KOREA UNIVERSITY
Center for Software Security and Assurance

IoTcube 2.0 Hatbom
User Manual-eng

V0.0.3

2025-09-25

KOREA UNIVERSITY
Center for Software Security and Assurance

[Supported Languages]

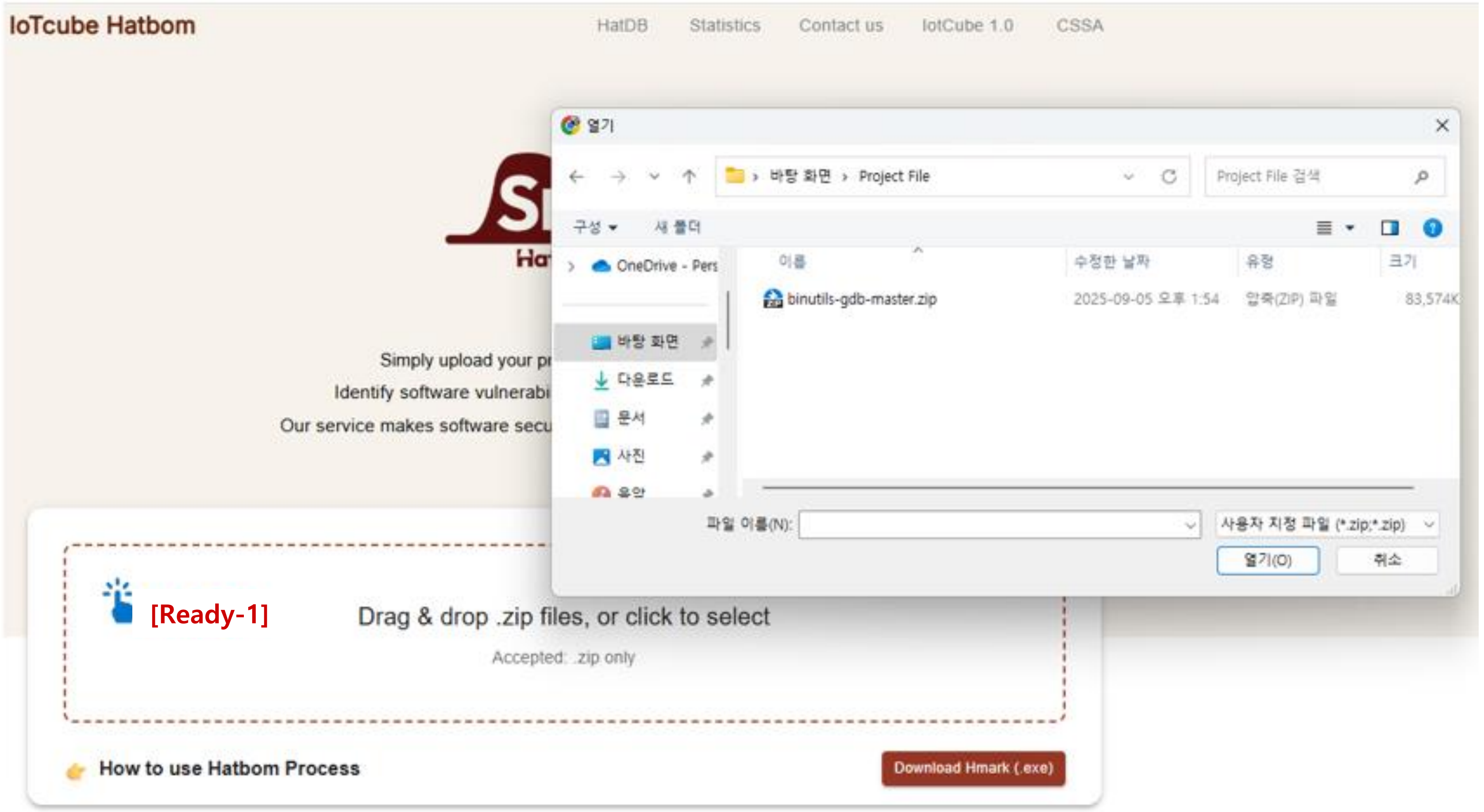
※ Supported Languages by Tool (as of September 8, 2025)→ The range of supported languages will be gradually expanded.

SBOM	OSS Dependency Graph	Vulnerability	Static Analysis
C/C++, java, python, go, php	C/C++	C/C++, java, python	C/C++

①SBOM Step ② Vulnerability Step ③VEX Step

[Ready]Project zip File Drag&Drop

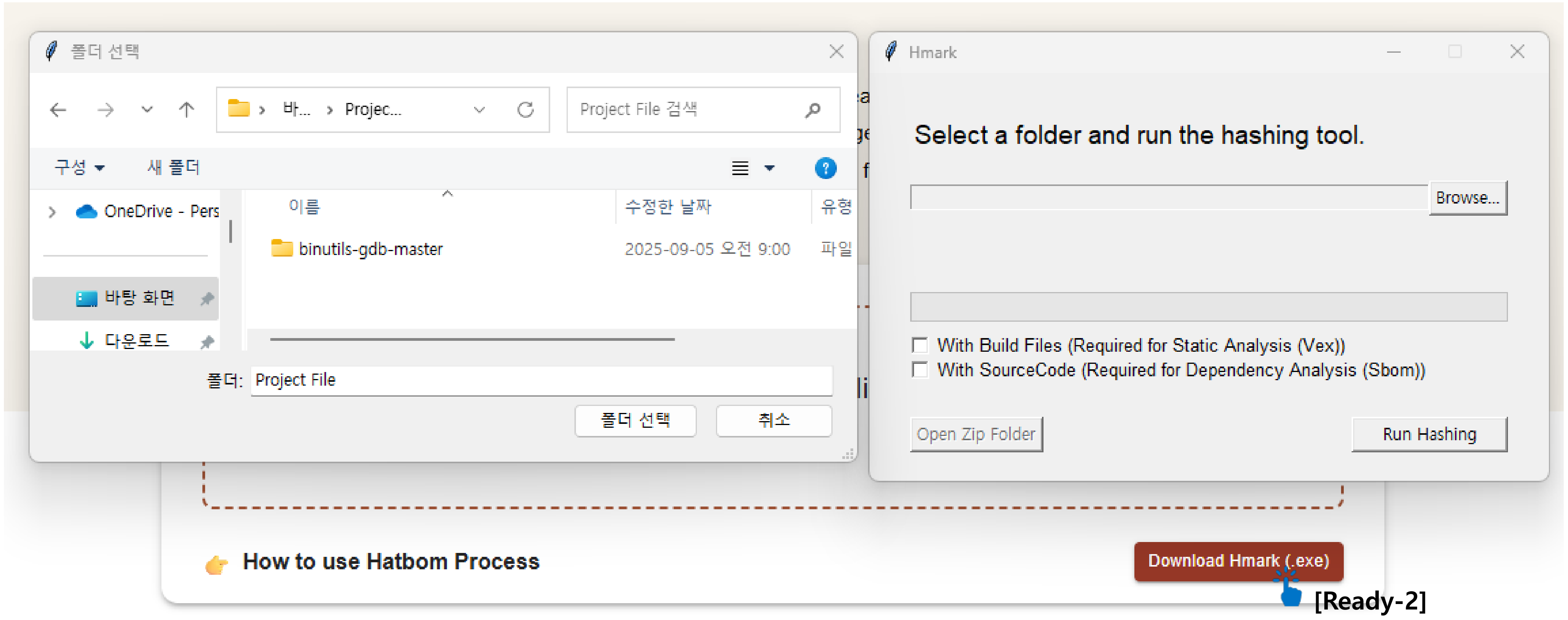
[Ready-1] Drag & drop the project folder containing the source code as a .zip file.



KOREA UNIVERSITY
Center for Software Security and Assurance

[Read] Project zip file drag & drop

[Ready-2] Drag & drop the .zip file generated after hashing with “Download Hmark.”



폴더 선택

Project File 검색

이름 | 수정한 날짜 | 유형

binutils-gdb-master 2025-09-05 오전 9:00 파일

폴더: Project File

Hmark

Select a folder and run the hashing tool.

☐ With Build Files (Required for Static Analysis (Vex))
☐ With SourceCode (Required for Dependency Analysis (Sbom))

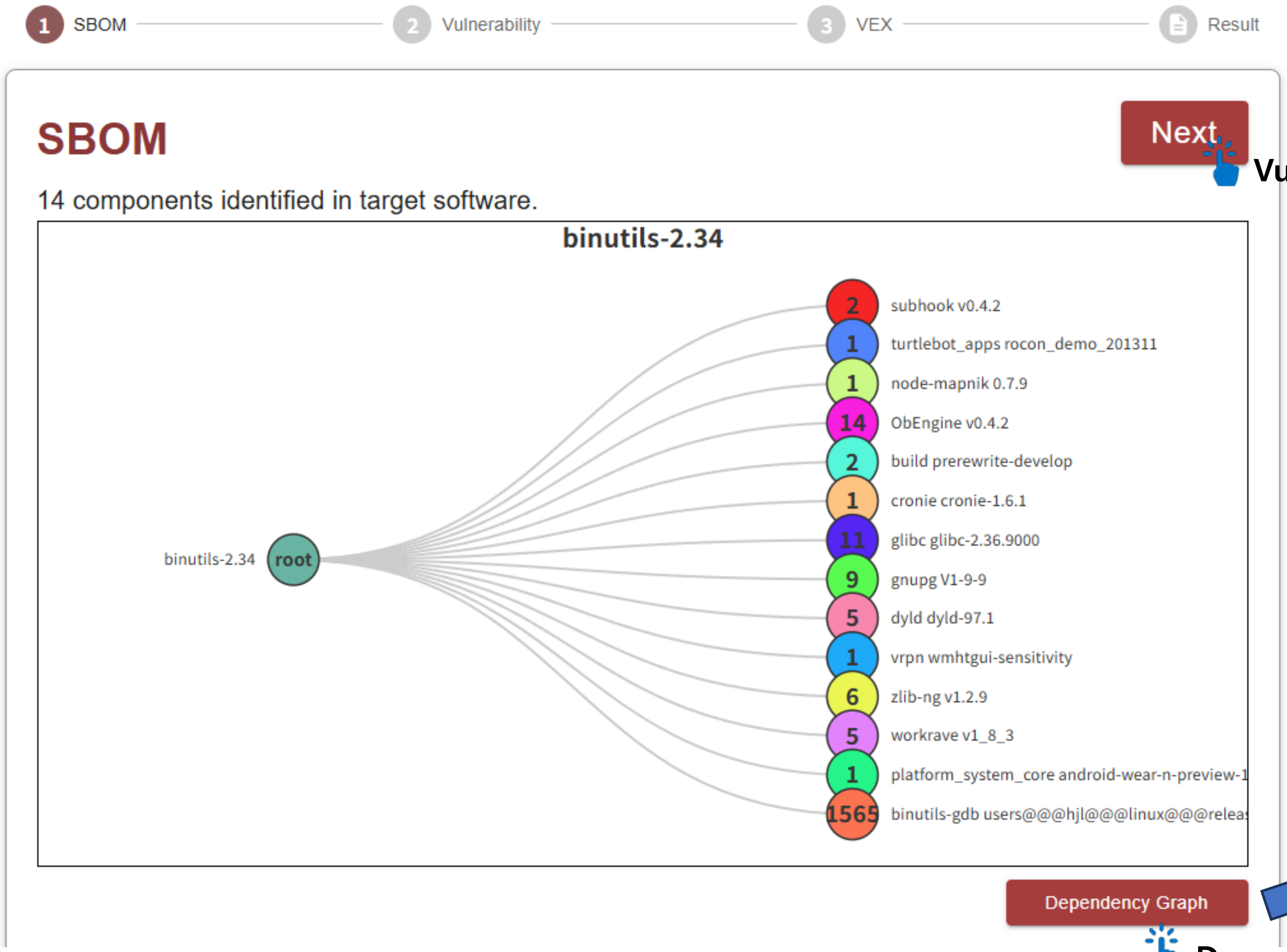
-Hmark is a local hashing program, which has the advantage of not requiring you to provide your source code to the platform (code privacy).
-If you check With Build Files, the binary files required for static analysis will be included (for completed C/C++ build projects only).
-If you check With Source Code, the source code needed to generate the dependency graph will also be included (C/C++ only).

KOREA UNIVERSITY
Center for Software Security and Assurance

1. SBOM Step

The SBOM Step is the process that detects Open Source Software (OSS) and automatically generates the corresponding SBOM document.
You can download the SBOM document using the download button below.

For C/C++ projects, if you provide the source code, the dependencies of the detected OSS can also be analyzed.



SBOM

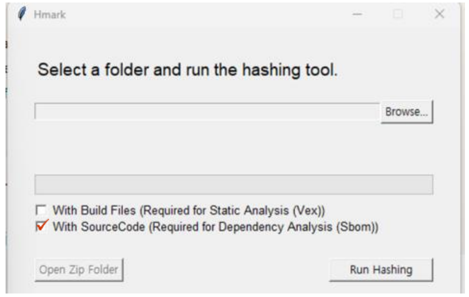
14 components identified in target software.

binutils-2.34

binutils-2.34 root

2	subhook v0.4.2
1	turtlebot_apps rocon_demo_201311
1	node-mapnik 0.7.9
14	ObEngine v0.4.2
2	build prerewrite-develop
1	cronie cronie-1.6.1
11	glibc glibc-2.36.9000
9	gnupg V1-9-9
5	dyld dyld-97.1
1	vrpn wmhtgui-sensitivity
6	zlib-ng v1.2.9
5	workrave v1_8_3
1	platform_system_core android-wear-n-preview-1
1565	binutils-gdb users@@@hjl@@@linux@@@releas

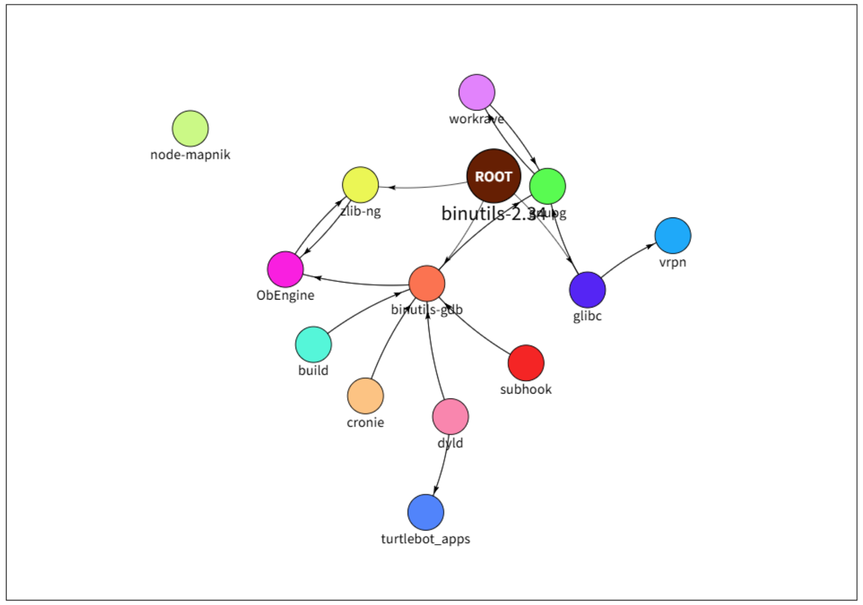
Vulnerability Step



Select a folder and run the hashing tool.

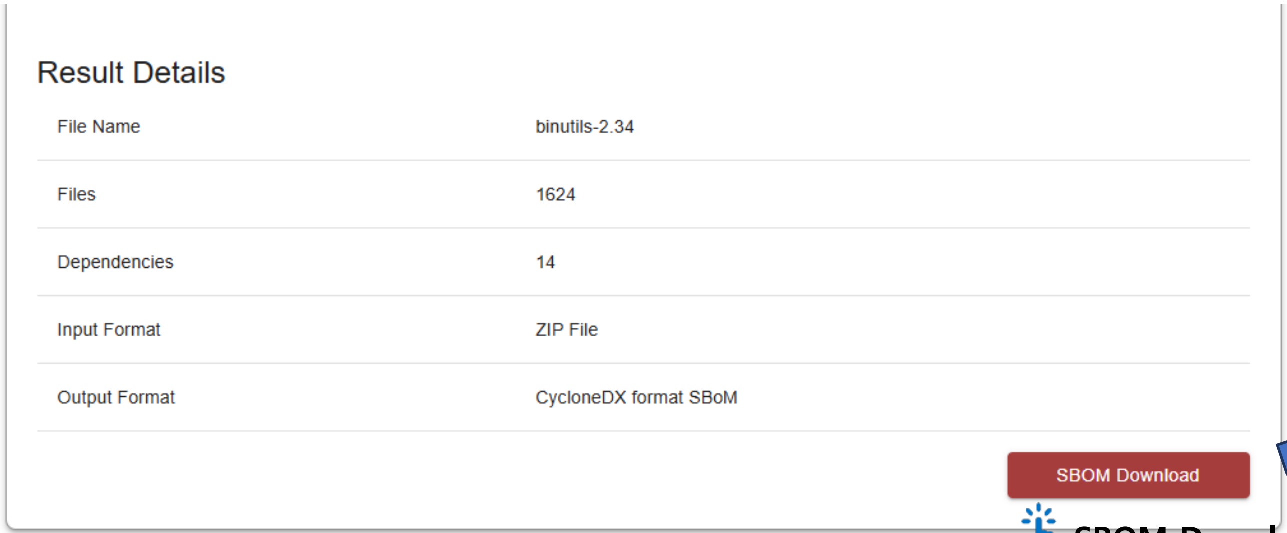
☐ With Build Files (Required for Static Analysis (Vex))
☑ With SourceCode (Required for Dependency Analysis (Sbom))

Dependency graph for binutils-2.34



Dependency Graph

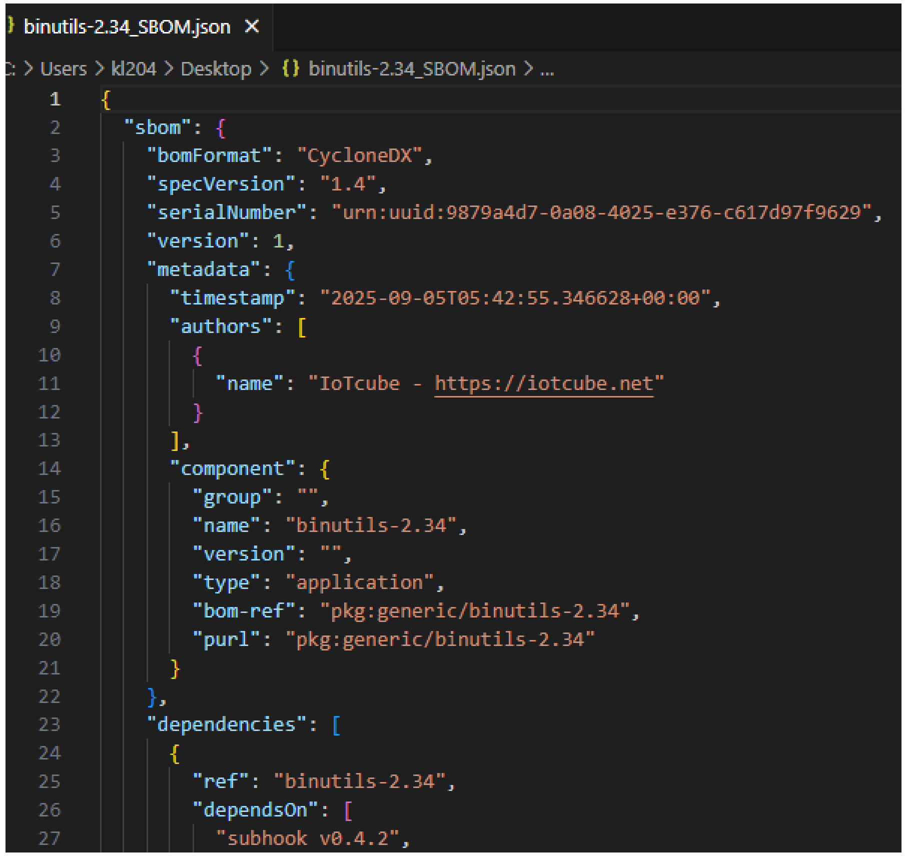
1. SBOM Step



Result Details

File Name	binutils-2.34
Files	1624
Dependencies	14
Input Format	ZIP File
Output Format	CycloneDX format SBoM

SBOM Download


binutils-2.34_SBOM.json
C: > Users > kl204 > Desktop > {} binutils-2.34_SBOM.json > ...

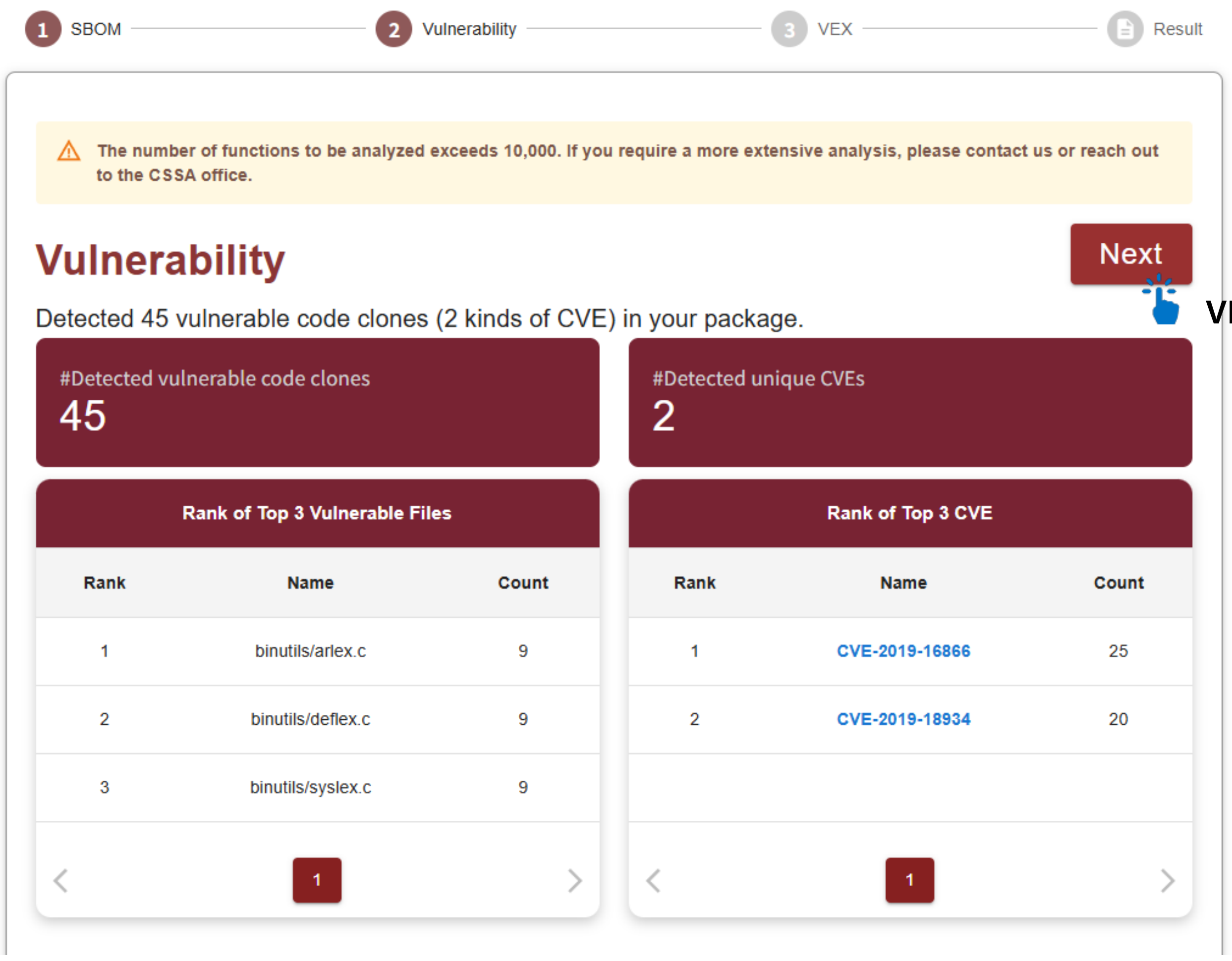
```
{
  "sbom": {
    "bomFormat": "CycloneDX",
    "specVersion": "1.4",
    "serialNumber": "urn:uuid:9879a4d7-0a08-4025-e376-c617d97f9629",
    "version": 1,
    "metadata": {
      "timestamp": "2025-09-05T05:42:55.346628+00:00",
      "authors": [
        {
          "name": "IoTcube - https://iotcube.net"
        }
      ],
      "component": {
        "group": "",
        "name": "binutils-2.34",
        "version": "",
        "type": "application",
        "bom-ref": "pkg:generic/binutils-2.34",
        "purl": "pkg:generic/binutils-2.34"
      }
    },
    "dependencies": [
      {
        "ref": "binutils-2.34",
        "dependsOn": [
          "subhook v0.4.2",
```

2. Vulnerability Step

- The Vulnerability Step is where vulnerabilities are detected. You can check which vulnerabilities exist in the provided project.

- CVSS (Common Vulnerability Scoring System) represents the severity score of a vulnerability.
The rating ranges are as follows: None (0), Low (0.1–3.9), Medium (4.0–6.9), High (7.0–8.9), and Critical (9.0–10.0).

- To generate a VEX document, you need to select the detected CVEs.



⚠ The number of functions to be analyzed exceeds 10,000. If you require a more extensive analysis, please contact us or reach out to the CSSA office.

Vulnerability

Detected 45 vulnerable code clones (2 kinds of CVE) in your package.

#Detected vulnerable code clones
45

#Detected unique CVEs
2

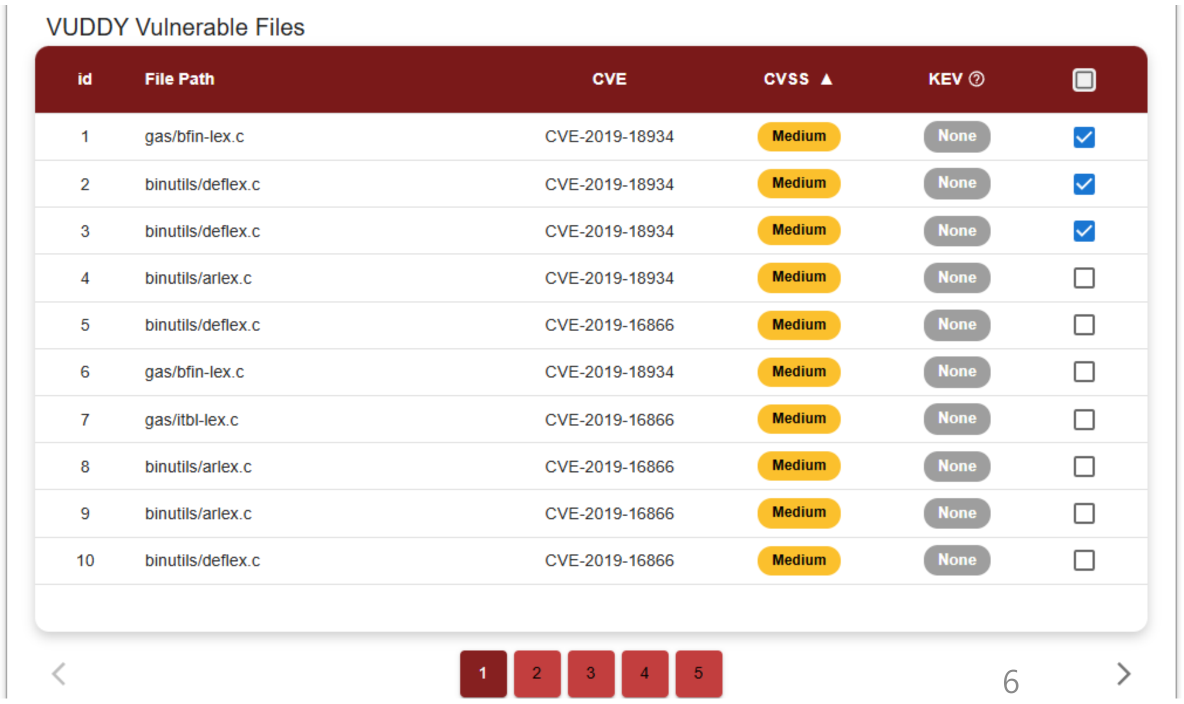
Rank of Top 3 Vulnerable Files

Rank	Name	Count
1	binutils/arlex.c	9
2	binutils/deflex.c	9
3	binutils/syslex.c	9

Rank of Top 3 CVE

Rank	Name	Count
1	CVE-2019-16866	25
2	CVE-2019-18934	20

VEX Step



VUDDY Vulnerable Files

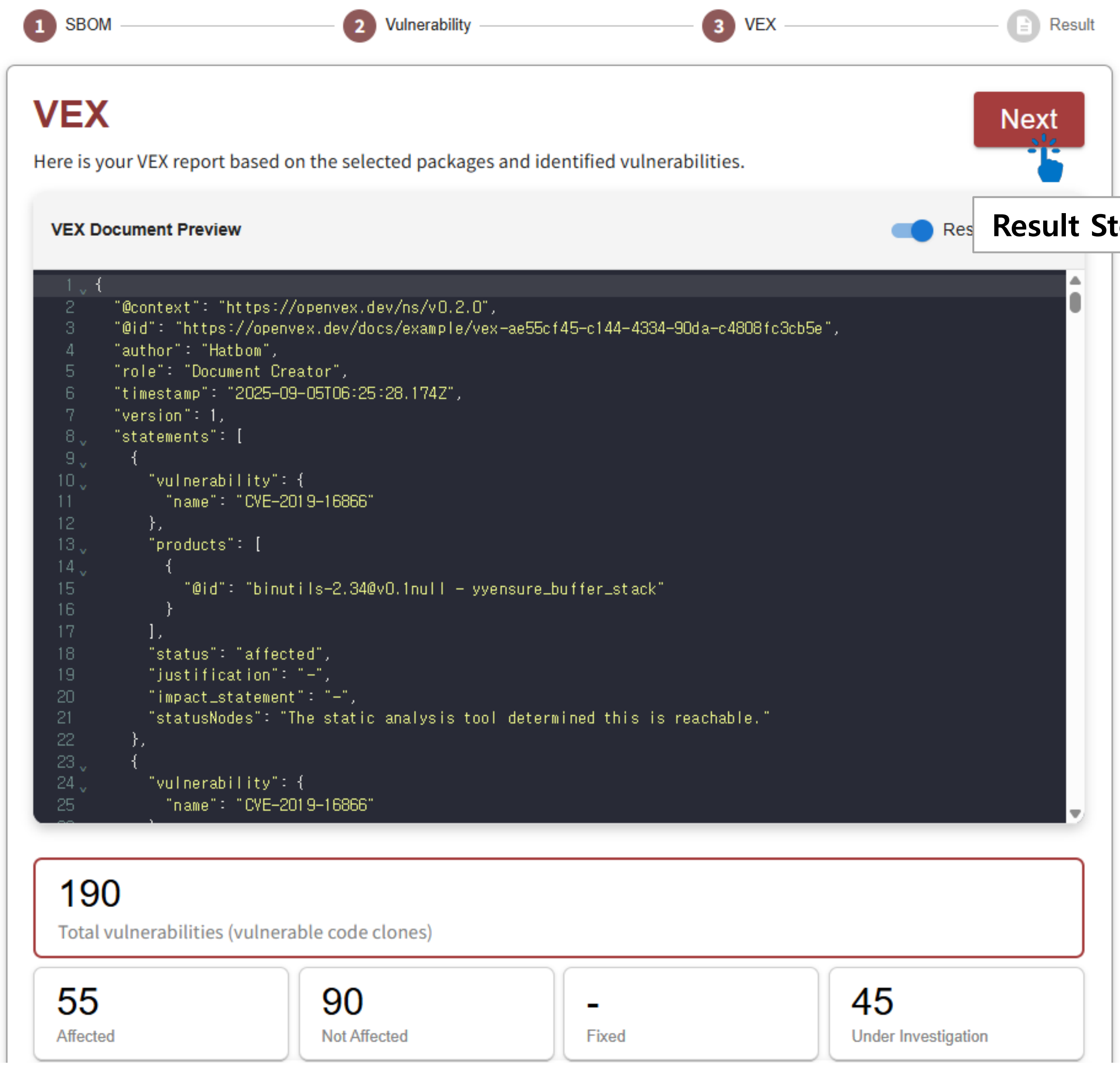
id	File Path	CVE	CVSS ▲	KEV ⓘ	☐
1	gas/bfin-lex.c	CVE-2019-18934	Medium	None	✓
2	binutils/deflex.c	CVE-2019-18934	Medium	None	✓
3	binutils/deflex.c	CVE-2019-18934	Medium	None	✓
4	binutils/arlex.c	CVE-2019-18934	Medium	None	☐
5	binutils/deflex.c	CVE-2019-16866	Medium	None	☐
6	gas/bfin-lex.c	CVE-2019-18934	Medium	None	☐
7	gas/itbl-lex.c	CVE-2019-16866	Medium	None	☐
8	binutils/arlex.c	CVE-2019-16866	Medium	None	☐
9	binutils/arlex.c	CVE-2019-16866	Medium	None	☐
10	binutils/deflex.c	CVE-2019-16866	Medium	None	☐

KOREA UNIVERSITY
Center for Software Security and Assurance

3. VEX Step

The VEX Step is where the validity of detected vulnerabilities is documented. This step allows you to edit and add detailed information for each vulnerability.

For C/C++ projects, if the project is already built, static analysis will be performed automatically.



VEX

Next

Here is your VEX report based on the selected packages and identified vulnerabilities.

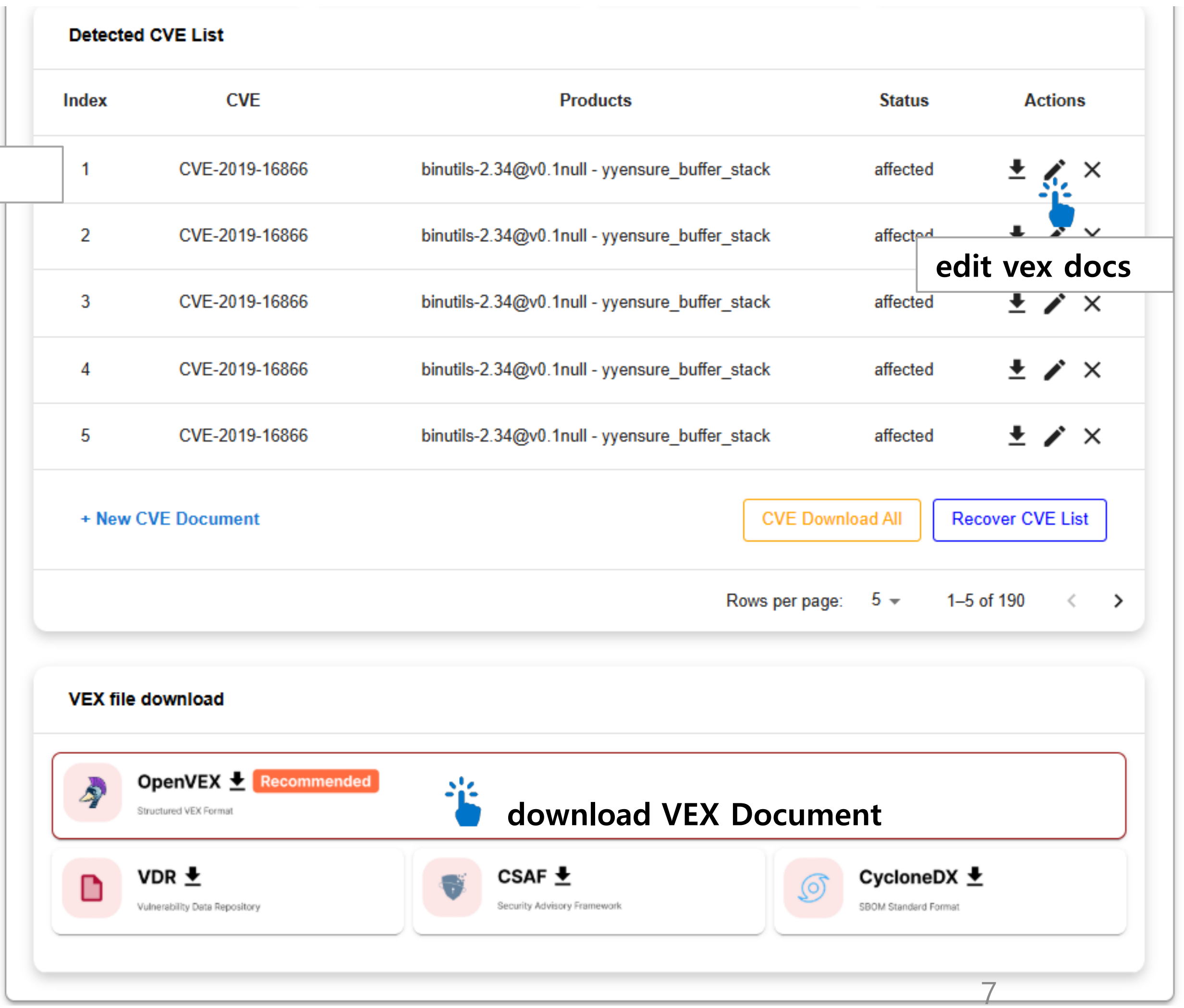
Result Step

VEX Document Preview

```
{
  "@context": "https://openvex.dev/ns/v0.2.0",
  "@id": "https://openvex.dev/docs/example/vex-ae55cf45-c144-4334-90da-c4808fc3cb5e",
  "author": "Hatbom",
  "role": "Document Creator",
  "timestamp": "2025-09-05T06:25:28.174Z",
  "version": 1,
  "statements": [
    {
      "vulnerability": {
        "name": "CVE-2019-16866"
      },
      "products": [
        {
          "@id": "binutils-2.34@v0.1null - yyensure_buffer_stack"
        }
      ],
      "status": "affected",
      "justification": "-",
      "impact_statement": "-",
      "statusNodes": "The static analysis tool determined this is reachable."
    },
    {
      "vulnerability": {
        "name": "CVE-2019-16866"
```

190
Total vulnerabilities (vulnerable code clones)

55	90	-	45
Affected	Not Affected	Fixed	Under Investigation



Detected CVE List

Index	CVE	Products	Status	Actions
1	CVE-2019-16866	binutils-2.34@v0.1null - yyensure_buffer_stack	affected	
2	CVE-2019-16866	binutils-2.34@v0.1null - yyensure_buffer_stack	affected	
3	CVE-2019-16866	binutils-2.34@v0.1null - yyensure_buffer_stack	affected	
4	CVE-2019-16866	binutils-2.34@v0.1null - yyensure_buffer_stack	affected	
5	CVE-2019-16866	binutils-2.34@v0.1null - yyensure_buffer_stack	affected	

edit vex docs

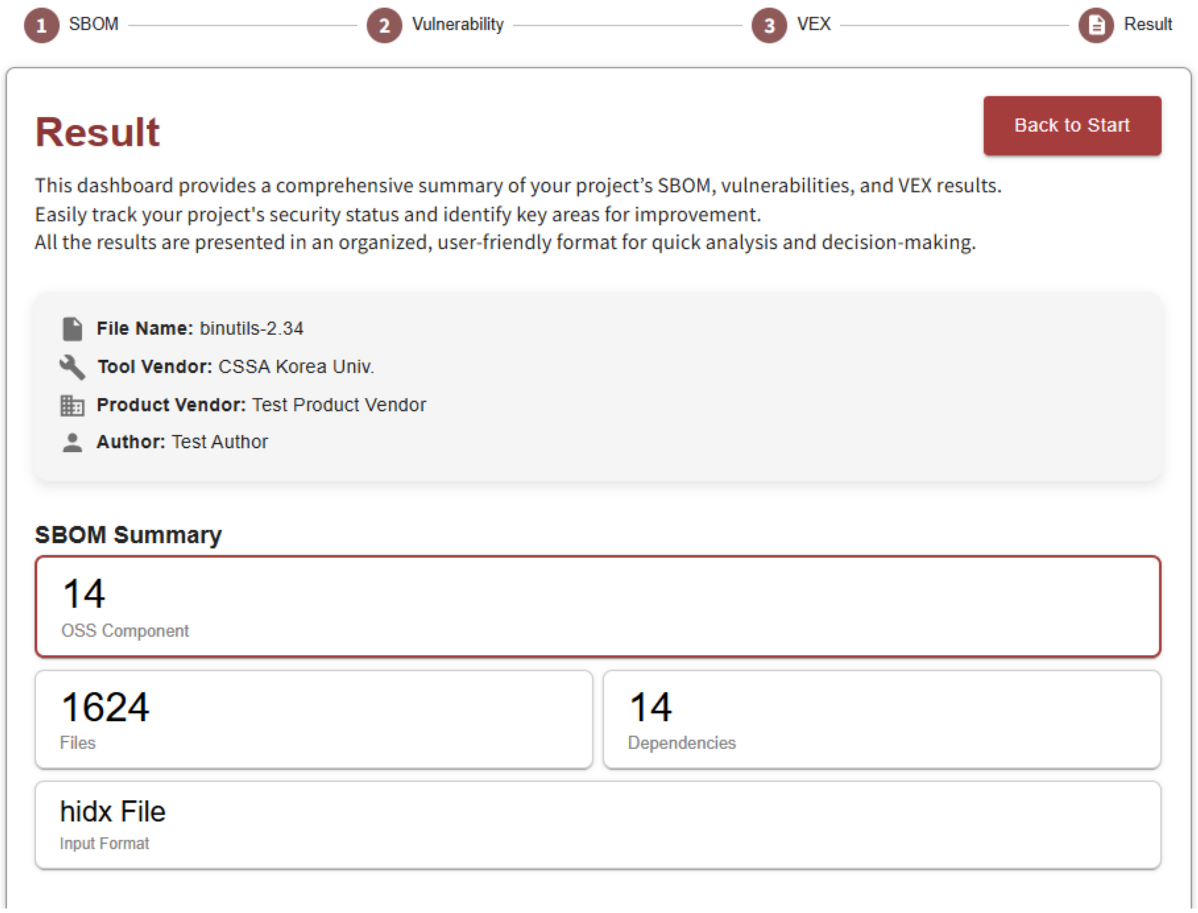
VEX file download

download VEX Document

KOREA UNIVERSITY
Center for Software Security and Assurance

4. Result Step

You can review a summary of the overall steps.



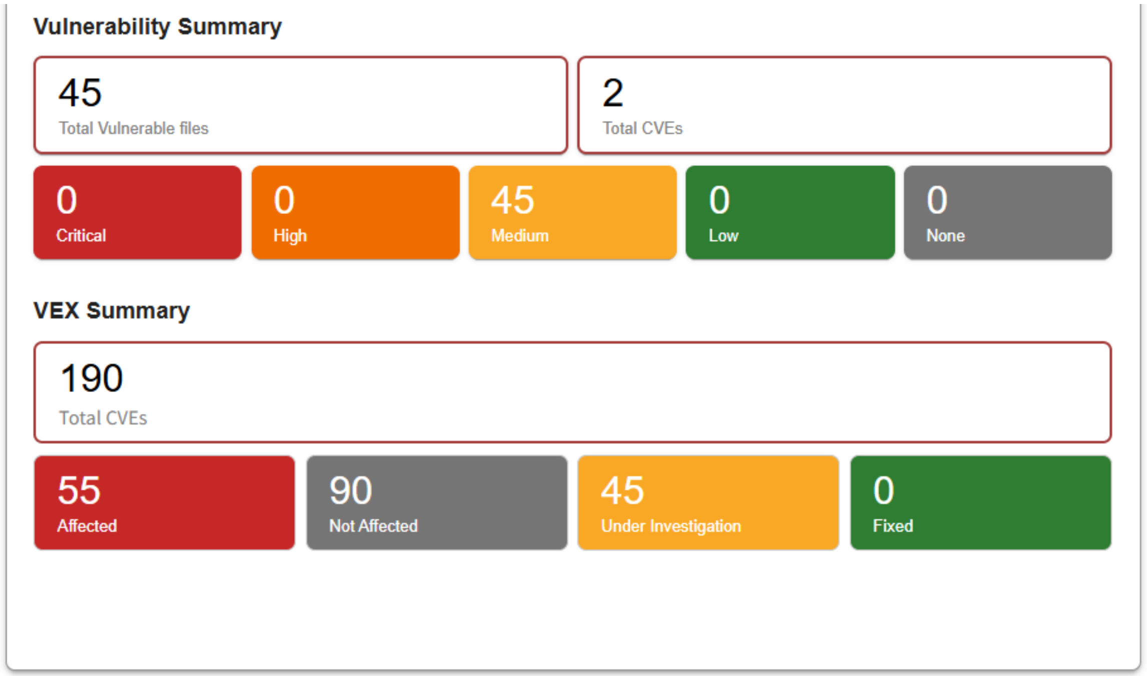
Result

This dashboard provides a comprehensive summary of your project’s SBOM, vulnerabilities, and VEX results.
Easily track your project's security status and identify key areas for improvement.
All the results are presented in an organized, user-friendly format for quick analysis and decision-making.

File Name: binutils-2.34
Tool Vendor: CSSA Korea Univ.
Product Vendor: Test Product Vendor
Author: Test Author

SBOM Summary

14	OSS Component
1624	Files
14	Dependencies
hidx File	Input Format



Vulnerability Summary

45	Total Vulnerable files
2	Total CVEs
0	Critical
0	High
45	Medium
0	Low
0	None

VEX Summary

190	Total CVEs
55	Affected
90	Not Affected
45	Under Investigation
0	Fixed