CIRCL
Computer Incident
Response Center
Luxembourg

Vulniverse

Vulniverse

A Modular Editor for Vulnerability Records

https://github.com/vulnerability-lookup/vulniverse – https://gcve.eu/

info@gcve.eu - @gcve@social.circl.lu

GCVE Workshop Vulnopticon 2026

CIRCL https://www.circl.lu

TLP:CLEAR

Motivation and Design

Why a New Vulnerability Editor?

Experience with Vulnogram

- Powerful editor for CVE records
- UI and workflow are difficult to customize
- Additional functionality requires modifications inside the editor
- Integration into Vulnerability-Lookup therefore creates maintenance overhead (especially when updating)

What we need

- Reusable editor core
- Backend-independent persistence
- Schema-driven record handling
- Modules and panels for extensions
- CVE and GCVE support
- Easy embedding into existing applications

Vulniverse separates the editor from the surrounding application, allowing platforms such as Vulnerability-Lookup to integrate, customize and update it independently.

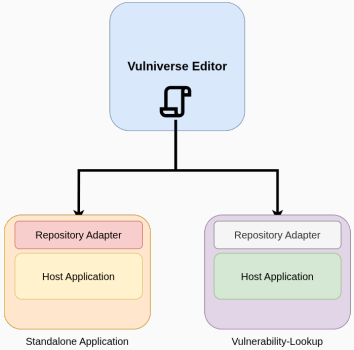
Design Decisions

Observed Problem		Vulniverse Design Decision
Editor UI and workflow are difficult to adapt to different applications	⟶	**Reusable and embeddable editor core**
Backend and persistence logic become coupled to the editor	⟶	**Repository abstraction** with a small, well-defined contract
Application-specific modifications make upstream updates difficult	⟶	**Modules** extend functionality without modifying the core
Vulnerability record formats and schemas evolve over time	⟶	**Schema-driven record handling and validation**
Different ecosystems require CVE, GCVE and additional metadata	⟶	**Format-independent architecture** with reusable components
Large vulnerability records are difficult to navigate as a single form	⟶	**Structured sections and navigation**

Architecture

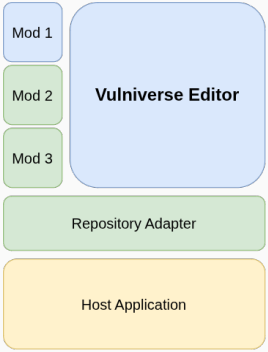
Backend Independence through a Repository Contract

- The editor does **not communicate directly** with a specific backend or database BUT uses **repository contract**.
- Each host application provides an implementation of this contract.
- The same editor can therefore be used with:
  - the standalone Vulniverse backend,
  - other vulnerability-management platforms (e.g. Vulnerability-Lookup)



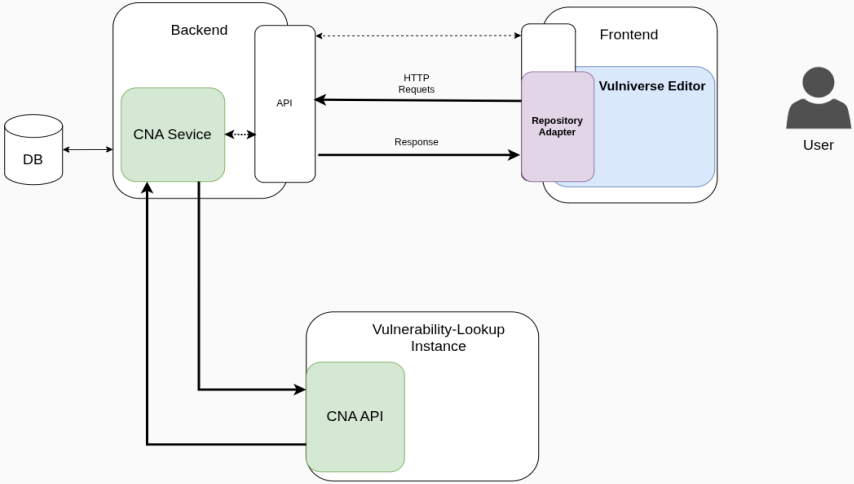
Extensibility through Modules

- Application-specific functionality should **not require changes to the editor core**.
- **Modules** add functionality and integration logic.
- Examples:
  - CPE integration,
  - additional record metadata.



New functionality is added **around the editor core**, rather than by maintaining a customized fork of it.

Example: Connect to Vulnerability-Lookup



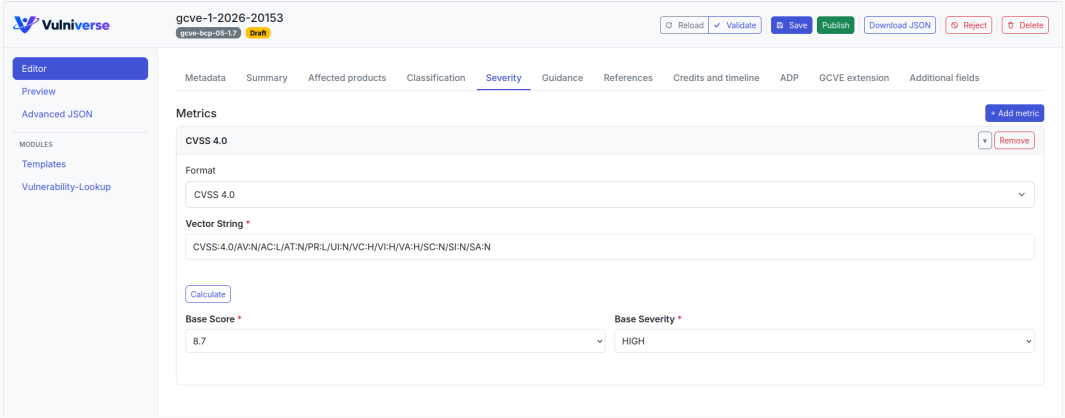
Schema-Driven Record Editing

- The record schema defines the **structure and constraints** of the data.
- Schema information is mapped to **reusable editor components**.
- Validation follows the corresponding CVE or GCVE-compatible schema.
- Complex structures can use **specialized components** instead of generic fields.
- Changes to the record format can be incorporated without rebuilding the complete editor.

The **schema defines the data model**; Vulniverse defines how that model is presented and edited.

Editing (G)CVE Records

Demo



https://github.com/vulnerability-lookup/vulniverse

TLP:CLEAR

Contacts

- Issues, feedback or ideas: `https://github.com/vulnerability-lookup/vulniverse`
- GCVE **@@gcve@social.circl.lu**