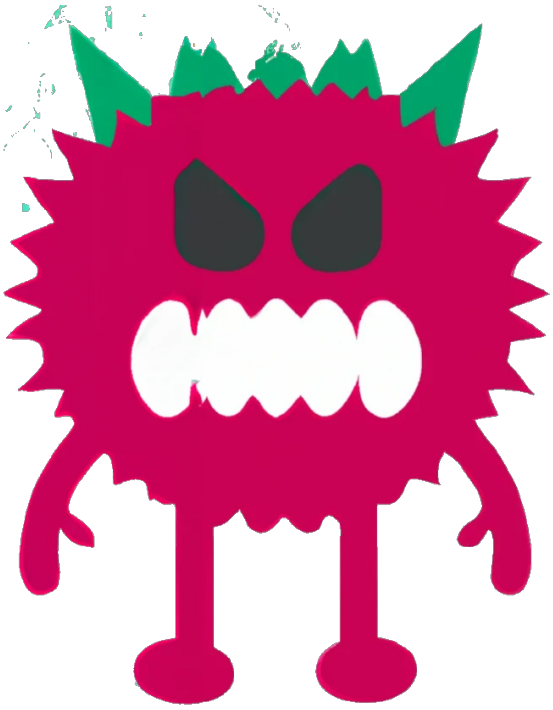


Raspirus docs

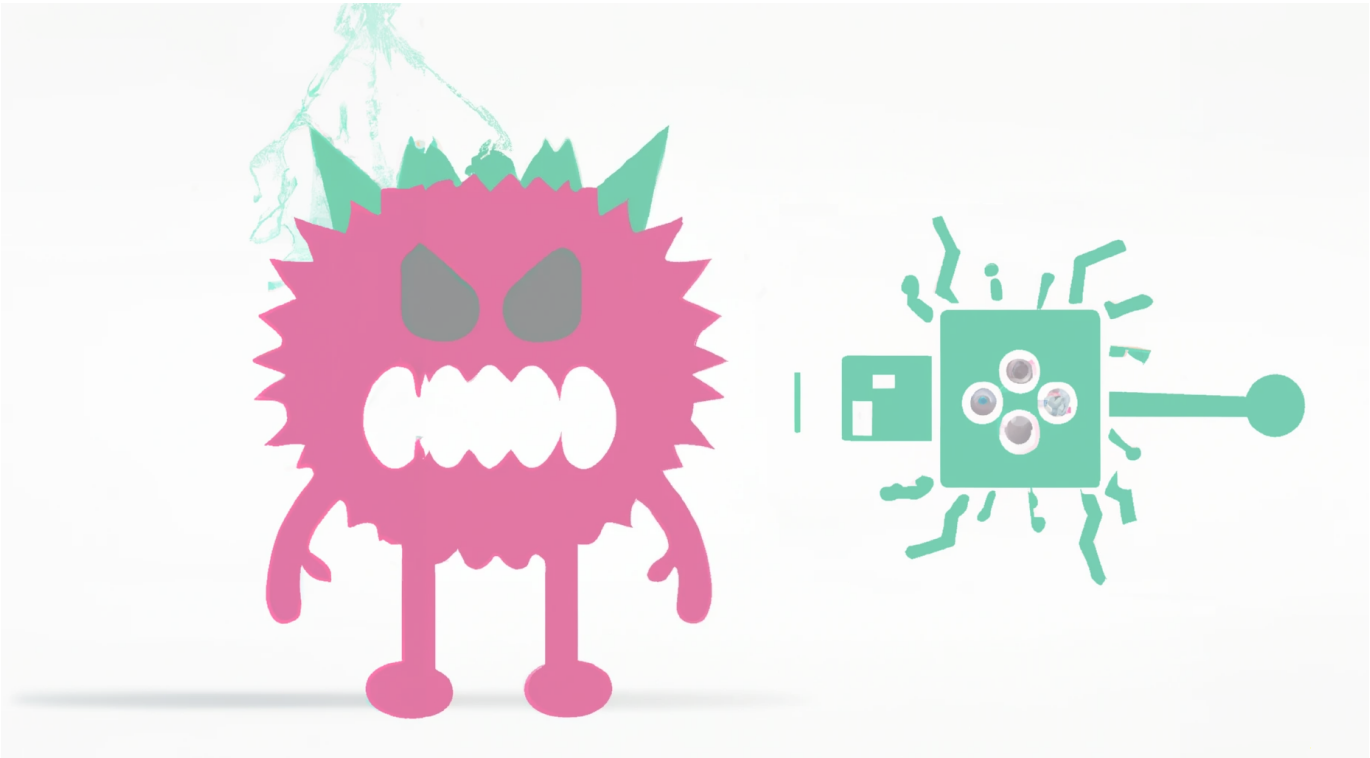
A simple hash-based virus scanner

Benjamin Demetz

Table of contents

1. HOME 3
1.1 Einführung 3
1.2 Getting Started 3
1.3 Questions? 4
2. CONTRIBUTING 5
2.1 Enhance the Codebase 5
2.2 Enrich the Documentation 5
2.3 Drive Translations 5
2.4 Infuse Artwork and Media 5
2.5 Provide Feedback 5
3. DEVELOPERS 6
3.1 Navigating the Architecture 6
3.2 Starting Your Development Journey 6
3.3 Exploring the Backend 7
3.4 Unpacking the Frontend 7
3.5 Evaluating Test Coverage 7
4. FAQ 8
4.1 Kommentare 8
5. Anleitungen 9
5.1 Export to PDF 9
5.2 Übersetzungen 9
5.3 Kommentare 9
6. SCREENSHOTS 10
6.1 Zuhause 10
6.2 Scanning 11
6.3 Result 12
6.4 Raspberry Pi Setup 12
7. STARS 14
7.1 Sponsors 14
7.2 Contributors 14
7.3 Special Credits 14

1. HOME



1.1 Einführung

Raspirus: Empowering Your Malware Protection

Welcome to the official documentation for Raspirus, your lightweight signature-based malware scanner. Originally designed to scan attached USB drives using a Raspberry Pi, Raspirus has evolved into a versatile tool capable of scanning local files and folders as well. Some of its standout features include:

- **Cost-Free Protection:** Raspirus operates solely on donations, ensuring top-notch protection without any financial burden.
- **Tailored Signature Detection:** Our custom signature-based approach guarantees accurate identification of malware.
- **Comprehensive File Scans:** Raspirus can efficiently scan compressed files, ensuring no threat goes undetected.
- **Privacy Prioritized:** Offering a privacy-friendly option, Raspirus keeps your personal information secure.
- **Cross-Platform Convenience:** Enjoy the benefits of Raspirus protection on a variety of operating systems.
- **Swift and Dependable:** Count on Raspirus for fast and reliable malware detection.
- **Sleek Modern Interface:** With user-friendliness at its core, Raspirus boasts a beautiful and intuitive UI.

1.2 Getting Started

1.2.1 For Regular Users

Getting started with Raspirus is a breeze. Follow these simple steps:

1. Visit our website or head to the GitHub release page.
2. Download the executable that matches your operating system.

> **Note**
>
> If you plan to use Raspirus on the Raspberry Pi as a standalone application, we have a dedicated guide for it.

1.2.2 For Developers

Are you a developer looking to set up Raspirus? We've got you covered. Check out our comprehensive guides for various operating systems in the Developers section.

1.3 Questions?

Got questions about Raspirus? We're here to help!

- Visit our FAQ section for answers to common queries.
- Join our thriving community on the Discord server to engage with fellow users.
- If you've encountered a bug, browse the GitHub issues to see if it's already reported.

Thank you for choosing Raspirus for your malware protection needs. Together, we're making the digital world safer for everyone.

22. Dezember 2023

3. April 2023

2. CONTRIBUTING

2.1 Enhance the Codebase

As a developer looking to amplify Raspirus's functionality, your contributions are invaluable. Please adhere to the following guidelines:

- **Check Existing Issues:** Before diving in, peruse existing issues to avoid creating duplicates. If none exist, feel free to open a new one.
- **Documentation is Key:** Always ensure your code is well-documented. Remember to adapt tests to maintain code integrity.
- Read the Code of Conduct

2.2 Enrich the Documentation

Raspirus's documentation is crafted using Markdown and powered by MkDocs and Python. To contribute to the documentation:

- **Follow the Guide:** To begin, set up the documentation on your local machine by following the guide.
- **Material Theme:** Our documentation utilizes the Material Theme, which extends Markdown functionality. Discover more about it here.

2.3 Drive Translations

Our documentation is multilingual thanks to Crowdin, an intuitive external service. GUI translations are handled via JSON files. Refer to the guide for comprehensive insights.

2.4 Infuse Artwork and Media

The repository for artwork (logos, banners, etc.) and media (powerpoints, articles, graphs, etc.) is hosted separately at this repository. Feel free to contribute and bolster our visual presence. Note that the current logo and banner were AI-generated due to the lack of artistic expertise.

2.5 Provide Feedback

Beyond the above contributions, there are other impactful ways to get involved:

- **Join Discord:** Engage with our community on the Discord server to share valuable feedback and ideas.
- **Consider Donating:** If you find value in Raspirus, consider donating to support ongoing development.
- **Spread the Word:** A simple yet effective way to contribute is by downloading Raspirus and introducing it to friends.

Thank you for your dedication to Raspirus's growth and improvement. Your involvement is a cornerstone of our success.

22. Dezember 2023

22. August 2023

3. DEVELOPERS

3.1 Navigating the Architecture

```
graph LR
     A[Start] --> B{Scan location specified?};
     B --> |Yes| C[Start scan];
     C --> |Start Loop| D[File found];
     D --> E[Create Hash];
     E --> F[Compare Hash];
     F --> G{Hash found in DB?};
     G --> |Yes| H[Flag as Malware];
     G --> |No| I[Flag as Safe];
     H & I --> J[Continue iteration];
     J --> K{Last file?};
     K --> |Yes| L[Stop scanner];
     L --> M[Display Results];
     K --> |No| N[Start again];
     N --> D;
     B --> |No| O[Stop]
```

Raspirus is structured into two integral components: frontend and backend. These components, built using distinct languages and frameworks, are interconnected via a third-party framework called Tauri. This framework not only facilitates communication between the frontend and backend but also enables us to incorporate Rust functions into the frontend. Furthermore, Tauri empowers the distribution of Raspirus across various operating systems.

3.2 Starting Your Development Journey

Windows **Linux** **macOS**

1. Clone the repository
2. Install Tauri and Prerequisites
3. Install npm
4. Install Next.js with `npm install next@latest react@latest react-dom@latest`
5. Install npm dependencies with: `npm i`
6. Start development with `cargo tauri dev`
7. or build Raspirus with `cargo tauri build`

1. Clone the Repository
2. Execute `make install`
3. Run the application with `raspirus`

1. Clone the repository
2. Install Tauri and Prerequisites
3. Install npm
4. Install Next.js with `npm install next@latest react@latest react-dom@latest`
5. Install npm dependencies with: `npm i`
6. Start development with `cargo tauri dev`
7. or build Raspirus with `cargo tauri build`

Should you encounter any hiccups during your initial run or build, ensure that you've followed each step diligently. Confirm the accurate creation of both logs and config files.

3.3 Exploring the Backend

```
classDiagram
     Main <|-- DBOps
     Main <|-- Configs
     Main <|-- FileLogs
     Main <|-- Scanner
     Main: +Config config_file

     class DBOps {
          +Connection db_conn
          +String db_file
          +TauriWindow t_win
          +new()
          +update_db()
          +hash_exists()
     }

     class Configs {
          +Data data
          +new()
          +save()
          +load()
     }

     class FileLogs {
          +File file
          +log()
          +create_file()
     }

     class Scanner {
          +String scan_loc
          +DbOps db_ops
          +Vec malware_list
          +search_files()
          +create_hash()
          +get_folder_size()
     }
```

The backend, an essential cog in the Raspirus machinery, is meticulously crafted in Rust for superior performance. The primary file houses functions accessible from the frontend, which must yield JSON-compatible outcomes. For a detailed breakdown, reference the graph above outlining the backend's modular arrangement.

3.4 Unpacking the Frontend

Our frontend, developed with JavaScript via the Next.js framework, emphasizes user-friendliness and functionality. Comprising components and pages, it mirrors the simplicity and robustness of Next.js. Refer to the illustrated graph above for an approximate visual representation of the frontend's architecture.

3.5 Evaluating Test Coverage

- Backend tests, authored in Rust, can be executed via the `cargo test` command. Access these tests in the tests directory. Check test coverage on Codecov.
- Frontend tests, created with Selenium, are currently in development.

Thank you for your interest in contributing to Raspirus's development. Your expertise fuels our progress.

22. Dezember 2023

22. August 2023

4. FAQ

???+ question "App crashes when updating" On Windows, it has been observed that the app crashes when attempting to update the database. We are aware of this issue and actively working to resolve it. The problem arises because the update function requires administrative privileges, which Windows does not automatically provide. To temporarily resolve this issue, you can execute the app with administrative privileges. Right-click on the app and select "Run as administrator" to launch it with the necessary privileges.

???+ question "Where does the Raspirus logo come from?" The logo of the Raspirus app features a red monster named Stuart, who is designed to represent a virus-eating creature. The logo was generated using DALL-E, along with creative image editing and merging. Stuart is a friendly monster, except when he's hungry for viruses. You can find additional media and documents in the dedicated repository. Feel free to use these images to create your own artwork and share them in the discussion boards.

My VScode setup is giving me issues

The Rust Analyzer plugin in Visual Studio Code searches for a `Cargo.toml` file in the current directory or its parent directory. To address this issue, you can add an option to the plugin settings and specify the location of your `Cargo.toml` file.

As mentioned in this comment, you can add the following lines to the end of your plugin settings JSON. Afterward, restart the Rust Analyzer for the modifications to take effect.

```
{
    "rust-analyzer.linkedProjects": [
        "/home/stuart/raspirus/raspirus/Cargo.toml"
    ]
}
```

???+ question "Can't select directories/files" Unfortunately, as of this issue with Tauri, we currently can't allow users to select both files and folders. To switch between selecting a single file or folder, you need to change it in the Raspirus settings. There you will find a switch for it.

???+ question "What is obfuscated mode?" Raspirus was originally intended for enterprise usage and therefore needed to be privacy-friendly. To ensure that, it added the "Obfuscation Mode", which will hide everything, detect malware faster and only display: "Malware found" or "No malware found". It is on by default, so if you want to know a bit more about your scan, you should probably deactivate it. You can do so in the settings.

???+ question "error: found a virtual manifest instead of a package manifest" If you get this error when performing `cargo install` or using the Makefile, please note that it's a know issue. The solution is simple, as explained on the this Stackoverflow answer, simply change the command to include the workspace, like this: `cargo install --path src-tauri/`

22. Dezember 2023

5. April 2023

4.1 Kommentare

5. Anleitungen

Here you will find helpful guides on how to export the documentation to PDF format and how to contribute translations to this project. If you have any additional requests for guides, please leave a comment below!

5.1 Export to PDF

If you would like to have an offline version of this documentation in PDF format, you can follow these step-by-step instructions:

1. Clone this repository by following the instructions on GitHub.
2. Install the necessary requirements for the PDF conversion tool. You can find the requirements specific to your operating system here.
3. Navigate to the cloned directory and install the required dependencies. Please note that you will need Python3 and pip for this step. You can install the dependencies by running the following command: `pip install -r requirements.txt`.
4. Install mkdocs and execute the build command: `mkdocs build`.
5. If everything goes smoothly, the resulting PDF file should be located in the `site/pdf` directory with the name `document.pdf`.

Please be aware that the exported PDF may have some issues with images and iFrames, but the text should be readable and suitable for sharing offline.

5.2 Übersetzungen

We welcome contributions to translate this documentation into other languages. If you are interested in translating the document, you can find the necessary information and resources at the following link: Translate this document.

Your contributions are highly appreciated, and they will help make this documentation more accessible to a wider audience.

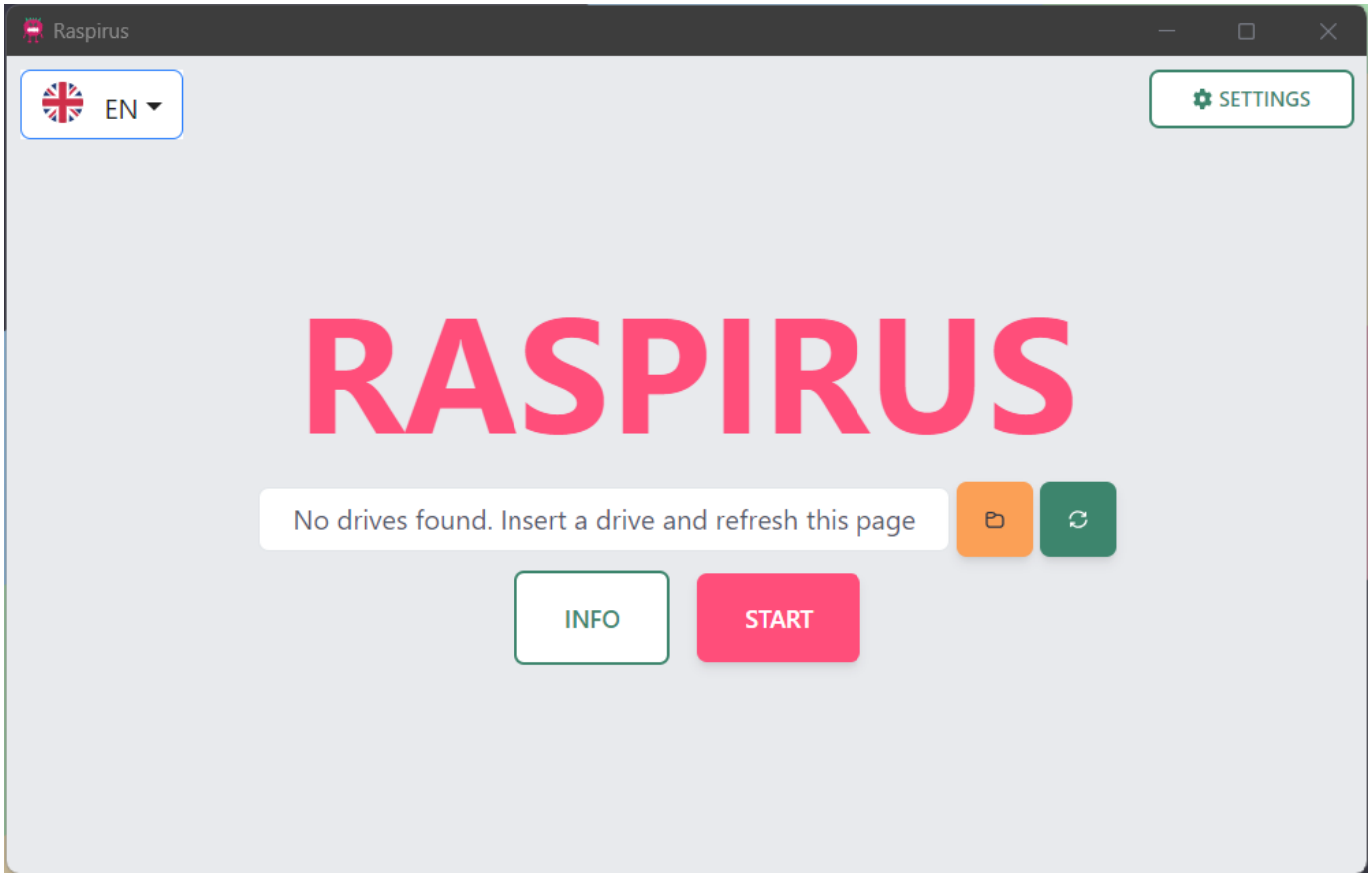
22. Dezember 2023

3. April 2023

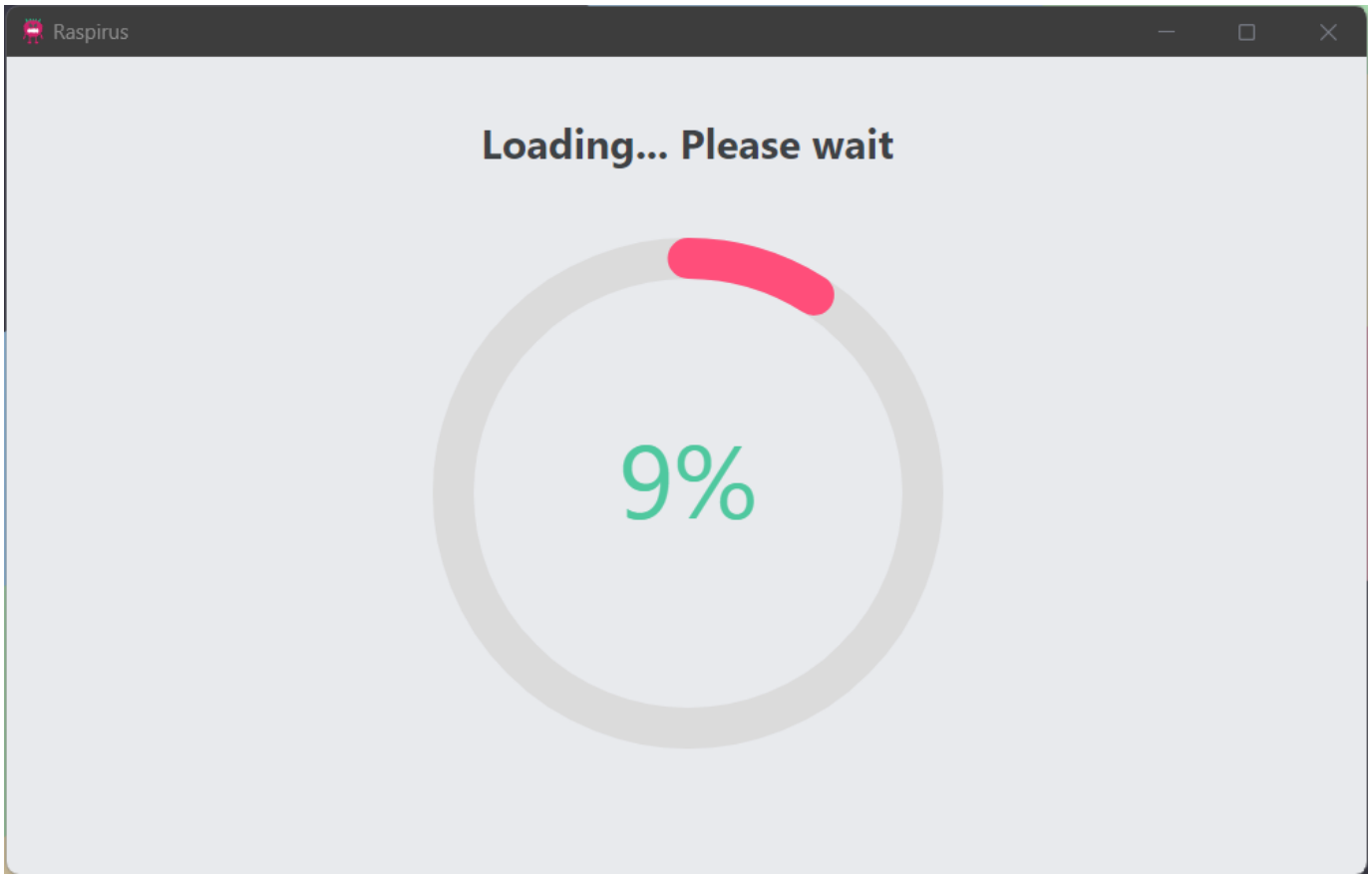
5.3 Kommentare

6. SCREENSHOTS

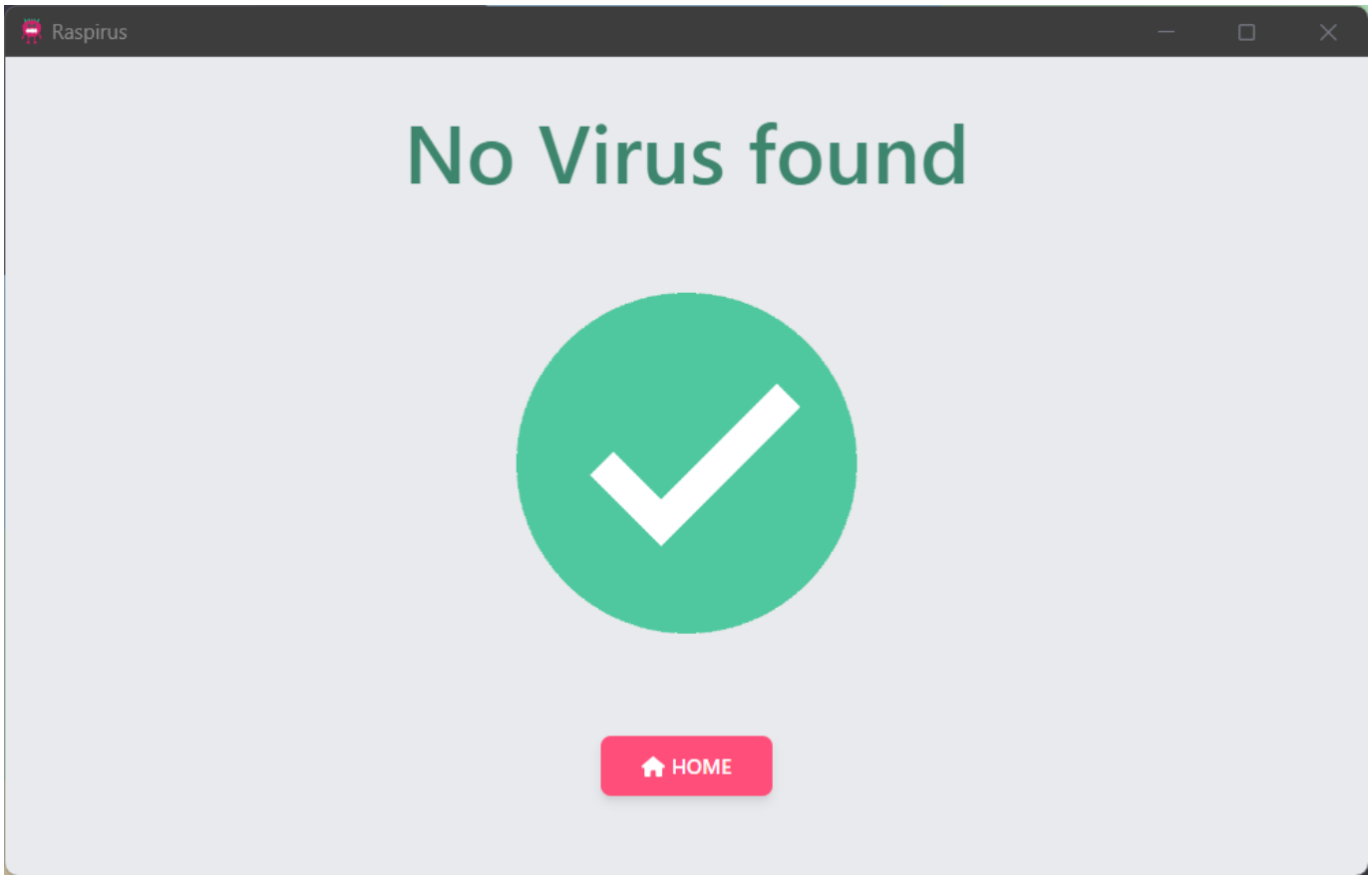
6.1 Zuhause



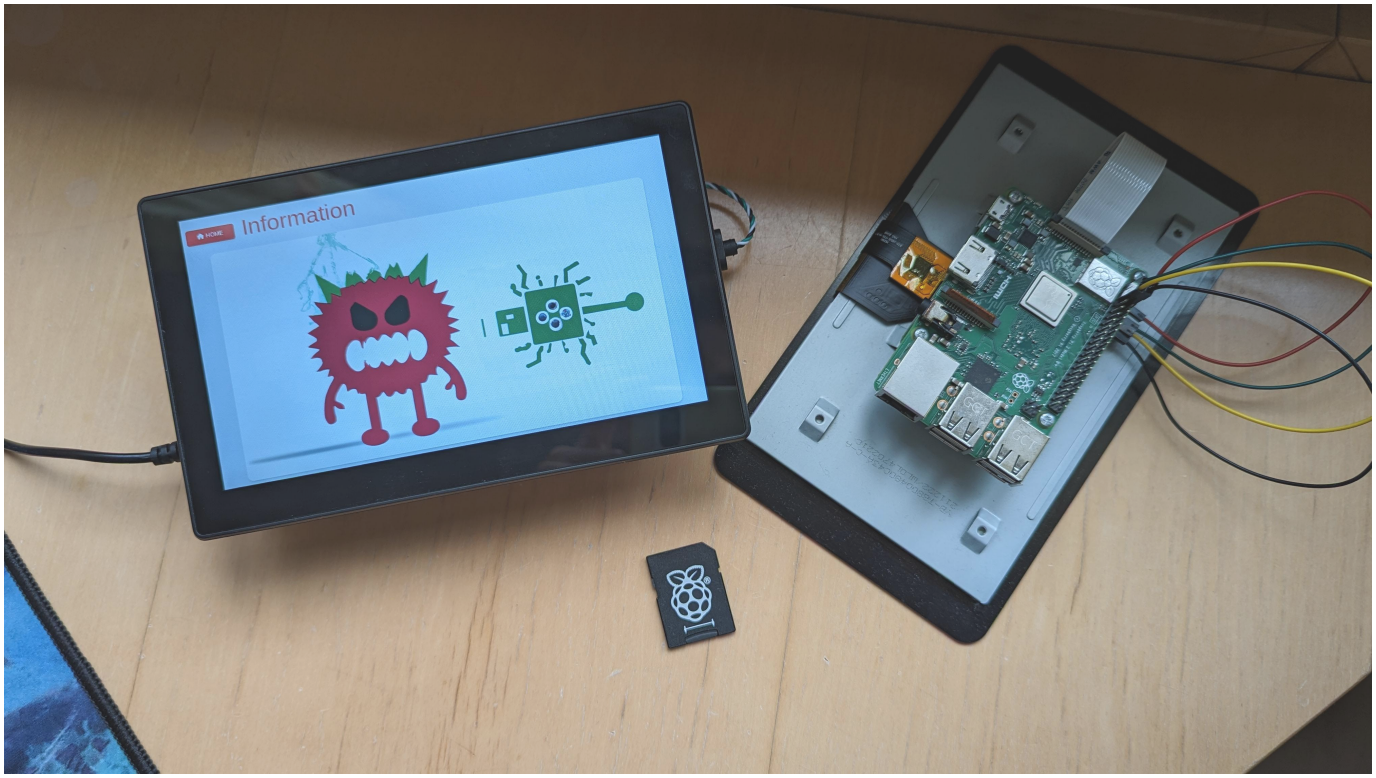
6.2 Scanning



6.3 Result



6.4 Raspberry Pi Setup



22. Dezember 2023

22. August 2023

7. STARS

7.1 Sponsors

7.1.1 Benjamin Demetz

❤️ Maintainer

7.1.2 Nurkanat Baysenkul

💰 Sponsor

7.2 Contributors

7.2.1 Matthias Dieter Wallnöfer

👩‍🏫 Mentoring

7.2.2 GamingGuy003

💻 Backend Developer

7.2.3 Zack Amoroso

📦 Linux Tester

7.2.4 Paul Guyot

💻 GitHub Action

7.3 Special Credits

7.3.1 Lairex59

💡 Ideas and Support

22. Dezember 2023

22. August 2023