CIRCL
Computer Incident
Response Center
Luxembourg

# Kunai Introduction

## An Open-Source Threat-Detection Tool for Linux

https://github.com/kunai-project

Quentin JEROME

2025/07/09

CIRCL Virtual Summer School — 2025

# Specific Terminology

- **Security event**: an event happening on a system which may indicate a potential security incident

  **Example**: A log entry that indicates the execution of a given binary

- **Security monitoring tool**: a tool monitoring a system or an infrastructure and generating security events for analysis.

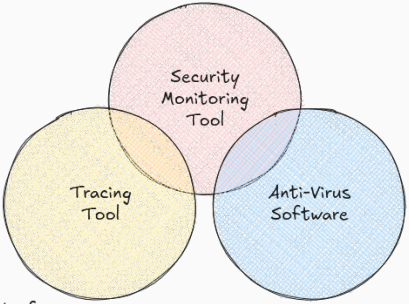
  **Example**: Intrusion Detection Systems (IDS), Endpoint Detection and Response (EDR)

- **Security visibility**: the ability to monitor,detect and analyze security events

  **Example**: A company using a security monitoring software increases its security visibility

# Different Software for Different Purposes

Goal: be a great source of events to improve security visibility




Goal: be good at detecting known threats

Non-Goal: provide many security events
- not good for security analysts
- hard to understand why the software took this or that decision

Goal: provide events for performance diagnosis / debugging
- some event can be used as security event

Non-Goal: provide security sound data
- information is limited

# So What is Kunai?

**Kunai** aims at being a **Linux** based **security monitoring** tool helping **individual/organisations** to increase their **security visibility** by generating several **security events**

A bit more than this actually...

# Brief History

Project[1] started **end-2022** as a "good first Rust project":

**12/2022 - 01/2024**: worked on it under my own company
**since 01/2024**: joined **CIRCL** and working on the project in the context of an EU co-funded project

Why starting such a project:

- I was disappointed by **Sysmon for Linux** for many reasons
- Yet there are many good ideas in Sysmon and I think we can do much better by:
  - getting rid of XML (for configuration and events)
  - do not transpose something primarily done for Windows into Linux

[1] `https://github.com/kunai-project/kunai`

# What can we do with Kunai?

Get high quality **security events** to perform **threat detection/hunting**

- Monitor many **events**[2] (execve, shared object loaded, BPF programs loaded, files read/write/delete . . . )
- Events comes with the following:
  - Relevant information to build solid **behavioral detections**
  - In chronological order
  - Grouping capability through a uuid
  - Parent/child tracking
  - Enriched with data from previous events (i.e. network connect/send)
- Accurately track security events generated by **Linux container** solutions

[2] https://why.kunai.rocks/docs/category/kunai—events

# Example of execve event


```
{
  "data": {
    "ancestors": "/usr/lib/systemd/systemd|[...]r/bin/bash",
    "parent_command_line": "bash -ec [...] rm -rf $t; done\""
    "parent_exe": "/usr/bin/bash",
    "command_line": "mktemp -d -p /tmp/trash",
    "exe": {
      "path": "/usr/bin/mktemp",
      "md5": "0d7660dac3bffd6b76d3054da0fa1216",
      "sha1": "817329148a765bc2dc82baa230638d1987d7a28
4",
      "sha256": "f32938cf25ddd6f6800a8e9b406595534d0eb
27993587bbdeee2e83dd97d8406",
      "sha512": "22d716d4e16492928ec90380d8c6d4749a00
82ffa04630355a1d9a59bd3d196e83169223354a1edd28c5bbec1
37da96ccc6f42558e30cfe2a4d456a0b4c50fba",
      "size": 39144,
      "error": null
    }
  },
  "info": "[...]"
}
```



```
{
  "data": "...",
  "info": {
    "host": {
      "uuid": "c030b40d-0eab-417b-b33a-22d952357984",
      "name": "hal",
      "container": {
        "name": "ubuntu-kunai-test",
        "type": "docker"
      }
    },
    "event": {
      "source": "kunai",
      "id": 1,
      "name": "execve",
      "uuid": "87f5cb64-11ca-fd34-26af-b47f84132543",
      "batch": 131
    },
    "task": {
      "name": "ls",
      "pid": 1981502,
      "tgid": 1981502,
      "guuid": "cd5415c4-7750-0000-7c85-4ef53e3c1e00",
      "uid": 1000,
      "gid": 1000,
      "namespaces": {
        "mnt": 4026531841
      },
      "flags": "0x400000",
      "zombie": true
    },
    "parent_task": {
      "...": "...",
    },
    "utc_time": "2024-11-19T09:55:53.138247256Z"
  }
}
```

# Detection Rules

**Goal**: detect an **attack/suspicious** pattern

```
# name of the rule
name: mimic.kthread
# default type is detection so the following line is not mandatory
type: detection
match-on:
    events:
        # we match on kunai execve and execve_script events
        kunai: [execve, execve_script]
matches:
    # 0x200000 is the flag for KTHREAD
    $task_is_kthread: .info.task.flags &= '0x200000'
    # common kthread names
    $kthread_names: .info.task.name ~= '^(kworker)'
# if task is NOT a KTHREAD but we have a name that
# looks like one we want the rule to kick-in
condition: not $task_is_kthread and $kthread_names
# severity is bounded to 10 so it is the maximum score
severity: 10
```

Any rule may encode **actions** to take on a given detection

- **kill**: kill the process triggering the detection
- **scan-files**: see if file matches malicious **byte patterns**

# Filtering Rules

**Goals**: save resources & keep **context**

Two ways to filter logs:

1. **toggle** a **boolean** switch in configuration

   Simple and fast, but not granular

2. use **fine grained** filtering
   - fine granularity
   - without context a security alert is **useless**!

```
name: log.mprotect_exec
type: filter
match-on:
    events:
        # applies on kunai mprotect_exec
        kunai: [ mprotect_exec ]
matches:
    # exe matches regex
    $browser: .data.exe.path ~= '/usr/lib/(firefox/firefox|chromium/
chromium)'
# if exe is neither firefox nor chromium
condition: not $browser
```

# Raise Alerts on IoCs

Indicator of Compromise (**IoC**): an artifact observed on a network or a system that indicates a computer intrusion

> **Examples**: domain name, IP address, file name, etc.

Kunai uses a straightforward **IoC format**

```
{
  "type": "domain",
  "uuid": "07574cee-4f0e-4fb7-8267-b9dcdbe0c919",
  "source": "CIRCL OSINT Feed",
  "value": "fdh32fsdfhs.shop",
  "severity": 7
}
```

1. kunai perfectly knows which field of its events can be an IoC
2. so it takes only a few lookups (per events) in a **hash map**

This make **IoC scanning** very fast and not depending on the number of **IoCs** being loaded

# Integration with other OSS

# IoC Provider Integration

- So far it is integrated with **MISP**[3] through **misp-to-kunai.py** [4]
  - Can be configured to ingest **MISP feeds** (**no** API key needed)
  - Can be configured to export events from a given MISP instance (API key **required**)
  - Able to run as a service to regularly pull updates

This script lives in a repository[5] where you can find other **tools** (mainly written in Python)

[3] MISP
[4] misp-to-kunai.py
[5] kunai tools repository

# File Scanning Using Yara

Recently **Kunai** has integrated a **Yara** scanning engine

What is **Yara**[6]:

- a solid project maintained by **VirusTotal**[7]
- rule format to **match patterns** within file
- command line tool and a library **to scan files** with a set of rules
- it is a **well known** format used to share **malware** detection rules

[6] Yara-x project
[7] VirusTotal

# Integration with Network-Analysis Tools

**Goal**: being able to investigate **system events** from **network events** and **vice versa**

Corelight designed a **network flow hashing algorithm** named **Community ID**[8]

Several **network traffic** analysis tools use it: Corelight / Zeek, Suricata, Wireshark, etc.

And now **Kunai** too, in all the network related events it generates

[8] Community ID specification

# Conclusions

# Takeaways

- **Kunai** is an **open-source** Linux-based security monitoring tool with **detection capabilities**
- Tracks a wide range of **security events** with detailed context to enable **behavioral analysis**
- Embeds a **rule engine** to implement **detection** or **filtering** rules
- Integrates with other **OSS** projects: **MISP**, **Yara**, **Zeek**, **Suricata**
- **Kunai** can be used for several purposes:
  - Enhance **security visibility**
  - Implement **threat hunting/detection** strategies
  - Bring valuable insights for **incident response**

# Final Words

Thank you all for your attention

It is time to ask questions!

**References:**

- Project: `https://github.com/kunai-project/`
- Documentation: `https://why.kunai.rocks/docs/quickstart`
- Tools: `https://github.com/kunai-project/pykunai`