Team 4

Chrome Extension Threat & Risk Analyzer (CETRA)

Group Members

Kevin Deshayes — kede23@student.bth.se
Fajer Alhamwi — faaa22@student.bth.se
Majid Mohamed Hamid — mamd15@student.bth.se
Hugo Jonsson — hujo23@student.bth.se
Kasper Thunell — katu22@student.bth.se
Theo Svensson — thsn23@student.bth.se

DV1512
January 20, 2026

Contents

Contents

1 CETRA Documentation

This folder contains the documentation required for the DV1512 submission.

1.1 Contents

- Architecture Description
- API Documentation
- Requirements and Scope
- Dependencies
- Prerequisites
- Deployment Guide (Build & Run)
- User Guide
- Threat Modeling
- Vulnerability Scanning

1.2 Diagrams and Models

- Diagrams: `docs/diagrams/`
- UML/PlantUML: `docs/uml/`

1.3 Architecture Diagram

The following diagram shows the high-level component architecture of CETRA, including the frontend container, backend services, database, and external analysis systems.

The architecture diagram was created using PlantUML and follows a C4-style component diagram approach.

2 API Documentation

2.1 Overview

CETRA does not expose a public API. All application functionality is accessed through a Django-based Web UI. Internal endpoints coordinate extension analysis, report generation, and data retrieval.

2.2 Authentication

Authentication is handled using Django's built-in authentication mechanism. Access to analysis functionality and stored reports is restricted to authenticated

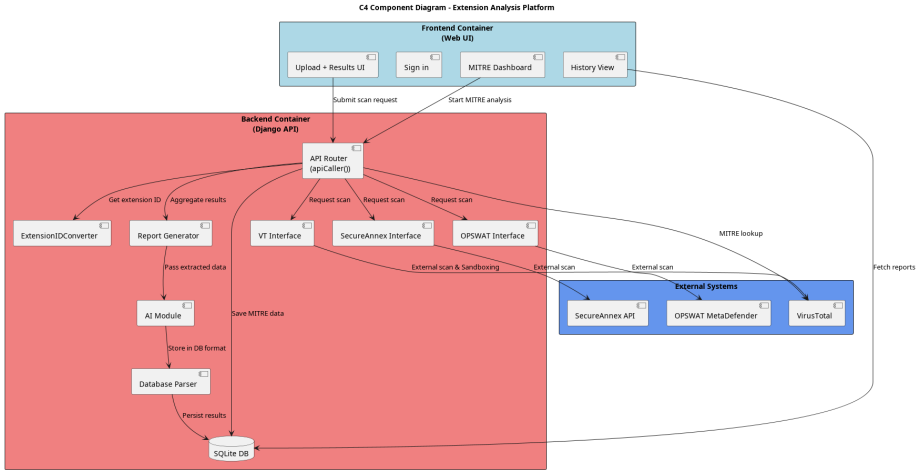

Figure 1: CETRA Component Architecture

users.

2.3 Internal Application Endpoints (High-Level)

- **Extension Submission**
  - Accepts Chrome Web Store IDs, ZIP files, or CRX files
  - Initiates the analysis workflow through the backend API router
- **Report Retrieval**
  - Displays stored analysis results in the Web UI
  - Supports exporting reports in JSON format
- **MITRE ATT&CK Analysis**
  - Initiates behavior mapping based on sandbox analysis results
  - Displays tactics and techniques through the Web UI

2.4 External APIs

The system integrates with the following external services:

- **VirusTotal API** – malware detection and reputation analysis
- **OPSWAT MetaDefender API** – multi-engine malware scanning
- **SecureAnnex API** – sandbox-based behavioral analysis
- **Google Gemini API** – AI-assisted interpretation of findings

API credentials are provided via environment variables.

3 Requirements and Scope

3.1 Functional Requirements

The CETRA system fulfills the following functional requirements:

- The system accepts uploaded Chrome extensions for analysis.
- The system extracts an extension ID from uploaded data.
- The system submits scan requests to multiple external analysis services.
- The system aggregates results from multiple scanners.
- The system generates an analysis report.
- The system retrieves stored reports for later viewing.
- The system persists analysis results in a database.
- The system requires authentication to access the application, including generating and viewing analysis reports.

3.2 Non-Functional Requirements

The system is designed to satisfy the following non-functional requirements:

- The system uses a modular, component-based architecture.
- The system supports integration with multiple external APIs.
- The system isolates untrusted file analysis from the core system.
- The system enforces authorization at the application level so that only authenticated users can access stored analysis reports.
- The system remains usable while long-running analyses are performed.
- The system validates uploaded files as Chrome extensions (e.g., presence and structure of `manifest.json`) and rejects invalid uploads.
- The system reduces supply-chain risk by restricting analysis inputs to Chrome Web Store extensions only.

3.3 Security Scope and Limitations

The system enforces authentication and authorization at the application level. Analysis results are stored in a local SQLite database, which is not encrypted at rest.

As a result, users with direct operating system access to the host machine may read database contents outside of the application. Passwords are stored using secure cryptographic hashing.

Protecting data at rest (e.g., database encryption and OS-level access control) is considered out of scope for the current prototype and would be required for a production deployment.

4 Dependencies

- Linux OS (native, WSL, or virtual machine)
- Python 3.x
- pip
- Python packages listed in `requirements.txt`
- External tools used during analysis:
  - OWASP ZAP
  - OWASP Threat Dragon

5 Prerequisites

The application relies on external malware analysis APIs and an AI module. To enable these integrations, create a `.env` file (based on `.env.example`) and provide valid API keys:

- VirusTotal API key: https://www.virustotal.com
- OPSWAT MetaDefender API key: https://id.opswat.com/login
- SecureAnnex API key: https://app.secureannex.com/login
- Google Gemini API key (AI module): https://aistudio.google.com/app/api-keys

Make sure all required API keys are inserted before running the system, as several components depend on them.

6 Deployment Guide (Build & Run)

This document describes how to deploy and run the CETRA application from source in a development environment.

The instructions assume a Linux-based system (native Linux, WSL, or virtual machine).

6.1 1. Dependencies

See `docs/dependencies.md`.

6.2 2. Prerequisites (.env)

See `docs/prerequisites.md`.

6.3 3. Install Dependencies

Install the required Python packages using pip:

```
pip install -r requirements.txt
```

6.4 4. Setup & Run

1. **Migrate Database**: Run `python manage.py migrate` to apply changes to the database.
2. **Run Tests (Optional)**: Run `python manage.py test` to ensure the system is stable.
3. **Create User**: Execute `python manage.py createsuperuser --username=joe --email=joe@example.com` to create a user in the Django database.
   - Follow the terminal instructions.
   - Bypass password strength validation when prompted.
4. **Start Server**: Run `python manage.py runserver` to launch the program.
   - Open your browser to the localhost address (usually http://127.0.0.1:8000).
   - Log in with the credentials you entered during the `createsuperuser` step.
5. **Security Note**: For security, too many incorrect login attempts will trigger a temporary lockout with a 60-second countdown before you can try again.

7 User Guide

This document describes how to use CETRA (Chrome Extension Threat & Risk Analyzer) from an end-user perspective.

7.1 Intended Users and Use Cases

CETRA is designed for anyone who wants to assess the security and trustworthiness of Chrome extensions.

Typical users include:

- **End users** who want to verify whether a Chrome extension is safe to install.
- **Students** learning about software security and malware analysis.
- **Security analysts** investigating extension permissions, risks, and behavior.
- **Developers** who want to inspect what their own extensions expose.

7.2 Getting Started

To use CETRA, the application must be running and the user must be authenticated. Refer to the Deployment Guide for setup instructions.

Once logged in, the user is presented with the main navigation interface.

7.3 Submitting an Extension for Analysis

The **Home** page serves as the primary analysis interface.

Users can submit a Chrome extension using one of the following methods:

- **Chrome Web Store ID**
- **ZIP file**
- **CRX file**

After submission, the system validates the input and initiates the analysis workflow.

If a report for the same extension exists from the last 30 days, the user may choose to either: - Open the existing report, or - Run a new analysis

Analysis typically takes a few minutes to complete.

7.4 Viewing Analysis Results

Once the analysis is complete, CETRA presents a detailed result view including:

- Overall verdict and threat score
- Requested permissions
- Findings from external analysis services
- Malware family and category information

The system aggregates results from multiple sources to provide a unified view.

7.5 History and Report Management

The **History** tab allows users to:

- View previously analyzed extensions
- Browse results using pagination
- Download analysis reports in JSON format

This enables users to retain and reuse analysis results for later review.

7.6 MITRE ATT&CK Analysis

The **MITRE ATT&CK** tab provides deeper behavioral analysis.

For supported reports, users can: - Initiate MITRE ATT&CK mapping - View detected tactics and techniques - Track analysis status (Not Started, Completed, Not Available)

MITRE analysis is based on sandbox and behavioral data from external services.

7.7 Settings and Interface Features

The **Settings** page allows users to:

- Toggle between light and dark mode themes
- Adjust basic user interface preferences

Users can log out securely at any time using the logout option in the navigation bar.

7.8 Security Notes

- Authentication is required to access all analysis and reporting functionality.
- Multiple failed login attempts will trigger a temporary lockout with a cooldown period to reduce brute-force attempts.

7.9 Summary

Using CETRA consists of submitting a Chrome extension, waiting for the analysis to complete, and reviewing the aggregated results. The system is designed to be simple to use while providing meaningful security insights into Chrome extensions.

8 Threat Modeling

8.1 Overview

Threat modeling was performed to identify and assess potential security threats to the CETRA system. The analysis focused on user input, the Web UI, backend API routing, and external API integrations.

The threat modeling process was conducted using OWASP Threat Dragon and follows a STRIDE-based approach.

8.2 Identified Threats and Mitigations

8.2.1 User Input

8.2.1.1 Denial of Service (DDoS)

- **Threat Type:** Denial of Service
- **Description:** An attacker may attempt to overwhelm the system by submitting excessive files or requests.
- **Mitigation:** The system restricts users to submitting only one file at a time.
- **Risk Score:** 59 (Medium)

8.2.1.2 SQL Injection

- **Threat Type:** Tampering
- **Description:** Malicious input could be used to manipulate SQL queries.
- **Mitigation:** Input validation is enforced.
- **Risk Score:** 80 (High)

8.2.1.3 SQL Injection – Authentication

- **Threat Type:** Tampering / Authentication Bypass
- **Description:** Improper input handling may allow bypassing authentication.
- **Mitigation:** Input validation is enforced.
- **Risk Score:** 80 (High)

8.2.2 Web UI Frontend

8.2.2.1 Spoofing

- **Threat Type:** Spoofing
- **Description:** Potential spoofing attempts against the frontend.
- **Mitigation:** Considered non-threatening as the software is not publicly deployed.
- **Risk Score:** 10 (Low)

8.2.2.2 Server Information Disclosure

- **Threat Type:** Information Disclosure
- **Description:** Server version information may be leaked via HTTP headers.
- **Mitigation:** Server configuration can be modified to remove or obfuscate identifying headers.

- **Risk Score:** 10 (Low)

8.2.3 API Router

8.2.3.1 File Download Information Disclosure

- **Threat Type:** Information Disclosure
- **Description:** Improper file handling may expose sensitive files.
- **Mitigation:** Files should be validated before being downloaded.
- **Risk Score:** 20 (Low)

8.2.4 API Interfaces

8.2.4.1 Man-in-the-Middle (MITM)

- **Threat Type:** Tampering
- **Description:** External API responses may be altered during transmission.
- **Mitigation:** API results should be validated before use.
- **Risk Score:** 40 (Low)

8.3 Risk Assessment

Risk levels were assessed using CVSS v4.0 scoring where applicable. High-risk threats were identified pr

9 Vulnerability Scanning

9.1 Overview

Security testing was performed to identify vulnerabilities in the CETRA Web UI. The testing focused on detecting common web application vulnerabilities and misconfigurations.

9.2 Tooling

- **Tool:** OWASP ZAP
- **Target:** CETRA Web UI (local development instance)

9.3 Scan Context and Authentication Scope

OWASP ZAP scanning was performed against an authenticated session of the CETRA Web UI. Authentication was required in order to access protected functionality, such as submitting extensions for analysis and viewing stored reports.

Unauthenticated scanning was not sufficient to exercise the full attack surface, as the application enforces access control on core functionality.

As a result, all identified SQL injection findings were discovered in an authenticated context. The authentication mechanism therefore mitigates the risk of unauthenticated exploitation. However, these findings remain valid, as SQL injection vulnerabilities can still be exploited by an authenticated attacker if proper query handling and input validation are not enforced.

9.4 Identified Vulnerabilities

9.4.1 High Severity

Vulnerability	Severity	Count
SQL Injection	High	4
SQL Injection – Authentication Bypass	High	4

9.4.2 Medium Severity

Vulnerability	Severity	Count
Content Security Policy (CSP) Header Not Set	Medium	14

9.4.3 Low Severity

Vulnerability	Severity	Count
Cookie No HttpOnly Flag	Low	10
Server Leaks Version Information via “Server” Header	Low	17
X-Content-Type-Options Header Missing	Low	1

9.4.4 Informational

Finding	Count
Authentication Request Identified	4
Session Management Response Identified	12
User Agent Fuzzer	153

9.5 High-Severity Vulnerabilities: Mitigation and Justification

9.5.1 SQL Injection

- **Description:** User-controlled input may be used to inject malicious SQL queries.
- **Impact:** Unauthorized access to or manipulation of stored data.
- **Mitigation:**
  - Enforce strict server-side input validation.
  - Use parameterized queries for all database operations.
  - Restrict access to vulnerable endpoints through authentication and authorization mechanisms.
- **Status:** Identified in an authenticated scan context; mitigation is required before production deployment.

9.5.2 SQL Injection – Authentication Bypass

- **Description:** Improper query handling may allow attackers to bypass authentication mechanisms.
- **Impact:** Unauthorized access to protected functionality and stored reports.
- **Mitigation:**
  - Enforce strict server-side input validation.
  - Ensure authentication and authorization checks occur before database queries on protected routes.
  - Use parameterized queries to prevent query manipulation.
- **Status:** Identified in an authenticated scan context; the authentication mechanism mitigates unauthenticated exploitation, but mitigation is required before production deployment.

9.6 Summary

- **Total identified vulnerabilities and threats:** 13
- **Mitigated vulnerabilities and threats:** 7
- **Residual vulnerabilities and threats:** 6

The remaining residual vulnerabilities are considered acceptable within the scope of the prototype and do not pose a critical risk in a controlled development environment.

9.7 Known Limitations and Future Work

- Lack of in-house malware analysis and verdict generation.
- Heavy reliance on external services such as VirusTotal.
- Future improvements include containerization and implementing structured return values to enable pipeline-based execution.

9.8 Conclusion

CETRA provides a valid foundational prototype for further development. The security analysis highlights areas requiring hardening before production deployment, while demonstrating a solid understanding of threat modeling and vulnerability management principles.