



Rule/Query's Decision Tree summarization with AI

Uncoder AI

In production

Q Detection Rules

Elastic Stack Query (EQL) ▼

Select Platform ▼



TRANSLATE ▼

```
1 from logs-endpoint.events.process-* | keep @timestamp, host.os.type, event.type, event.action,
  process.name, process.args, process.command_line, agent.id, host.name | where @timestamp >
  now() - 1 hours | where host.os.type == "linux" and event.type == "start" and event.action
  == "exec" and ( (process.name in ("base64", "base64plain", "base64url", "base64mime",
  "base64pem", "base32", "base16") and process.command_line like "*-d*") or (process.name ==
  "openssl" and process.args == "enc" and process.args in ("-d", "-base64", "-a")) or (process
  .name like "python*" and (process.args == "base64" and process.args in ("-d", "-u", "-t"))
  or (process.args == "-c" and process.command_line like "*base64*" and process.command_line
  like "*b64decode*") ) or (process.name like "perl*" and process.command_line like "
  *decode_base64*") or (process.name like "ruby*" and process.args == "-e" and process
  .command_line like "*Base64.decode64*") ) | stats cc = count(), agent_count = count_distinct
  (agent.id), host.name = VALUES(host.name), agent.id = VALUES(agent.id) by process.name,
  process.command_line | where agent_count == 1 and cc < 15 | sort cc asc | limit 100
```

AI-generated Decision Tree

The Elastic Query (EQL) you've provided is designed to detect potential base64 decoding activities in Linux systems. Here's a breakdown of the decision tree embedded within this query:

Initial Filtering

- Time Frame:** The query starts by filtering events from the last hour (`@timestamp > now() - 1 hours`).
- Linux Events:** It then narrows down to only include events where the host operating system is Linux (`host.os.type == "linux"`).
- Event Type and Action:** Further filtering includes only `start` type events with an `exec` action (`event.type == "start" and event.action == "exec"`).

Specific Process Detection

The core of the decision tree involves detecting specific processes that could be involved in base64 decoding, categorized by process name. The conditions are as follows:

- Base64 Utilities:**
 - Processes named `base64`, `base64plain`, `base64url`, `base64mime`, `base64pem`, `base32`, or `base16`.
 - These processes must have a command line containing `-d*`, suggesting a decode operation.
- OpenSSL:**
 - The process name is `openssl`.
 - It must be used with the `enc` argument and either `-d` (decode), `-base64`, or `-a` (which in some contexts might imply base64 encoding, but here it's part of decoding conditions).
- Python:**

Rule/Query's Decision Tree summarization with AI

When dealing with complex threat detection logic, clarity is a necessity. Uncoder AI analyzes a query/rule and explains how it works step by step, with all the embeddings, branches and other intricate logic. For this purpose, Uncoder AI uses Llama 3.3 customized for detection engineering and threat intelligence processing, hosted at SOC Prime SOC 2 Type II private cloud for maximum security, privacy, and IP protection.

- Specific concise explanations
- Multiple languages supported
- Data doesn't leave SOC Prime's infrastructure

Rule/Query's Decision Tree summarization with AI

48 languages supported in production

