CIRCL
Computer Incident
Response Center
Luxembourg

Fearless File Identification

Rust based implementation of libmagic

https://github.com/qjerome/magic-rs

Quentin JÉRÔME — quentin.jerome@circl.lu — qjerome@infosec.exchange

October 21, 2025

hack.lu

Who knows what libmagic is?

What is libmagic?

libmagic is a core library for identifying file types using "magic numbers" and other file signatures.

- Originally part of the `file` command in Unix-like systems.
- Detects file formats by examining:
  - Magic numbers (header signatures)
  - Searching specific content at given offsets
    - Integers
    - Strings
    - Regex
    - ...
- Used in security, forensics, and file management tools ... maybe somewhere in your code

What is my issue (purely personal) with libmagic?

I need to use it a program which must run as **root** → `https://why.kunai.rocks`

- Kunai is full Rust
- `libmagic` is full **C**

Vulnerability researchers in the room, be honest!

That didn't help me into choosing existing Rust magic crate

Safety

This crate is a binding to the `libmagic` C library and as such subject to its security problems. Please note that `libmagic` has several CVEs, listed on e.g. Repology. Make sure that you are using an up-to-date version of `libmagic` and ideally add additional security layers such as sandboxing (which this crate does *not* provide) and **do not use it on untrusted input** e.g. from users on the internet!

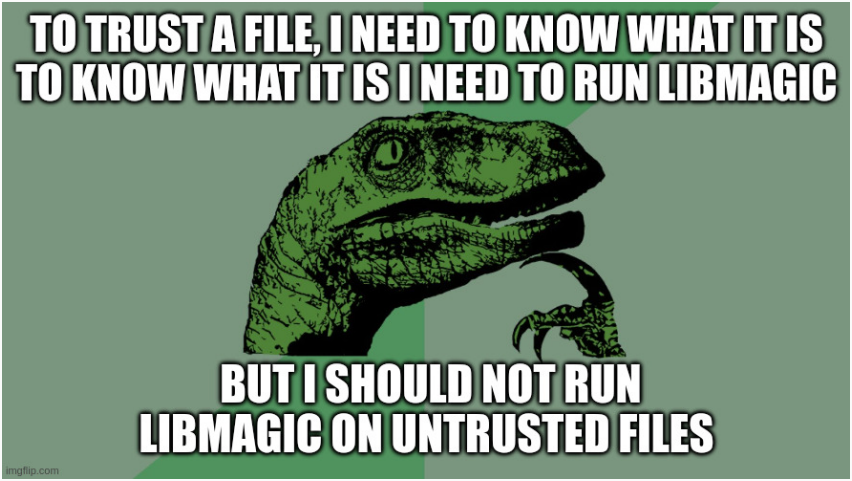
The Rust code of this crate needs to use some `unsafe` for interacting with the `libmagic` C FFI.

This crate has not been audited nor is it ready for production use.

This Rust project / crate is not affiliated with the original `file` / `libmagic` C project.

'do not use it on untrusted input'

How to address this paradox?



Re-implement libmagic in Rust

Four months after: it's almost cooked

- Understand the libmagic's rule format
- Support almost all the `tests` supported by libmagic (only missing `der` test)
- Implemented all the matching logic (how to apply magic tests on the content to identify)

Bonus

- Cross-platform thanks to Rust toolchains
- Compiled rules can be embedded in the library

Demo/failure Time

How to test your stuff?

```
# Use dev branch, no release yet
git clone --branch=dev https://github.com/qjerome/magic-rs.git
cd magic-rs
cargo build --release
./target/release/file scan -j /the/directory/you/wanna/scan
```

This project should still be considered as WIP

https://github.com/qjerome/magic-rs