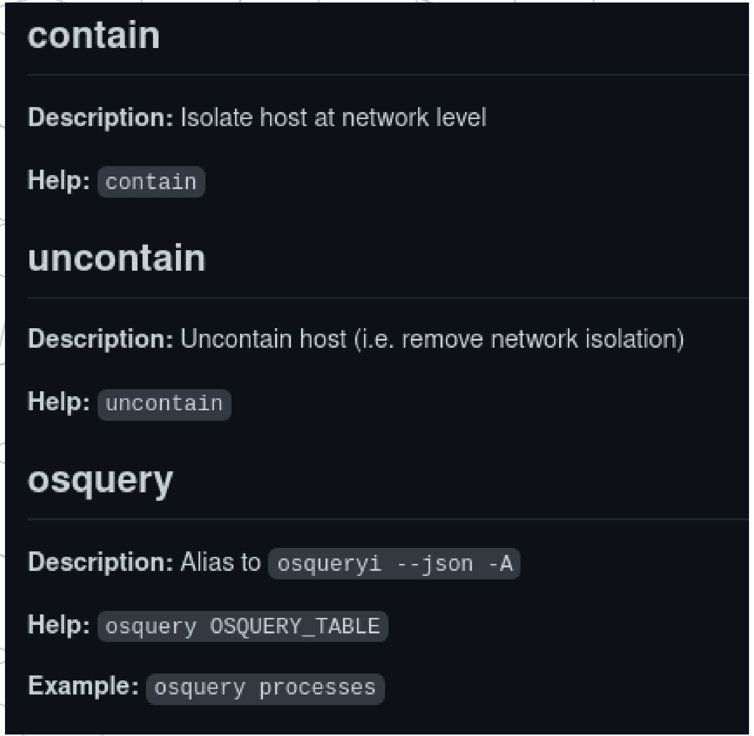
Detection Reports

GET /endpoints/reports Get all detection reports

Improved documentation (2)

Auto-generated documentation for EDR specific commands agents can execute

- Generated at every commit



contain

Description: Isolate host at network level

Help: `contain`

uncontain

Description: Uncontain host (i.e. remove network isolation)

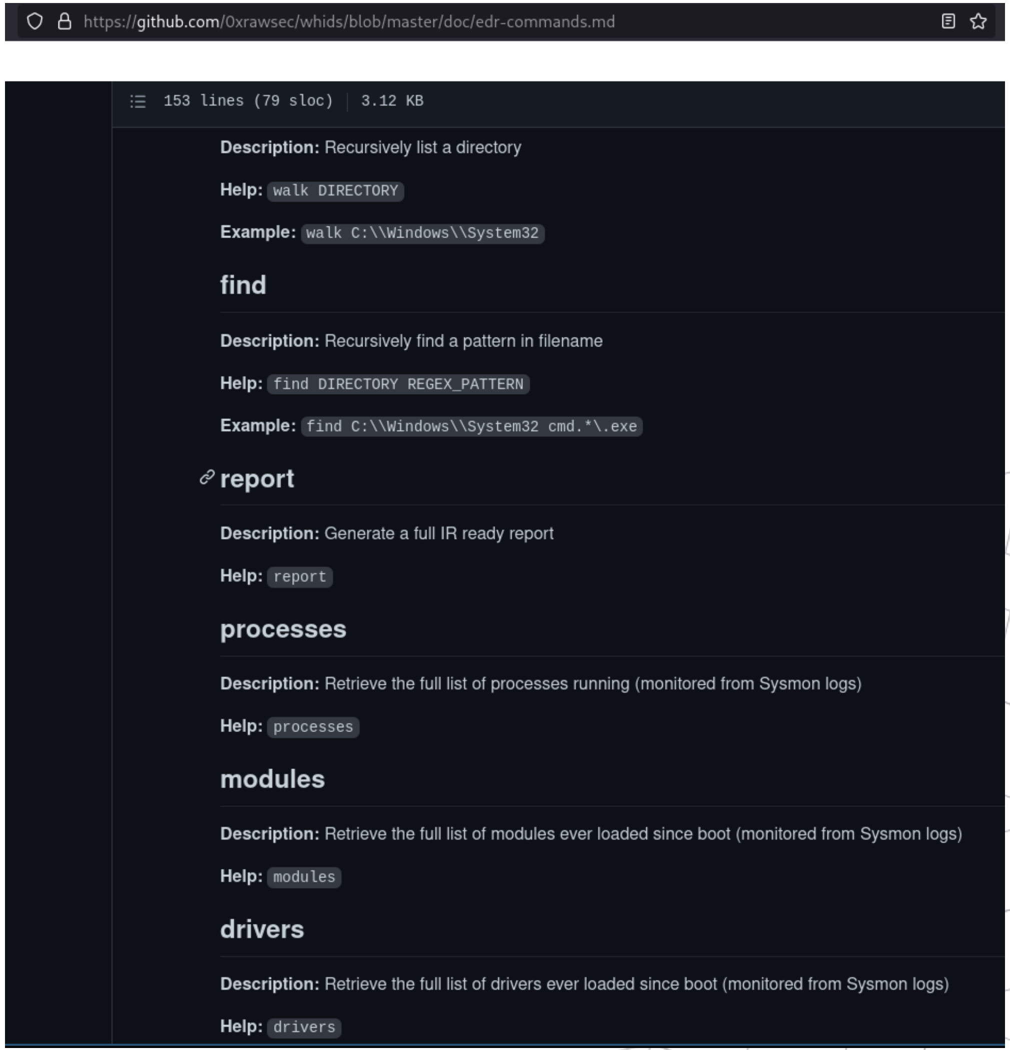
Help: `uncontain`

osquery

Description: Alias to `osqueryi --json -A`

Help: `osquery OSQUERY_TABLE`

Example: `osquery processes`



https://github.com/0xrawsec/whids/blob/master/doc/edr-commands.md

153 lines (79 sloc) | 3.12 KB

Description: Recursively list a directory

Help: `walk DIRECTORY`

Example: `walk C:\\Windows\\System32`

find

Description: Recursively find a pattern in filename

Help: `find DIRECTORY REGEX_PATTERN`

Example: `find C:\\Windows\\System32 cmd.*\.exe`

report

Description: Generate a full IR ready report

Help: `report`

processes

Description: Retrieve the full list of processes running (monitored from Sysmon logs)

Help: `processes`

modules

Description: Retrieve the full list of modules ever loaded since boot (monitored from Sysmon logs)

Help: `modules`

drivers

Description: Retrieve the full list of drivers ever loaded since boot (monitored from Sysmon logs)

Help: `drivers`

What you did/don’t/won’t see

A lot of invisible work (not bringing new “end-user” features):

- Getting rid of all “on-disk” configuration on manager’s side -> now any sort of configuration is done through HTTP API
- Continuous code refactoring
- Proper CI/CD pipeline design/improvement
- Improve/implement code coverage and test cases
- All the R&D behind: sometimes I spend days testing out stuff that will never see the light (mainly ETW related)
- Developing/fixing/improving dependencies:
  - gene (detection engine)
  - crony (scheduler) -> task scheduler used in agent
  - golog (logging) -> improved logging in agent and manager
  - toast (testing) -> testing library
  - sod (storage) -> pure golang database engine
  - golang-etw (event source) -> better performance and flexibility

What’s next ?

Short term: getting some traction

- Finalize release 😐
- Publish HowTos (open to suggestions)
- Publish blog posts / articles
- Make some use cases with malware samples

Long term: continue developing this project and others (always open source)

- I’d like to have time to port it to other OS -> more work since Sysmon is not portable
    1. Linux based
    2. Darwin based
- Write more plugins

How can you help ?

Do anything to make the project live:

- Develop a GUI :P
- Use/test the tool and give feedbacks, good or bad
- Make feature requests
- Talk about it if you think it is nice, and talk to me before telling anyone else it is crap (more constructive)
- Contribute to the code (Golang)
- Write plugins in PyWHIDS repo (Python) to integrate with other open source tools
- Give feedbacks, give feedbacks, give feedbacks
- If you don’t have time for all above, you can still sponsor the project on GitHub ☺

Thank you all !

Special thanks to @adulau, @gallypette and other anonymous supporters for believing in this project since the beginning and for boosting up my motivation

Contact via Twitter/Github **@0xrawsec**

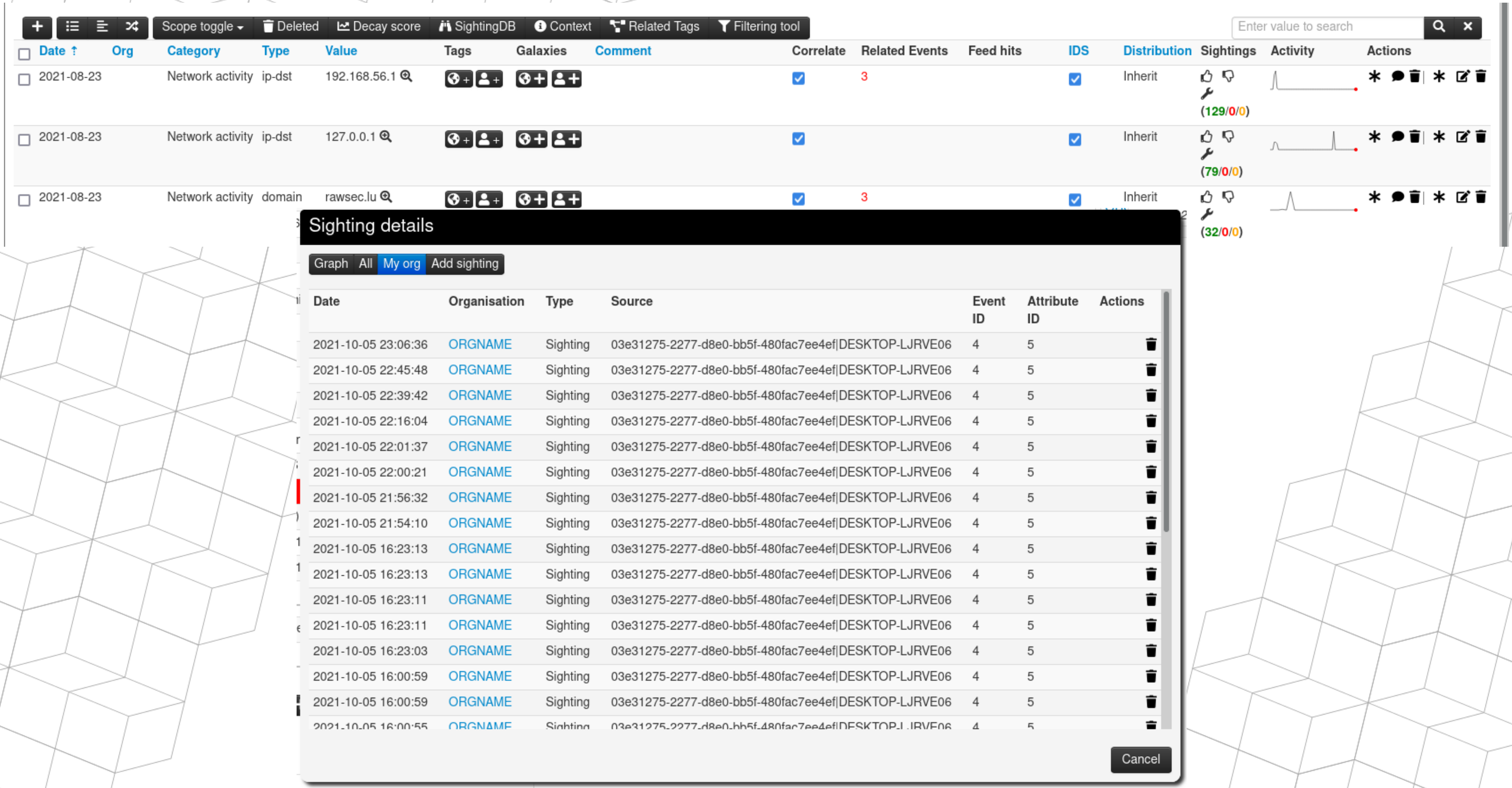
- Feel free to open issues, ask questions, give feedbacks/suggestions …

References:

- WHIDS: https://github.com/0xrawsec/whids
- PyWHIDS and EDR plugins: https://github.com/0xrawsec/pywhids
- golang-etw: https://github.com/0xrawsec/golang-etw
- Gene: https://github.com/0xrawsec/gene
- Gene rules: https://github.com/0xrawsec/gene-rules
- Gene Documentation: https://rawsec.lu/doc/gene/2.0

sightings.py

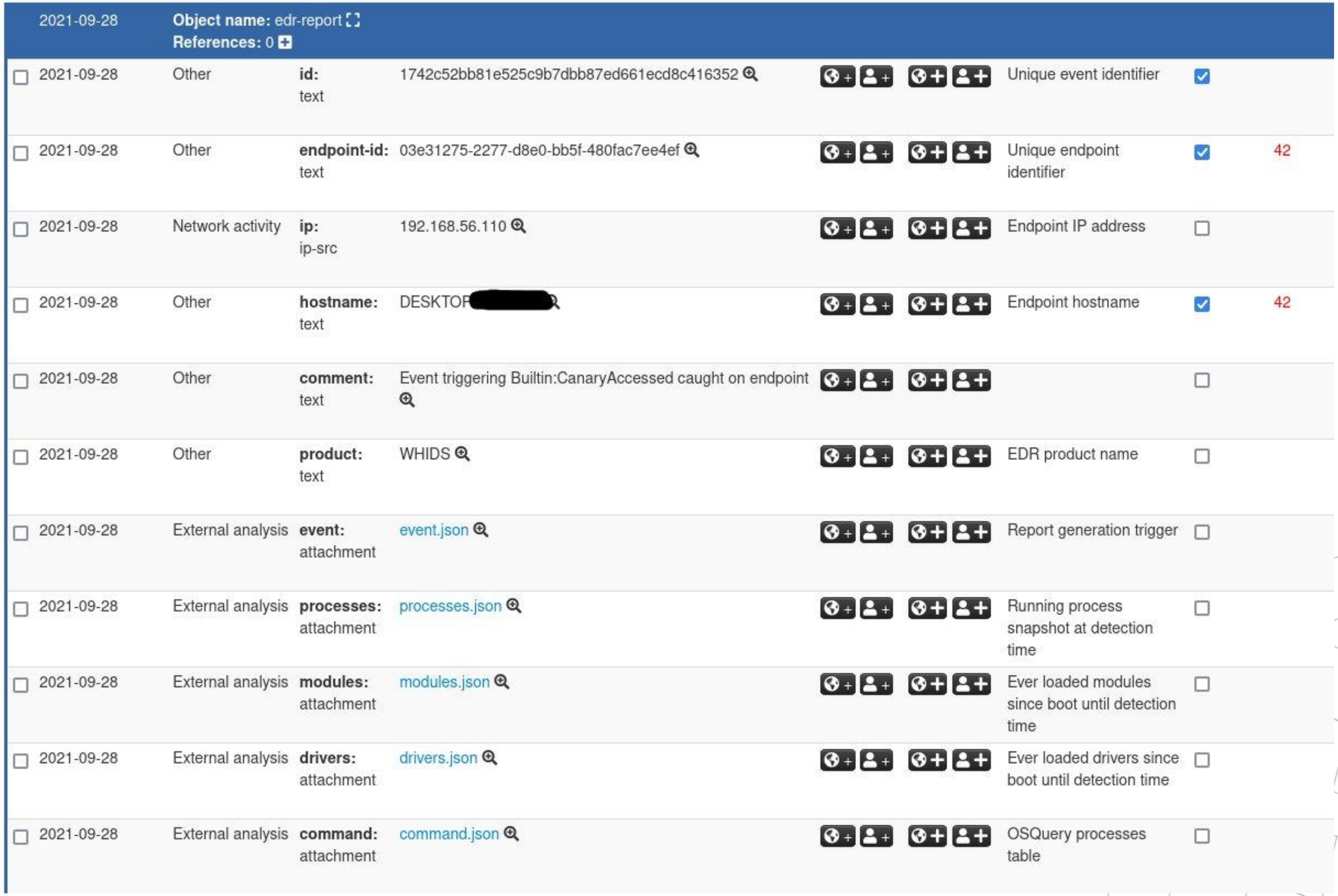
RawSec



☐	Date	Org	Category	Type	Value	Tags	Galaxies	Comment	Correlate	Related Events	Feed hits	IDS	Distribution	Sightings	Activity	Actions
☐	2021-08-23		Network activity	ip-dst	192.168.56.1				☑	3		☑	Inherit	(129/0/0)		
☐	2021-08-23		Network activity	ip-dst	127.0.0.1				☑			☑	Inherit	(79/0/0)		
☐	2021-08-23		Network activity	domain	rawsec.lu				☑	3		☑	Inherit	(32/0/0)		

Date	Organisation	Type	Source	Event ID	Attribute ID	Actions
2021-10-05 23:06:36	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	
2021-10-05 22:45:48	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	
2021-10-05 22:39:42	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	
2021-10-05 22:16:04	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	
2021-10-05 22:01:37	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	
2021-10-05 22:00:21	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	
2021-10-05 21:56:32	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	
2021-10-05 21:54:10	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	
2021-10-05 16:23:13	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	
2021-10-05 16:23:13	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	
2021-10-05 16:23:11	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	
2021-10-05 16:23:11	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	
2021-10-05 16:23:03	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	
2021-10-05 16:00:59	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	
2021-10-05 16:00:59	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	
2021-10-05 16:00:55	ORGNAME	Sighting	03e31275-2277-d8e0-bb5f-480fac7ee4ef|DESKTOP-LJRVE06	4	5	

reporting.py



	2021-09-28	**Object name:** edr-report **References:** 0					
□	2021-09-28	Other	**id:** text	1742c52bb81e525c9b7dbb87ed661ecd8c416352	Unique event identifier	☑	
□	2021-09-28	Other	**endpoint-id:** text	03e31275-2277-d8e0-bb5f-480fac7ee4ef	Unique endpoint identifier	☑	42
□	2021-09-28	Network activity	**ip:** ip-src	192.168.56.110	Endpoint IP address	□	
□	2021-09-28	Other	**hostname:** text	DESKTOP[redacted]	Endpoint hostname	☑	42
□	2021-09-28	Other	**comment:** text	Event triggering Builtin:CanaryAccessed caught on endpoint		□	
□	2021-09-28	Other	**product:** text	WHIDS	EDR product name	□	
□	2021-09-28	External analysis	**event:** attachment	event.json	Report generation trigger	□	
□	2021-09-28	External analysis	**processes:** attachment	processes.json	Running process snapshot at detection time	□	
□	2021-09-28	External analysis	**modules:** attachment	modules.json	Ever loaded modules since boot until detection time	□	
□	2021-09-28	External analysis	**drivers:** attachment	drivers.json	Ever loaded drivers since boot until detection time	□	
□	2021-09-28	External analysis	**command:** attachment	command.json	OSQuery processes table	□	