Whitepaper

The First Open Data Knowledge Base

Contributing to Open SCA and **Standardized SBOM generation.**

SCANOSS

Content

Questions?

Please get in touch through https://www.scanoss.com or try the Open Source Knowledge Base at https://osskb.org.

Background

Software Composition Analysis (SCA) tools perform source-code analysis, comparison and identification of Open Source components. Sadly, none of the SCA vendors have embraced Open Source themselves, most of their tooling consists of proprietary code and their OSS Knowledge Bases are also closed.

Most SCA vendors primarily target large (enterprise) organizations, are very expensive, and thus generally not economically viable for smaller companies who might be a critical part of several larger software distribution chains. As smaller companies for these reasons can't get access to proper SCA tooling, large organizations will have to carry the expense of auditing their suppliers' software products. Even for a larger organization this leads to a high cost of OSS governance, as they not only have to deal with audits of their own software, but also with auditing software provided by their suppliers.

Next to the challenging proprietary nature of SCA tooling, and supplier audits there's one more struggle. Most, if not all, OSS Knowledge Bases are completely closed, meaning that every SCA tool bases its assessments on different data sources and therefore their results cannot be compared. This complicates assessments and OSS sourcecode identification, and limits accuracy and quality, eventually driving the high costs of OSS governance.

The viability and security of open-source packages are cited as top concerns by most of the respondents to a 2020 Gartner survey: Market Guide for Software Composition Analysis.

We need common ground, no further differentiation!

OSS management is not a differentiator for software-driven businesses.
It's a highly necessary process to control licensing, costs and mitigate risk.

Thus, organizations are not looking to do things differently than others, they instead seek to establish best practice standards and comparable results. As part of the daily challenges with OSS governance, companies often seek common ground with other companies in their ecosystem. This has propelled a number of global community projects, focused on standardization of Open Source Compliance, like Linux Foundation's OpenChain and SPDX.

The core value of these OSS-projects is completely misaligned with the business models of most SCA-vendors, who continue to push proprietary tooling, closed data and even invoke software patents throughout the field. By doing so, they are keeping SCA-vendors far from participating in any industrial standardization forum.

Auditing VS. Validation

With a truly Open SCA Platform, software developers could fully embrace OSS compliance, delivering their products alongside their SBOM declarations.

So, we argue, there is a clear need for a developer-centric focus when it comes to source-code validation and SBOM generation. By 'starting left' and moving source-code validation to earlier stages in the development process, we could drastically reduce the cost of OSS governance and turn expensive OSS Compliance auditing processes into simple OSS, live, 'always on' source-code validation.

It's time for a truly Open Source SCA Platform

We believe that now is the time to reinvent Software Composition Analysis by releasing the first Open Source SCA Platform and Open Data OSS Knowledge Base. Supported by an:

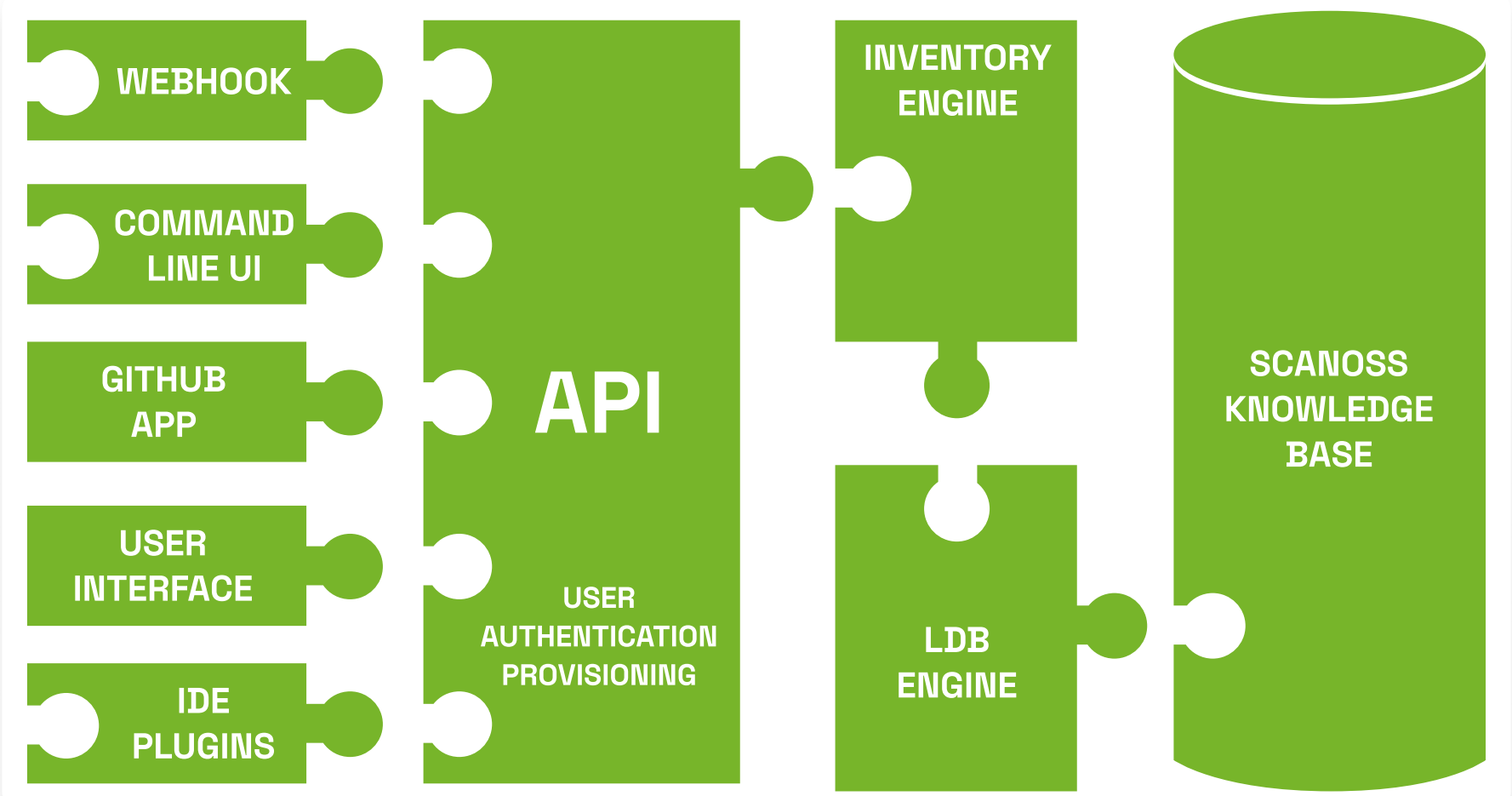
- Open Source Data Mining tool (minr)
- Open Source Database Engine (ldb)
- Open Source Inventory Engine
- Open Source RESTful API
- Open Source Webhook and Command Line Interface (CLI)

The goal of the platform is to start left and put a clear emphasis on the foundation of reliable SCA, the SBOM. An SBOM that does not require a small army of auditors to make it usable. An SBOM that is based on 'live' code and that is 'always on'.

Being specifically designed for modern development environments and DevOps teams, our platform uses open standards like SPDX or CycloneDX (in their JSON variant) as the core SBOM storage format. XML variants are also supported both for importation and exportation of SBOMs.

The complete SCANOSS platform is released as Open Source and available on our Github at https://github.com/scanoss.

The pillar of the architecture is an Open & compliant RESTful API that exposes functionality for the inventory engine to be interfaced from CI/CD tools as well as a full set of features for OSS auditing and other user interfaces.

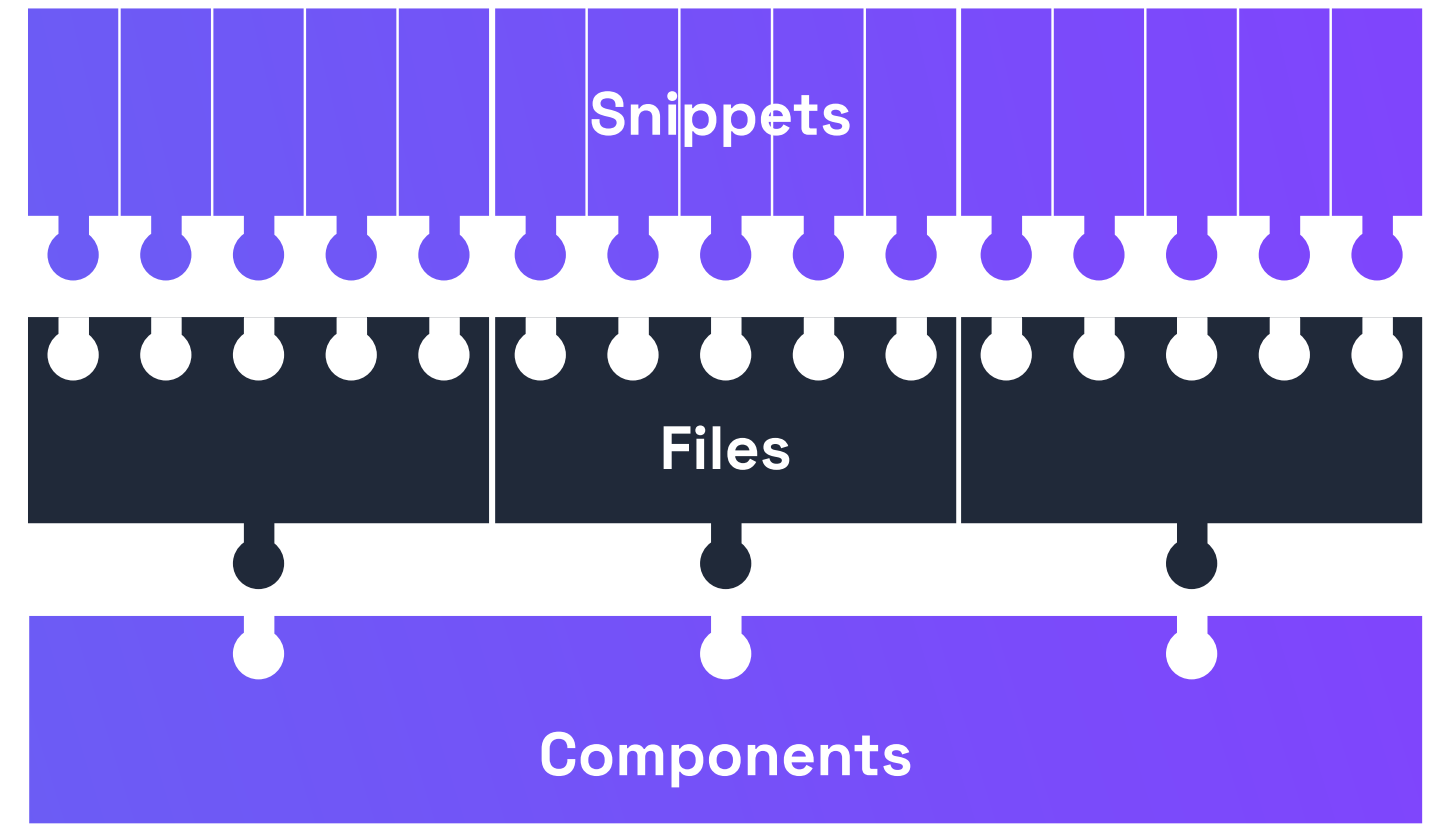


Client-side applications and middleware connect to the RESTful API via https. The API handles interaction with the Inventory Engine, which uses the LDB interfaces to query the database.

The First Open OSS Knowledge Base

From now on called 'OSSKB'. Try it at https://osskb.org.

Most Open Source projects continuously release new versions or updates for their project. Each individual version becomes a unique "component". Open Source components are usually composed of a series of different files, and each file is then composed of a series of code snippets.



There are millions of OSS-components available today, and getting all that information in a single & up to date database is challenging. Moreover, storing all source code in a single repository will most likely cause conflicts with their respective license conditions.

We named this Knowledge Base "OSSKB", which stands for Open Source Software Knowledge Base. In order to satisfy the OSSKB' requirements of such a vast amount of data, while delivering the fastest possible response times, we were forced to develop a specific database engine. This engine is called LDB and we released it as Open Source.

The OSSKB does not store source code, but instead source code metadata, which is linked to its cryptographic hashes (unique identifiers). The metadata stored in the OSSKB includes:

Components	**ID, name, author, url, license, dependencies.**
Files	**ID, component ID, file path, file size, license.**
Components	**ID, file ID, line number where snippet starts.**

The LDB database engine

The OSSKB contains millions of indexed components, which translates to billions of file identifiers and trillions of snippet identifiers. This vast amount of information challenges standard database engines, they often turn inefficient due to the overhead required to handle such a simplistic, yet voluminous use case.

In order to fulfill the requirements of such a high volume of data and specific requirements of the OSSKB, we developed the LDB database (Linked-list database), which we have released as Open Source: https://github.com/scanoss/ldb.

The LDB is a headless database management system focused in a singlekey, read-only application on vast amounts of data, while maintaining a minimal footprint and keeping system calls to a bare minimum. Information is structured using linked lists. Linked lists are mapped which eliminates the need of using data indexes.

Some of the features of the LDB are:

- Single, fixed size, numeric key (32-bit)
- Single-field records
- Larger keys also are supported by storing remaining data keys in data list
- No indexing overhead, instead List mapping is used
- Data tables define either fixed or variable-length data records

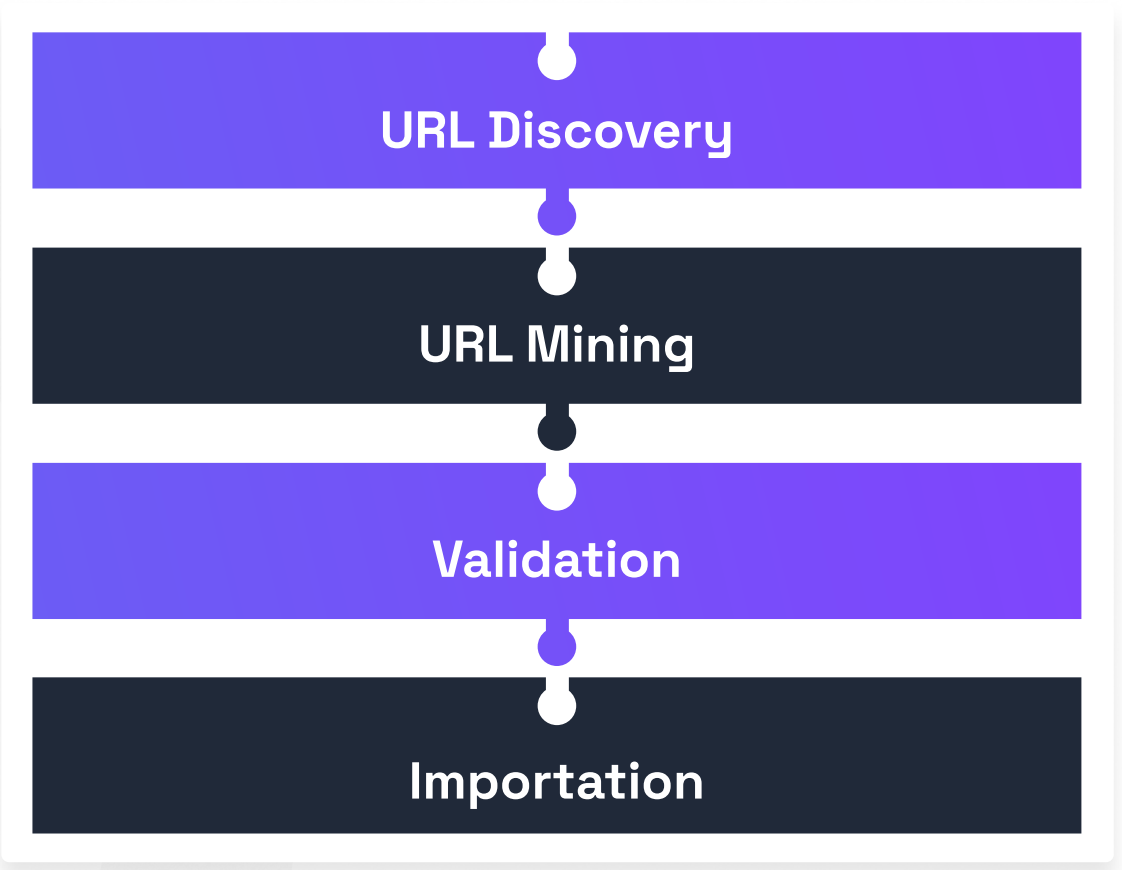
LDB features continued:

- Read-only, updateable by a single thread, which eliminates the overhead of semaphores
- Zlib data compression
- Data records are organized in a linked list
- C library for native development
- LDB shell allows interaction with external languages

Data mining cycle

Minr is the command-line tool used for mining data into the OSSKB. Updates are managed in cycles which perform component discovery, mining (downloading and indexing), validation and importation.

The OSSKB website features a request form that allows automatic inclusion of components into the database, in the case they have not been automatically downloaded. A great use-case to get the OSS community involved in keeping the LDB up to date.



The entire data mining codebase is released as Open Source in Github:

https://github.com/scanoss

Open Source client software

The SCANOSS Open Source platform includes a Webhook for easy integration into modern development and DevOps environments, as well as a command-line interface called scanner.py. The SCANOSS Scanner is a simple Python script that performs direct, recursive scanning of the provided directory. The scan is performed using the SCANOSS API against the OSSKB (Open Knowledge Base).

Scanner.py CLI in action:

```
$ Scanner.py
Scanning directory: .
Saved generated WFP File in ‘scan_wfp’
{
“./scanner.py”: [
{
“id”:”file”,
“elapsed”:”0.000035s”,
“lines”:”all”,
“matched”:”100%”,
“vendor”:”scanoss”,
“component”:”scanner.py”,
“version”:”1.0”,
“url”:”https://github.com/scanoss/scanner.py/archive/1.0.tar.gz”,
“file”:”scanner.py-1.0/scanner.py”,
“size”:”7652”,
}
]
}
```

Get involved

The SCANOSS Platform is made entirely available entirely as Open Source. The collaboration guidelines are available in the source code tree. Questions and suggestions are welcome at https://www.scanoss.com

Get in touch

SCANOSS offers commercial agreements with access to its complete Knowledge Base, additional features and Service Level Agreements.

Please contact info@scanoss.com for further information.

Published on 2020-07-08 by SCANOSS.com

SCANOSS