E2E processing of malware samples using open source technologies

Pass the Salt - July 2025

DATADOG

whoami(we)

Matt Muir
Security Researcher

Fred Baguelin
Security Researcher

Agenda

01 What is Malhound?

02 Leveraging FOSS for malware analysis

03 Analysing Skidmap

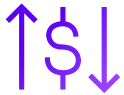
04 Analysing a malicious model - live MalHound demo

05 Conclusion

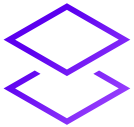
What is MalHound?

Problem Statement(s)

- We routinely collect and analyse large volumes of malware samples



- Traditional sandbox solutions are expensive and limited in a cloud-first world



- We need a mechanism for quickly extracting IoCs so we can deliver threat intel to customers

What do we currently have?

YETI

Open Source TIP

- Highly-extensible
- Helps having a contributor on the team 😛
- Ingests all honeynet data

What do we currently have?

Workload Protection Agent

eBPF Security Agent

- Excellent for monitoring Linux hosts and containers
- Works well in the environments our customers use
- Malware cannot easily evade kernel-level monitoring

What do we currently have?

Honeynet

Multi-honeypot Infrastructure

- Heavily based on Docker
- Incoming feed of malware samples/malicious images etc
- Allows us to gather cloud-specific threat intel

What do we currently have?

AssemblyLine4

Malware Analysis Pipeline

- Developed by the Canadian CERT
- Processes queues of files and hands them off to other services for analysis

💡 Maybe we can unite these technologies 💡

Leveraging FOSS for Malware Analysis

Assemblyline4

Front-end to pipeline

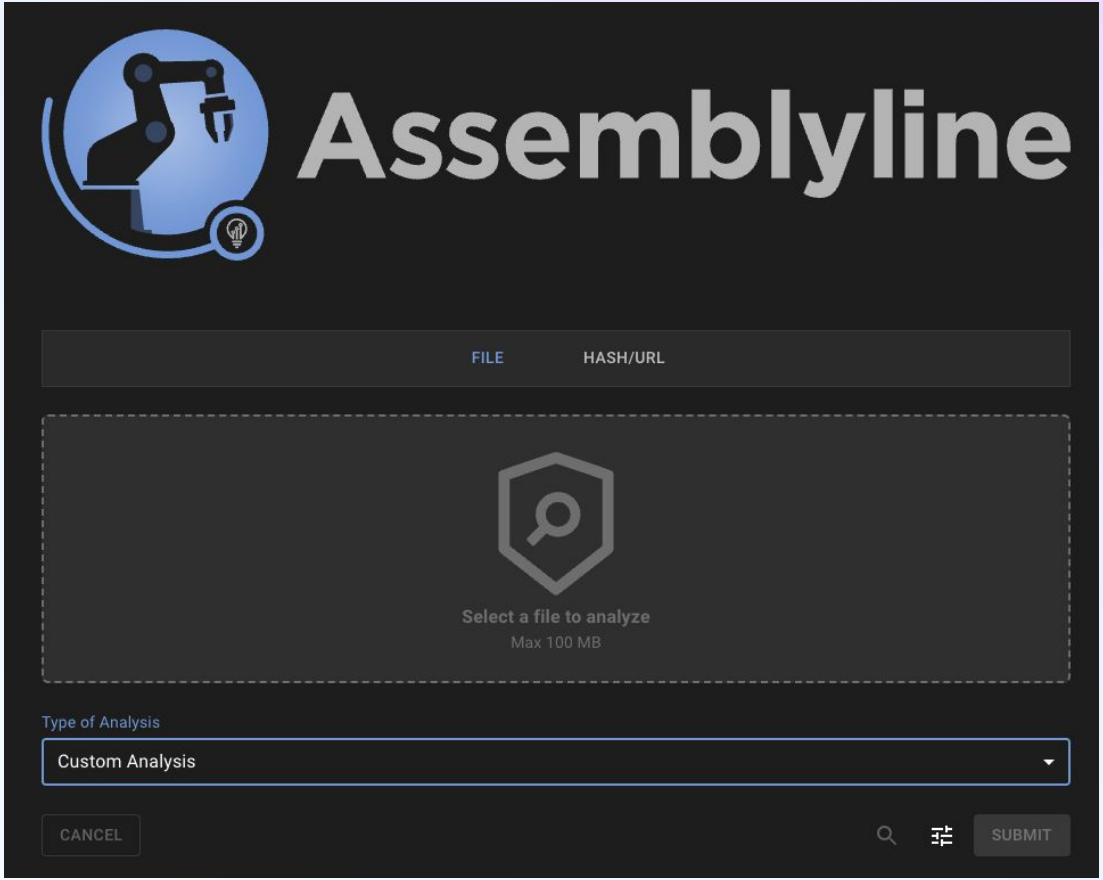
- Samples/models submitted via a web UI

Services

- Jobs in the pipeline
- Static/dynamic
- Written in Python

Conveyor-belt design

- Services executed in series
- Unique ID added to file, identify during processing



Could we use AL4 and our eBPF agent in combination as a “sandbox”?

MalHound

Detonate malicious samples

from ELF to models or docker images

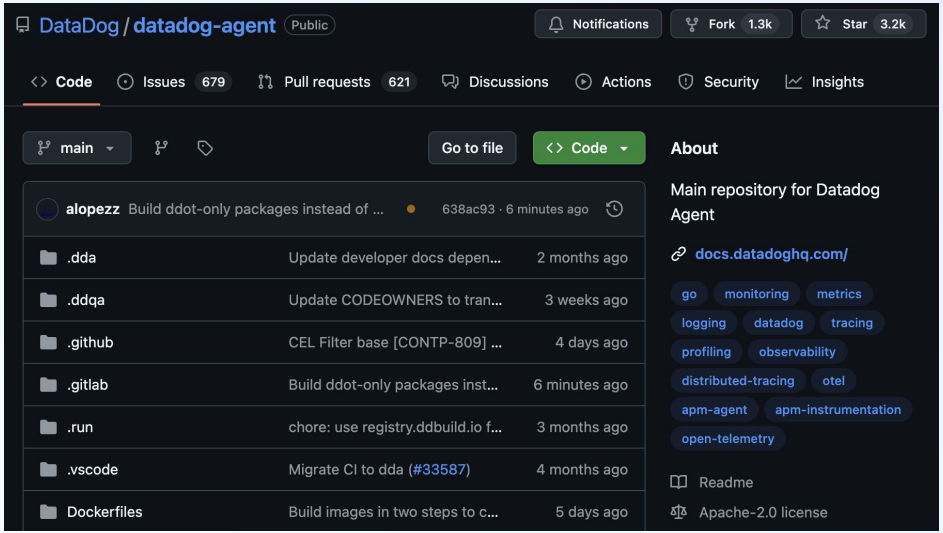
Rely on docker containers as execution unit

Automates all the burden of old school sandbox

Use Datadog open source agent

To profile processes activities (files, netflows, syscalls, …) and applications

```
2025-02-07 15:41:27,178 - detonator - INFO - Container cws-agent already running (id: 9f2c41c1d59b).
2025-02-07 15:41:27,220 - detonator - INFO - Starting malhound execution...
2025-02-07 15:41:27,220 - detonator - INFO - Bootstraping BaseSandbox sample detonation
2025-02-07 15:41:27,306 - detonator - INFO - Container cws-agent already running (id: 9f2c41c1d59b).
2025-02-07 15:41:57,307 - detonator - INFO - Initializing detonator container
2025-02-07 15:41:58,968 - detonator - INFO - Container c340e3d3ae7f769b4e88204dd08aa0f7b0145dffafe164d8e09c39b5a6
2025-02-07 15:41:59,260 - detonator - INFO - Container c340e3d3ae7f769b4e88204dd08aa0f7b0145dffafe164d8e09c39b5a6
2025-02-07 15:41:59,261 - detonator - INFO - Bootstraping container c340e3d3ae7f769b4e88204dd08aa0f7b0145dffafe16
2025-02-07 15:41:59,261 - detonator - INFO - Pulling image ubuntu:latest
2025-02-07 15:42:00,771 - detonator - INFO - Pulling from library/ubuntu
2025-02-07 15:42:01,179 - detonator - INFO - Pulling fs layer 5a7813e071bf...
2025-02-07 15:42:02,082 - detonator - INFO - Verifying Checksum
2025-02-07 15:42:02,082 - detonator - INFO - Download complete for ubuntu:latest. Extracting...
2025-02-07 15:42:04,972 - detonator - INFO - Pull complete
2025-02-07 15:42:05,103 - detonator - INFO - Digest: sha256:72297848456d5d37d1262630108ab308d3e9ec7ed1c3286a32fe0
2025-02-07 15:42:05,148 - detonator - INFO - Status: Downloaded newer image for ubuntu:latest
2025-02-07 15:42:07,520 - detonator - INFO - Container c340e3d3ae7f769b4e88204dd08aa0f7b0145dffafe164d8e09c39b5a6
2025-02-07 15:42:08,010 - detonator - INFO - Starting container c340e3d3ae7f769b4e88204dd08aa0f7b0145dffafe164d8e
2025-02-07 15:42:09,034 - detonator - INFO - Container c340e3d3ae7f769b4e88204dd08aa0f7b0145dffafe164d8e09c39b5a6
 running.
2025-02-07 15:42:10,348 - detonator - ERROR - Error while running pre-run script: b'\nWARNING: apt does not have
ate;\n'
2025-02-07 15:42:10,431 - detonator - INFO - Stopping activity dump for 5dcdb54a90085530cd4c7e954d3c551ceddd0546b
2025-02-07 15:42:11,769 - detonator - INFO - Waiting 5 seconds for dump termination...
2025-02-07 15:42:26,770 - detonator - INFO - Listing activity dumps...
2025-02-07 15:42:28,081 - detonator - INFO - Running command ./c340e3d3ae7f769b4e88204dd08aa0f7b0145dffafe164d8e0
39b5a6d0d7cb
2025-02-07 15:42:28,081 - detonator - INFO - Let container c340e3d3ae7f769b4e88204dd08aa0f7b0145dffafe164d8e09c39
2025-02-07 15:42:28,081 - detonator - INFO - [SAMPLE OUTPUT START]
```



docker

DATADOG

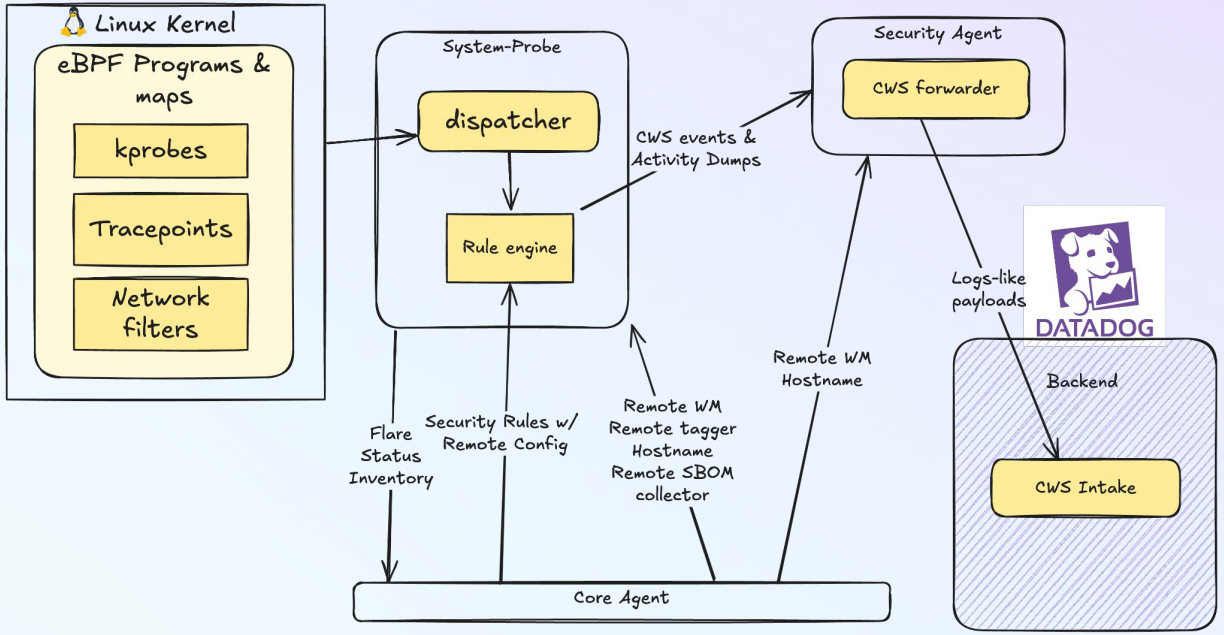
Datadog agent 101

Low level tracers

- eBPF (kernel land)
- System-probe (userland)

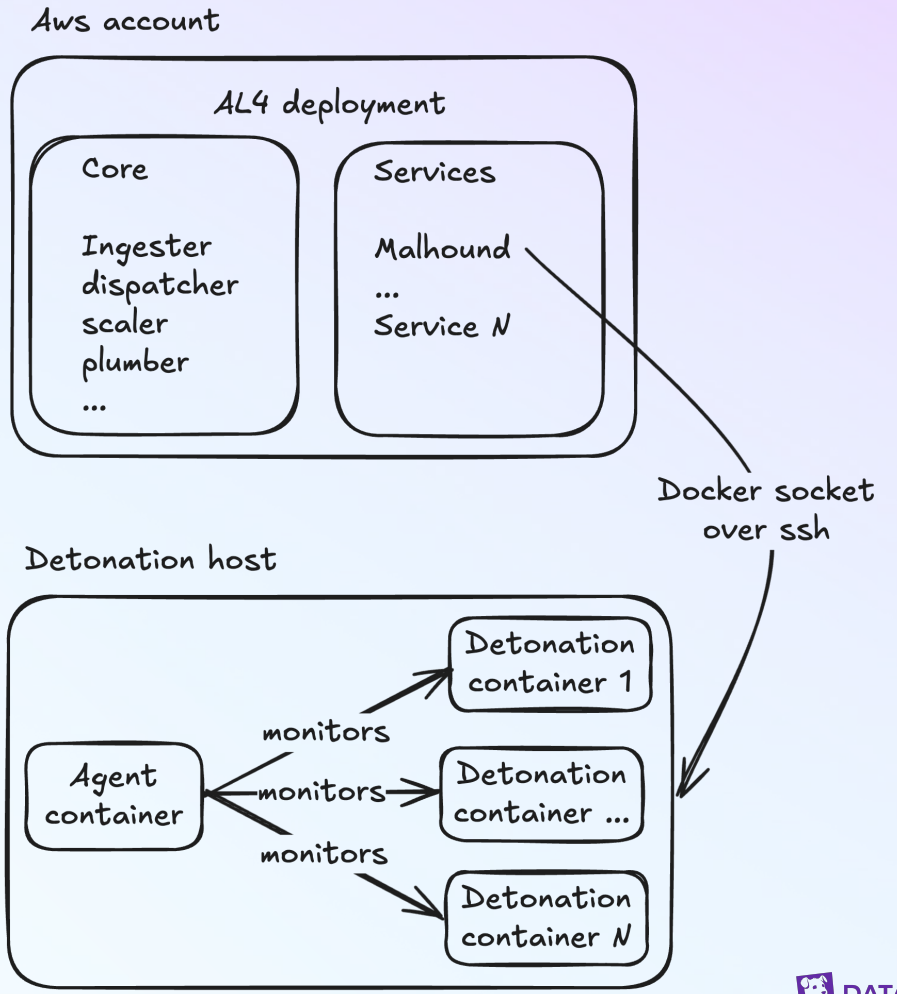
Application tracers

- Profile application for several programming languages
- Support runtime injection for Python or Nodejs for example
- Compatible with OTel



Uniting AL4 & Malhound

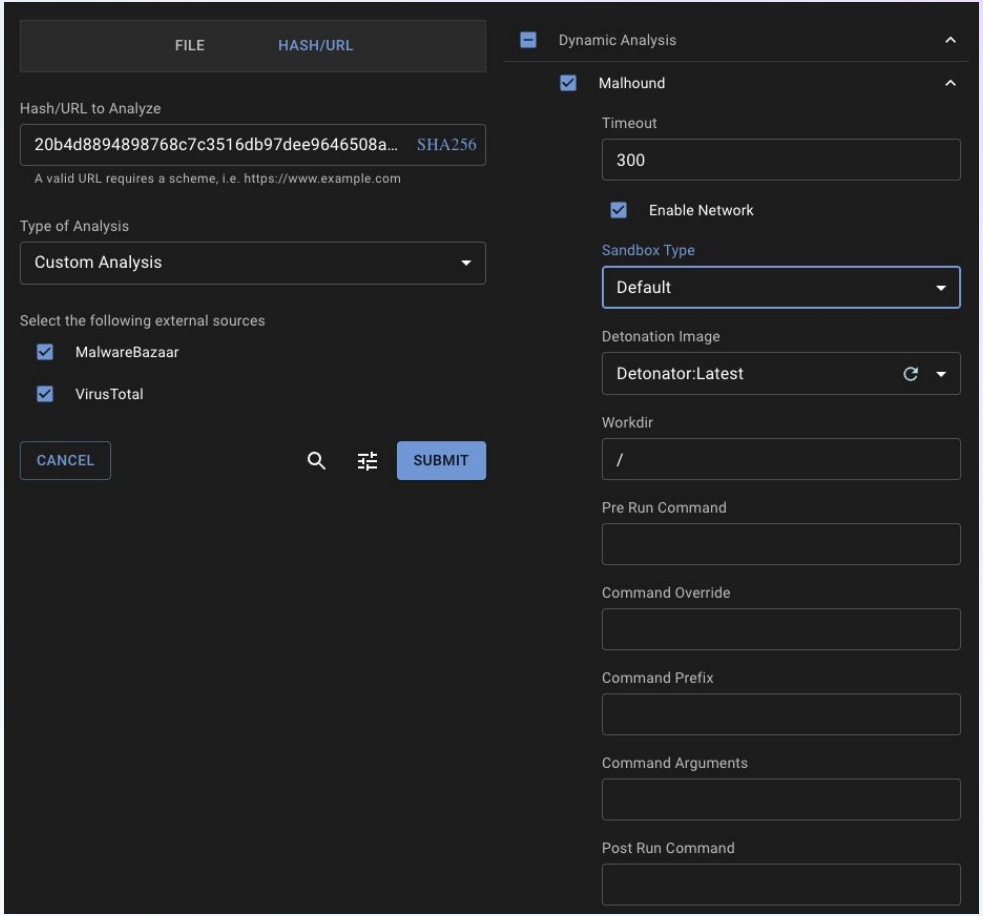
- **Rely on AL4 to handle storage / analysis / UI / scalability**
- **Add Malhound as a dynamic analysis service**
- **Renders Malhound results in AL4:**
  - IOCS
  - Process tree
  - Syscalls table
  - Files table
  - Netflows table



How does this look to an analyst?

Custom submission parameters

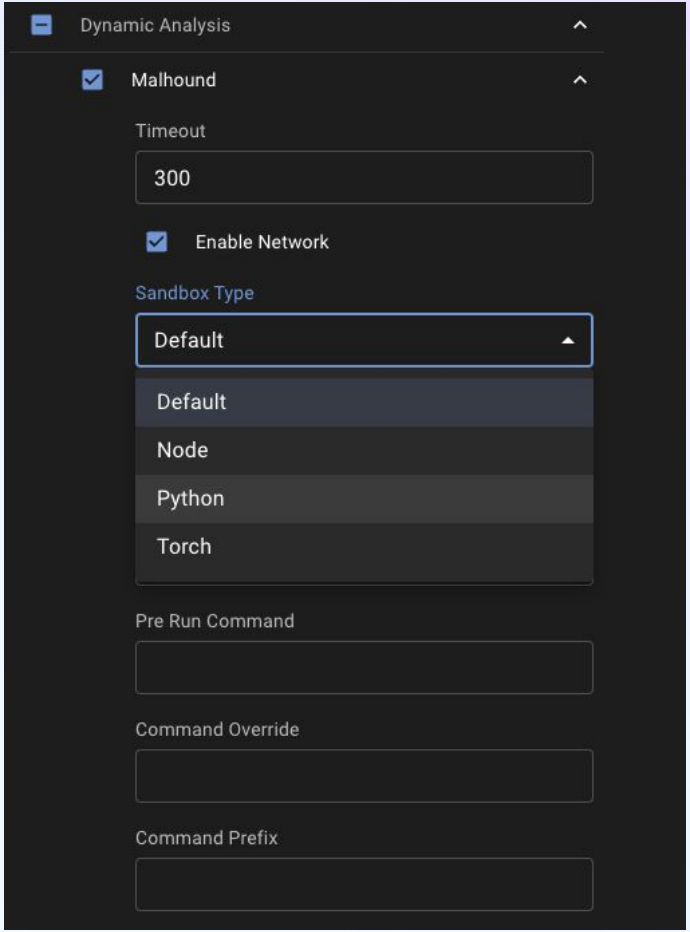
- Execute sample with argument
- Override container start command
- Configure networking/execution timeout
- Apply custom policies and filtering expressions



Detonation Environments

Environment is configurable

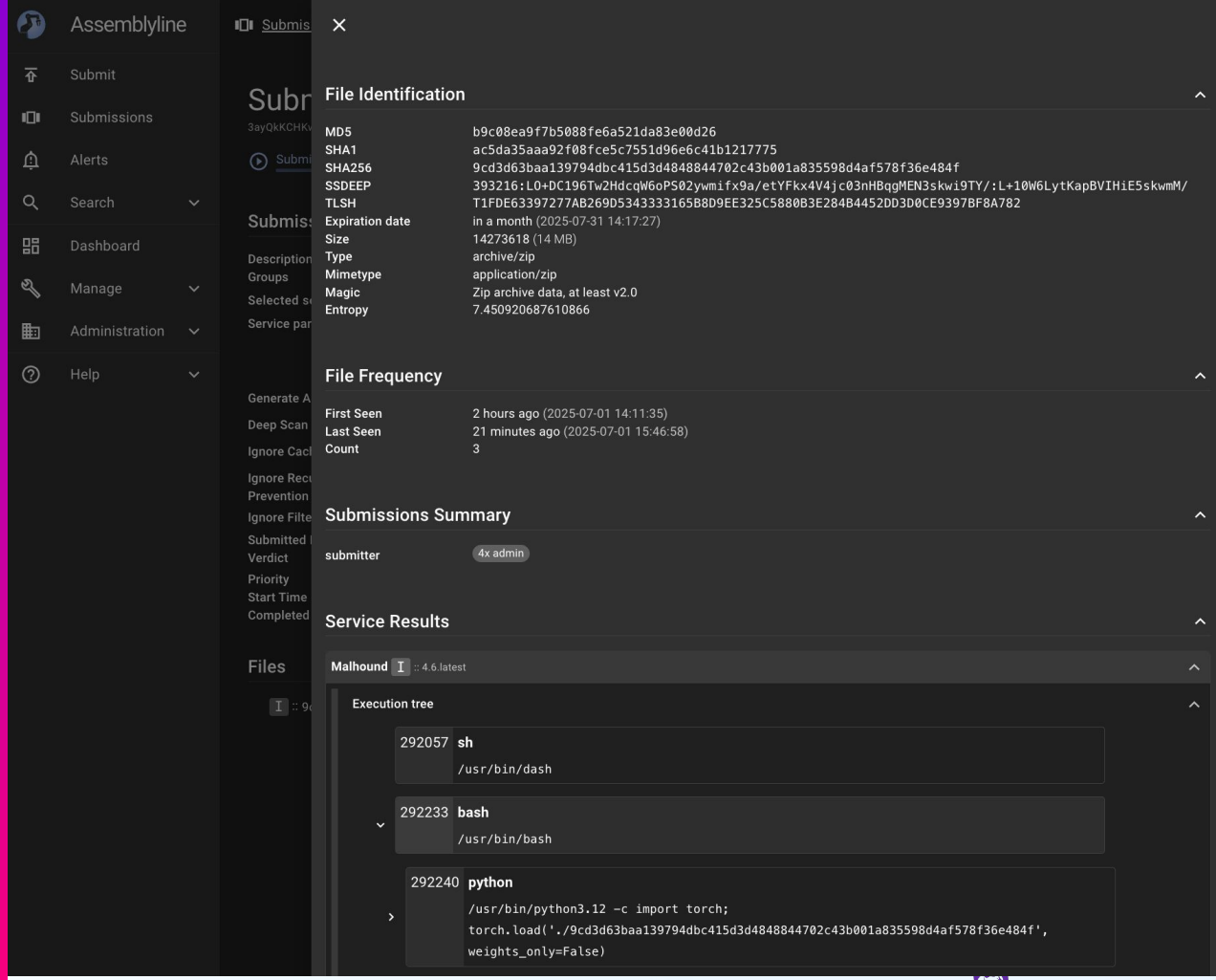
- Support for ELFs, Python and Node scripts along with PyTorch models
- Guarddog - requirement to analyse heavily-obfuscated nodeJS and Python scripts
- Can easily modify container to introduce new detonation environments



Malhound AL4 output

Rely on activity dumps

- Displays process tree
- Can test detection engineering rules
- Displayed in Datadog for slide but output from MalHound is all JSON
- Easily trace malware's behaviour
- Similar to a sandbox, but tracing done via eBPF or a profiler



Analysing Skidmap

Skidmap Overview

Cryptojacking malware targeting Linux

- Originally discovered in 2019 by Trend Micro

Hides mining activities using malicious kernel modules (rootkits)

- `getdents()` hooking - hide dropped files
- Fakes CPU and network usage statistics to conceal mining

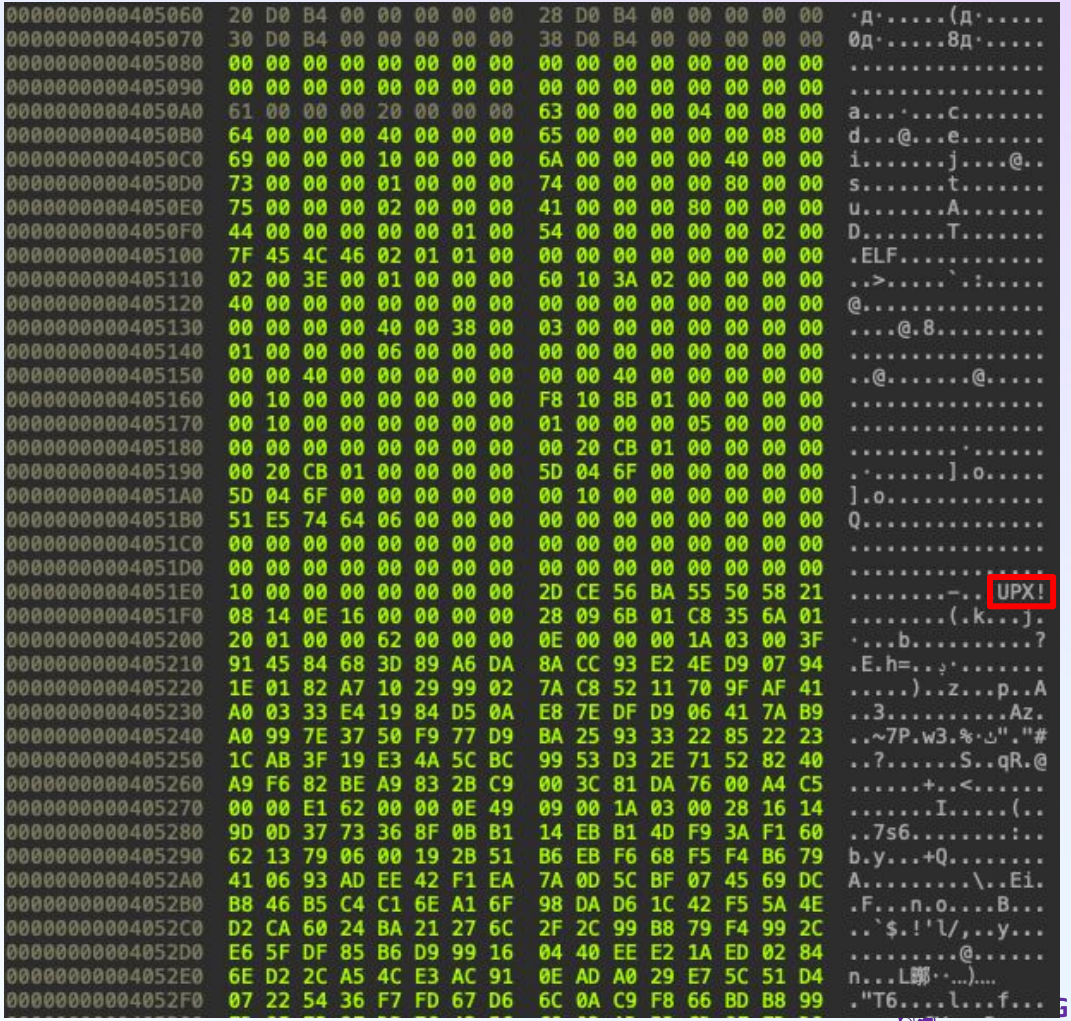
Constantly evolving family, new variants frequently discovered

- Trustwave variant in 2023 - Redis targeting
- We notice Skidmap activity in our own honeypot in Q2 2025

Skidmap Analysis

selinuxdefconed

- UPX-packed x86-64 ELF
- Primary payload for 2025 campaign
- Multiple embedded payloads
- Malformed UPX header
- upx_dec works for the “parent” but not the rest
- Could extract each payload and unpack but let’s use MalHound!



Skidmap Analysis

MalHound Raw JSON Output

- Quickly identify syscalls used by the malware
- Useful for binary triage
- Looks like the binary writes additional payloads

```
"syscall_nodes": [
  { "image_tags": ["latest"], "syscall": 0 }, // read
  { "image_tags": ["latest"], "syscall": 1 }, // write
  { "image_tags": ["latest"], "syscall": 3 }, // close
  { "image_tags": ["latest"], "syscall": 4 }, // stat
  { "image_tags": ["latest"], "syscall": 5 }, // fstat
  { "image_tags": ["latest"], "syscall": 9 }, // mmap
  { "image_tags": ["latest"], "syscall": 10 }, // mprotect
  { "image_tags": ["latest"], "syscall": 11 }, // munmap
  { "image_tags": ["latest"], "syscall": 12 }, // brk
  { "image_tags": ["latest"], "syscall": 16 }, // ioctl
  { "image_tags": ["latest"], "syscall": 17 }, // pread64
  { "image_tags": ["latest"], "syscall": 21 }, // access
  { "image_tags": ["latest"], "syscall": 87 }, // unlink
  { "image_tags": ["latest"], "syscall": 90 }, // chmod
  { "image_tags": ["latest"], "syscall": 158 }, // arch_prctl
  { "image_tags": ["latest"], "syscall": 218 }, // set_tid_address
  { "image_tags": ["latest"], "syscall": 231 }, // exit_group
  { "image_tags": ["latest"], "syscall": 257 }, // openat
  { "image_tags": ["latest"], "syscall": 318 }, // getrandom
  { "image_tags": ["latest"], "syscall": 334 } // rseq
],
"network_devices": []
```

Key Insights for Malware Analysis

1. **The malware process (PID 140643) shows significantly more syscall diversity** - typical of malicious behavior
2. **File access is tracked separately** - you can correlate file access patterns with syscall usage
3. **Syscalls 87 (unlink) and 90 (chmod)** in the malware process suggest file manipulation
4. **Syscall 257 (openat)** indicates modern file opening operations

Skidmap Analysis

Additional payloads identified

- Sample is a loader that writes out the following files:
  - `reviews`
  - `mldconfigs`
  - `kmod` (masquerade)
- Hashes displayed for each
- Additional payloads self-extracted from main binary
- TTP overlap with 2023 variant


```
"image_tags": ["latest"],
"name": "reviews",
"is_pattern": false,
"file": {
  "uid": 0,
  "user": "root",
  "gid": 0,
  "group": "root",
  "mode": 33188,
  "ctime": "1750691192431070471",
  "mtime": "1750691192431070471",
  "mount_id": 2186,
  "inode": "2152064",
  "in_upper_layer": false,
  "path": "/usr/bin/reviews",
  "basename": "reviews",
  "filesystem": "overlay",
  "package_name": "",
  "package_version": "",
  "package_srcversion": "",
  "hashes": [
    "sha1:6306f546bbb787f7aad8c32089262ec984b9e2cc",
    "sha256:8dee19b985b6d50da3a44063c91c6d9dc91640f57e58e31bdddccc90aa5b54c8",
    "md5:6bdb3c97f821e6262626c93e71c4f27c"
  ],
  "hash_state": "DONE"
```

Detonating a Malicious Model - Live Demo

What's next

Open source Malhound and AL4 service

Soon™ 🙃

Improve integration between AL4 and Yeti

Yeti plugin to submit command_line to AL4

AL4 plugin to submit selected IOCs to Yeti

Support other tracers and container runtimes

Kunai, Tracee, Falco… And move from default docker runtime to kata container.

Takeaway

Relying on Open Source is important when dealing with sandboxes

You need to understand how things work to be sure you're not missing something.

Cloud environment is not well supported in the sandbox ecosystem, FOSS helps to build your own pipeline.

We needed support for packages, docker images, models, …

FOSS community is always helpful!

AL4 community was super helpful and the project is very well thought.

Q&A