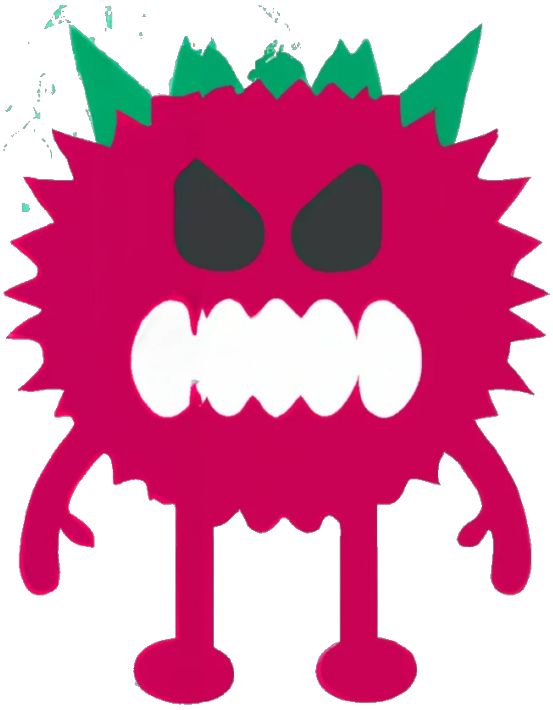


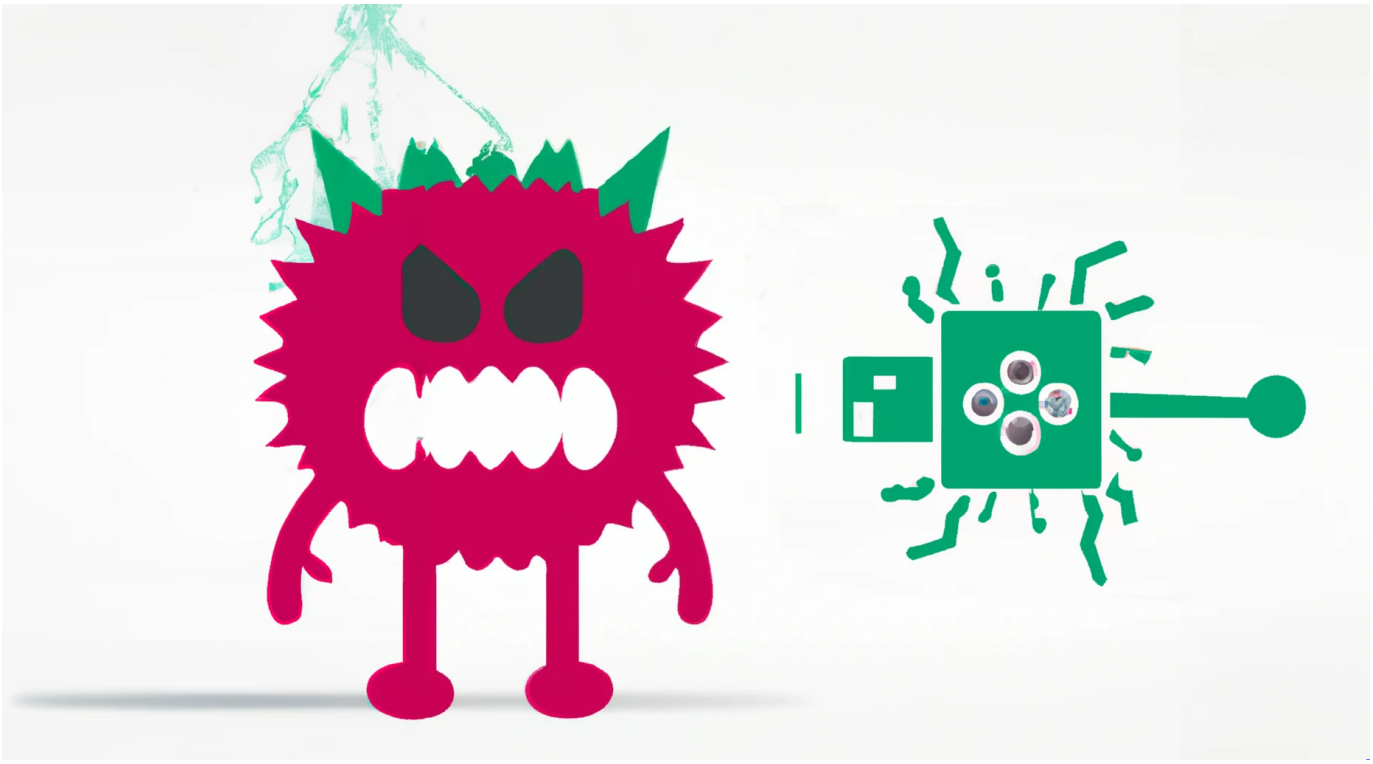
Raspirus docs

A simple hash-based virus scanner

Benjamin Demetz

Table of contents

1. HOME



1.1 Introduzione

Raspirus: Potenziare La Tua Protezione Malware

Benvenuti nella documentazione ufficiale per Raspirus, il vostro leggero scanner di malware basato sulla firma. Originariamente progettato per eseguire la scansione di unità USB collegate utilizzando un Raspberry Pi, Raspirus si è evoluto in un versatile strumento in grado di scansionare anche file e cartelle locali. Alcune delle sue caratteristiche distintive includono:

- **Protezione senza costi:** Raspirus opera esclusivamente su donazioni, garantendo una protezione di prim'ordine senza alcun onere finanziario.
- **Rilevamento firma su misura:** Il nostro approccio personalizzato basato sulla firma garantisce un'identificazione accurata del malware.
- **Comprehensive File Scans:** Raspirus può eseguire in modo efficiente la scansione dei file compressi, garantendo che nessuna minaccia non venga rilevata.
- **Privacy Prioritizzata:** Offrendo un'opzione per la privacy, Raspirus mantiene le tue informazioni personali al sicuro.
- **Convenienza cross-piattaforma:** Goditi i vantaggi della protezione Raspirus su una varietà di sistemi operativi.
- **Swift e Dependable:** Contare su Raspirus per un rilevamento rapido e affidabile di malware.
- **Elegante interfaccia moderna:** Raspirus vanta una bellissima e intuitiva interfaccia utente: grazie alla sua semplicità di utilizzo.

1.2 Per Iniziare

1.2.1 Per Utenti Regolari

Cominciare con Raspirus è una brezza. Seguire questi semplici passaggi:

1. Visita la nostra website o vai alla pagina di rilascio di GitHub.
2. Scarica l'eseguibile che corrisponde al tuo sistema operativo.

> **Nota**
>
> Se hai intenzione di utilizzare Raspirus sul Raspberry Pi come applicazione standalone, abbiamo una [guida dedicata](https://raspirus.github.io/docs/guides#Raspberry-Pi) per questo.

1.2.2 Per Gli Sviluppatori

Sei uno sviluppatore che cerca di impostare Raspirus? Ti abbiamo coperto. Scopri le nostre guide complete per vari sistemi operativi nella sezione Sviluppatori.

1.3 Domande?

Hai domande su Raspirus? Siamo qui per aiutare!

- Visita la nostra sezione FAQ per le risposte alle query comuni.
- Unisciti alla nostra fiorente comunità sul server Discord per entrare in contatto con altri utenti.
- Se hai incontrato un bug, sfogliare i problemi di GitHub per vedere se è già segnalato.

Grazie per aver scelto Raspirus per le vostre esigenze di protezione malware. Insieme, stiamo rendendo il mondo digitale più sicuro per tutti.

22 dicembre 2023

3 aprile 2023

2. CONTRIBUTO

2.1 Migliora il Codebase

Come sviluppatore che cerca di amplificare la funzionalità di Raspirus, i tuoi contributi sono preziosi. Si prega di aderire alle seguenti linee guida:

- **Controllare i problemi esistenti:** Prima di fare immersioni in tuffo, perusare i problemi esistenti per evitare di creare duplicati. Se nessuno esiste, sentitevi liberi di aprirne uno nuovo.
- **La documentazione è la chiave:** Assicurati sempre che il tuo codice sia ben documentato. Ricordati di adattare i test per mantenere l'integrità del codice.
- Leggi il codice di condotta

2.2 Arricchisci la documentazione

La documentazione di Raspirus è realizzata utilizzando Markdown e powered by MkDocs e Python. Contribuire alla documentazione:

- **Segui la guida:** Per iniziare, imposta la documentazione sulla tua macchina locale seguendo la guide.
- **Material Theme:** La nostra documentazione utilizza il Material Theme, che estende la funzionalità Markdown. Scopri di più su di esso here.

2.3 Traduzioni Drive

La nostra documentazione è multilingue grazie ad Crowdin, un servizio esterno intuitivo. Le traduzioni GUI sono gestite tramite i file JSON. Fare riferimento alla guide per informazioni complete.

2.4 Infondere grafica e media

Il repository per le opere d'arte (loghi, banner, ecc.) e supporti (powerpoint, articoli, grafici, ecc.) è ospitato separatamente su questo repository. Sentitevi liberi di contribuire e rafforzare la nostra presenza visiva. Si noti che l'attuale logo e banner sono stati generati da IA a causa della mancanza di competenze artistiche.

2.5 Fornisci Feedback

Al di là dei suddetti contributi, esistono altri modi efficaci per coinvolgersi:

- **Unisciti a Discord:** Coinvolgi la nostra comunità sul server Discord per condividere preziosi feedback e idee.
- **Considera la donazione:** Se trovi valore in Raspirus, considera donating per supportare lo sviluppo in corso.
- **Diffondi la Parola:** Un modo semplice ma efficace per contribuire è scaricare Raspirus e presentarlo agli amici.

Grazie per la vostra dedizione alla crescita e al miglioramento di Raspirus. Il vostro coinvolgimento è una pietra angolare del nostro successo.

22 dicembre 2023

22 agosto 2023

3. SVILUPPI

3.1 Navigare nell'Architettura

```
grafico LR
     A[Start] --> B{Posizione di scansione specificata? ;
     B --> Si<unk> C[Start scan];
     C --> <unk> Start Loop<unk> D[File trovato];
     D --> E[Crea Hash];
     E --> F[Confronta Hash];
     F --> G{Hash trovato in DB? ;
     G --> <unk> Si<unk> H[Contrassegna come malware];
     G --> <unk> No<unk> I[Contrassegna come sicuro];
     H & I --> J[Continue iteration];
     J --> K{Ultimo file? ;
     K --> <unk> Si<unk> L[Stop scanner];
     L --> M[Display Results];
     K --> <unk> No<unk> N[Start again];
     N --> D;
     B --> <unk> No<unk> O[Stop]
```

Raspirus è strutturato in due componenti integrali: frontend e backend. Questi componenti, costruiti utilizzando linguaggi e quadri distinti, sono interconnessi tramite un framework di terze parti chiamato Tauri. Questo framework non solo facilita la comunicazione tra il frontend e il backend, ma ci permette anche di integrare le funzioni Rust nel frontend. Inoltre, Tauri consente la distribuzione di Raspirus su vari sistemi operativi.

3.2 Iniziare Il Tuo Viaggio Di Sviluppo

\=== "Windows" Clona il repository 2. Installa Tauri e Prerequisiti 3. Installa npm 4. 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` Installa dipendenze npm con: `npm i` 6. Inizia lo sviluppo con `cargo tauri dev` 7. o costruisci Raspirus con `cargo tauri build`

\=== "Linux" Clona il Repository 2. Esegui `make install` 3. 1. Clone the Repository 2. Execute `make install` 3. Run the application with `raspirus`

\=== "macOS" Clona il repository 2. Installa Tauri e Prerequisiti 3. Installa npm 4. 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` Installa dipendenze npm con: `npm i` 6. Inizia lo sviluppo con `cargo tauri dev` 7. o costruisci Raspirus con `cargo tauri build`

Se dovessi incontrare dei singhiozzi durante la tua esecuzione iniziale o costruire, assicurati di aver seguito diligentemente ogni passaggio. Confermare la creazione accurata di entrambi i log e file di configurazione.

3.3 Esplorare il 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()
}
```

Il backend, un cog essenziale nella macchina Raspirus, è meticolosamente realizzato in Rust per prestazioni superiori. Il file primario ospita funzioni accessibili dal frontend, che devono produrre risultati compatibili con JSon. Per una ripartizione dettagliata, fare riferimento al grafico sopra che delinea la disposizione modulare del backend.

3.4 Scompattare Frontend

Il nostro frontend, sviluppato con JavaScript tramite il framework Next.js, sottolinea la facilità d'uso e la funzionalità. Composto da componenti e pagine, rispecchia la semplicità e la robustezza di Next.js. Fare riferimento al grafico qui sopra illustrato per una rappresentazione visiva approssimativa dell'architettura del frontend.

3.5 Valutazione Della Copertura Dei Test

- I test di backend, scritti in Rust, possono essere eseguiti tramite il comando `cargo test`. Accedi a questi test nella directory test. Controlla la copertura del test su Codecov.
- I test Frontend, creati con Selenium, sono attualmente in fase di sviluppo.

Grazie per il vostro interesse a contribuire allo sviluppo di Raspirus. La vostra esperienza alimenta i nostri progressi.

22 dicembre 2023

22 agosto 2023

4. FAQ

???+ question "App crashes when updating" On Windows, it has been observed that the app crashes when attempting to update the database. Siamo consapevoli di questo problema e ci stiamo adoperando attivamente per risolverlo. Il problema sorge perché la funzione di aggiornamento richiede privilegi amministrativi, che Windows non fornisce automaticamente. Per risolvere temporaneamente questo problema, è possibile eseguire l'applicazione con privilegi amministrativi. Fare clic con il tasto destro sull'app e selezionare "Esegui come amministratore" per avviarla con i privilegi necessari.

dove viene il logo Raspirus?

Il logo dell'app Raspirus presenta un mostro rosso di nome Stuart, che è progettato per rappresentare una creatura che mangia virus. Il logo è stato generato utilizzando DALL-E, insieme all'editing e alla fusione di immagini creative. Stuart è un mostro amichevole, tranne quando ha fame di virus. Puoi trovare ulteriori supporti e documenti nel repository dedicato. Sentitevi liberi di utilizzare queste immagini per creare la vostra grafica e condividerle nelle schede di discussione.

nio VScode setup mi sta dando problemi

```
Il plugin Rust Analyzer in Visual Studio Code cerca un file `Cargo.toml` nella directory corrente o nella directory principale. Per risolvere questo problema, puoi aggiungere un'opzione alle impostazioni del plugin e specificare la posizione del tuo `Cargo. oml` file.

Come menzionato nel [questo commento](https://github. om/rust-lang/rust-analyzer/issues/2649#issuecomment-691582605), è possibile aggiungere le seguenti righe alla fine delle impostazioni del plugin JSON. Successivamente, riavviare l'Analizzatore Rust affinché le modifiche abbiano effetto.

```json
{
    "rust-analyzer. inkedProjects": [
        "/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. Per passare dalla selezione di un singolo file o cartella, è necessario modificarlo nelle impostazioni di Raspirus. Qui troverete un interruttore per esso.

s'è la modalità offuscata?

Raspirus era originariamente destinato all'uso aziendale e quindi doveva essere rispettoso della privacy. Per garantire che, ha aggiunto la "modalità obfuscation", che nasconderà tutto, rilevare il malware più velocemente e solo visualizzare: "Malware trovato" o "Nessun malware trovato". È attivata per impostazione predefinita, quindi se vuoi saperne di più sulla tua scansione, probabilmente dovresti disattivarla. Puoi farlo nelle impostazioni.

???+ 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. La soluzione è semplice, come spiegato sulla this Risposta allo stackoverflow, basta cambiare il comando per includere lo spazio di lavoro, come questo: `cargo install --path src-tauri/`

22 dicembre 2023

5 aprile 2023

4.1 Commenti

5. Guide

5.1 Impostazione Della Documentazione Locale

Impostare facilmente la documentazione sulla vostra macchina con questi semplici passi:

1. Clone this repository by following the instructions on GitHub.

```
git clone https://github.com/Raspirus/docs.git
```

1. Navigate to the cloned directory and install the required dependencies.

```
You can install the dependencies by running the following command: `pip install -r requirements.txt`.
```

1. Avvia il progetto localmente: Inizia il tuo progetto locale eseguendo questo comando:

```
servizio mkdocs
```

Questo avvierà un server di sviluppo, che ti permetterà di accedere alla documentazione tramite http://localhost:8000 nel tuo browser web preferito.

5.2 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. Segui il processo di configurazione descritto sopra.
2. Install mkdocs and execute the build command: `mkdocs build`.
3. 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. Tuttavia, il testo rimane leggibile e adatto per la condivisione offline.

5.3 Traduzioni

Per le traduzioni di questa documentazione, visita Crowdin, una piattaforma di collaborazione. Le traduzioni per le stringhe dell'interfaccia utente sono gestite attraverso i file JSON. Per contribuire, crea le tue traduzioni all'interno della cartella locales. Mantenere l'unicità delle chiavi quando si traduce dal file `en.json`.

5.4 Scanner Insights

5.4.1 Scansione Dei File Compressi

Nota

```
Questa opzione non è ancora disponibile per il Raspberry Pi. Scopri perché nella [sezione FAQ](https://raspirus.github.io/docs/faq)
```

Mentre Raspirus predefinito per la scansione delle cartelle, è possibile modificare questo comportamento tramite la pagina delle impostazioni. Attivando un interruttore, è possibile eseguire la scansione di singoli file, inclusi quelli compressi. È importante notare che un solo file può essere scansionato alla volta.

5.4.2 Navigazione di registri e configurazioni

In caso di eventi inaspettati o di crash improvvisi delle app, l'esame di registri e configurazioni può fornire approfondimenti. Individuare questi file nelle seguenti cartelle, in base al sistema operativo. Il ProjectDirs crate li memorizza nella posizione seguente:

File di configurazione

Piattaforma	Valore	Esempio
Linux	`$XDG_DATA_HOME` / `_project_path_` o `$HOME` /.local/share/ `_project_path_`	/home/alice/.local/share/barapp
macOS	`$HOME` /Library/Application Support/ `_project_path_`	/Users/Alice/Library/Application Support/ com.Foo-Corp.Bar-App
Finestre	`{FOLDERID_RoamingAppData}` \ `_project_path_` \data	C:\Users\Alice\AppData\Roaming\Foo Corp\Bar App\data

File di registro:

Piattaforma	Valore	Esempio
Linux	`$XDG_DATA_HOME` / `_project_path_` o `$HOME` /.local/share/ `_project_path_`	/home/alice/.local/share/barapp
macOS	`$HOME` /Library/Application Support/ `_project_path_`	/Users/Alice/Library/Application Support/ com.Foo-Corp.Bar-App
Finestre	`{FOLDERID_LocalAppData}` \ `_project_path_` \data	C:\Users\Alice\AppData\Local\Foo Corp\Bar App\data

Assicurarsi di includere questi file durante la segnalazione di bug, in quanto aiutano notevolmente nella risoluzione dei problemi.

5.5 GitHub Integration

5.5.1 Arricchire il database delle firme

COMANDO SUONO

5.6 Distribuzione Raspberry Pi

Originariamente su misura per distribuzione Raspberry Pi standalone con funzionalità touchscreen (simile alla modalità chiosco), lo scopo primario di questo progetto era la scansione delle unità USB collegate. Mentre il campo di applicazione del progetto si è ampliato, questa caratteristica rimane intatta. Segui la guida su Tauri e, se riscontri qualche problema, assicurati di farci sapere.

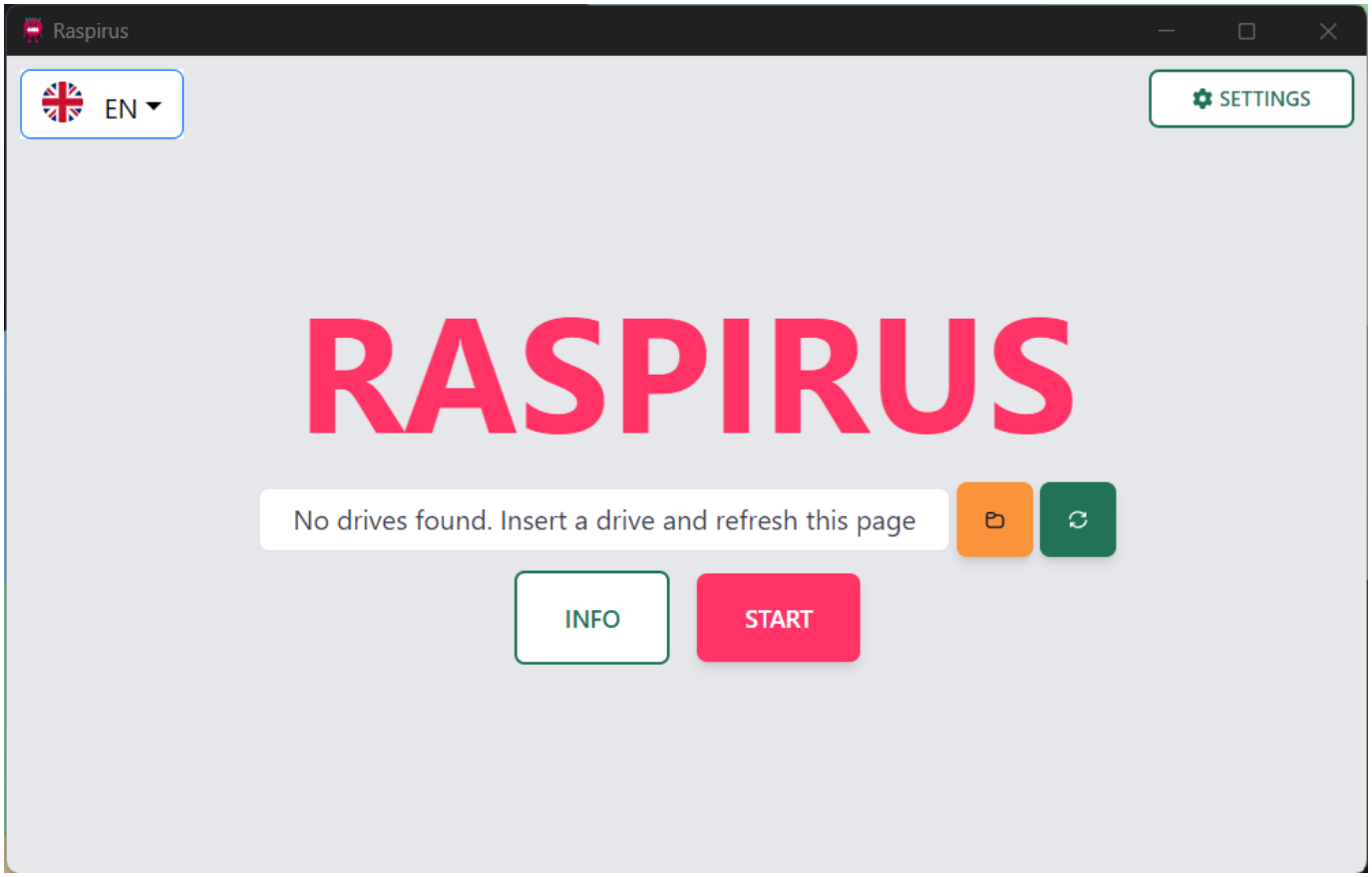
Grazie per aver scelto Raspirus come soluzione di protezione da malware. Queste guide complete garantiranno la vostra esperienza senza soluzione di continuità ed efficiente.

22 dicembre 2023

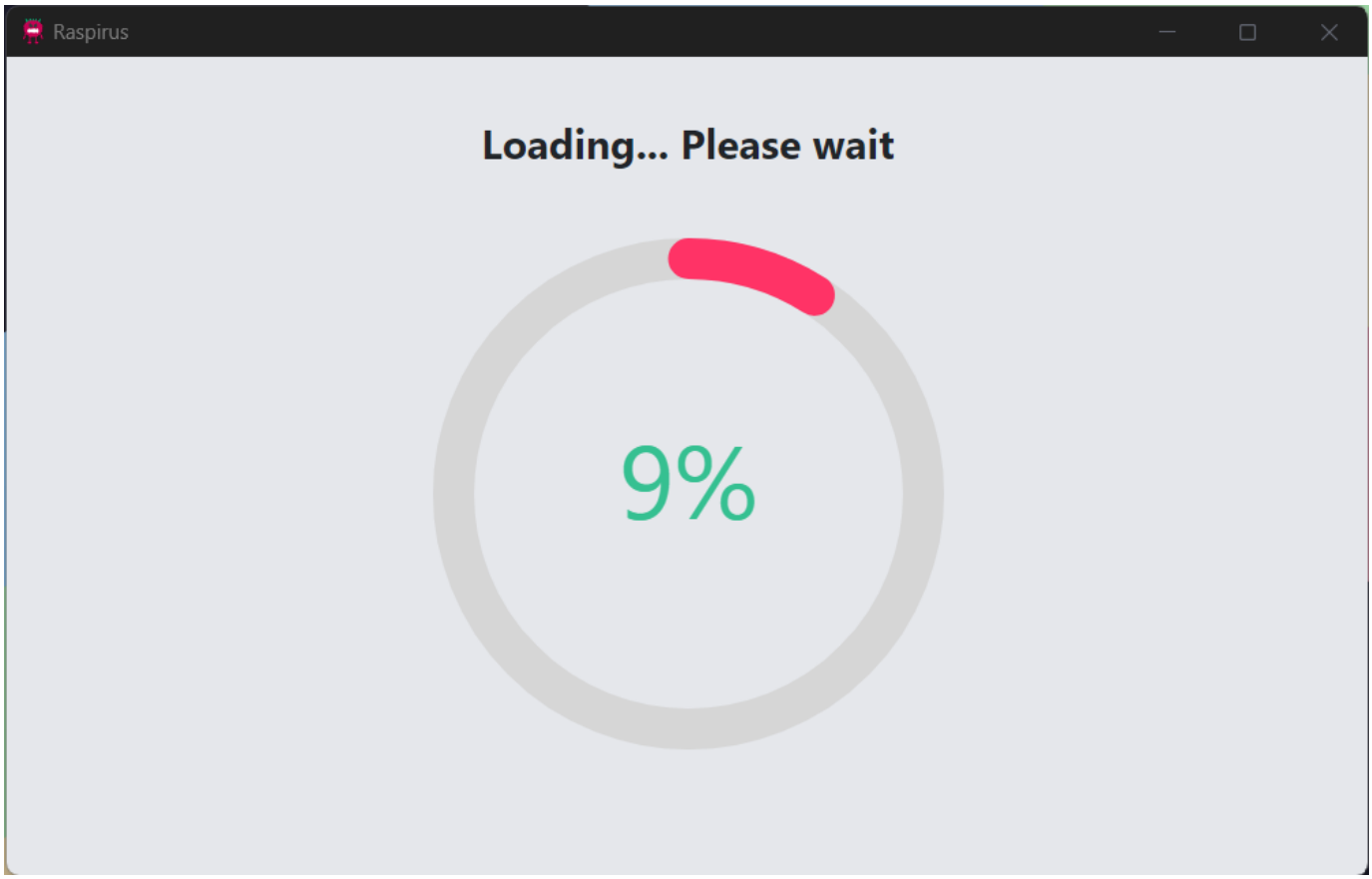
3 aprile 2023

6. SCREENSHOTS

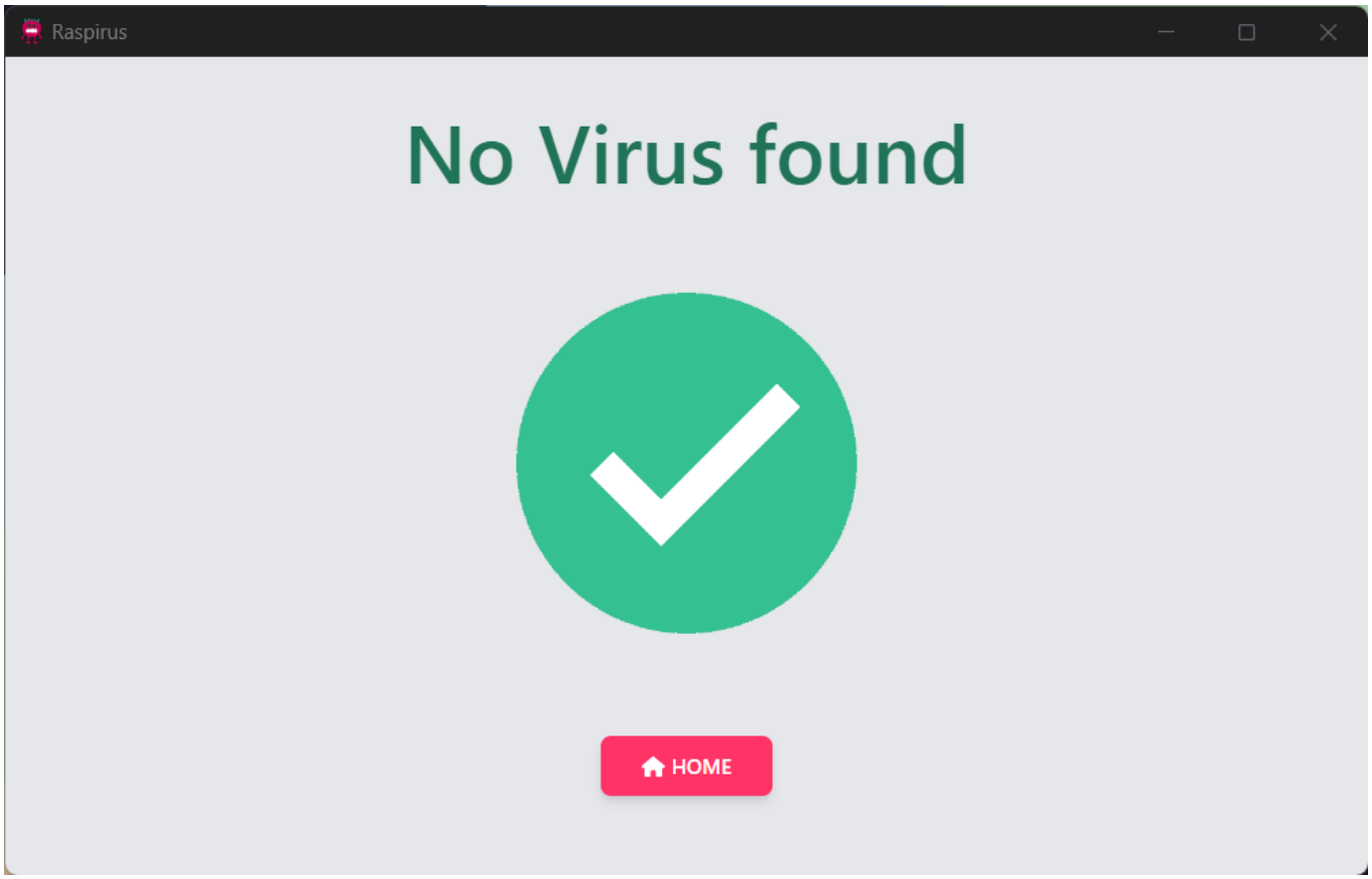
6.1 Home



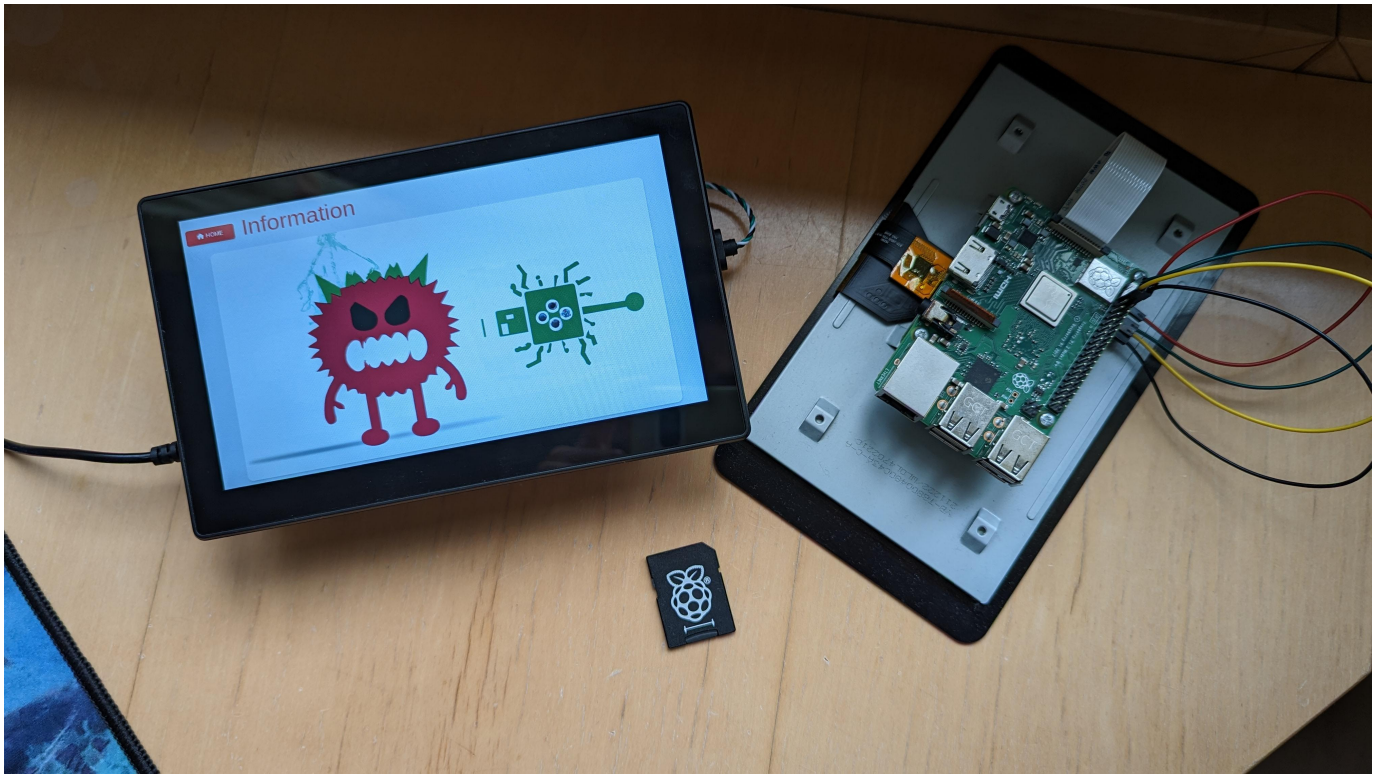
6.2 Scansione



6.3 Risultato



6.4 Configurazione Raspberry Pi



22 dicembre 2023

22 agosto 2023

7. STELLE

7.1 Sponsor

7.1.1 Benjamin Demetz

❤️ Manutentore

7.1.2 Nurkanat Baysenkul

💰 Sponsor

7.2 Collaboratori

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 Crediti Speciali

7.3.1 Lairex59

💡 Idee e supporto

22 dicembre 2023

22 agosto 2023