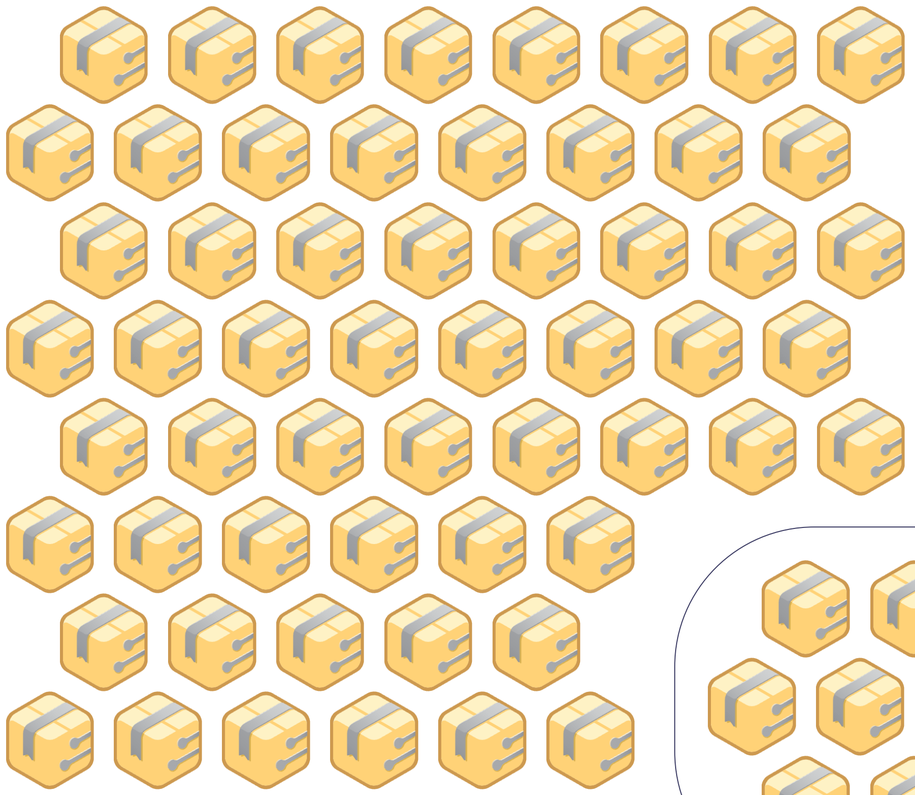
OSBC

FOSSID

APRIL 2023

Types of SCA

Types of SCA Security VS License Compliance



License Compliance Risks

There are over 150M components of software in FossID's Knowledge Base today.

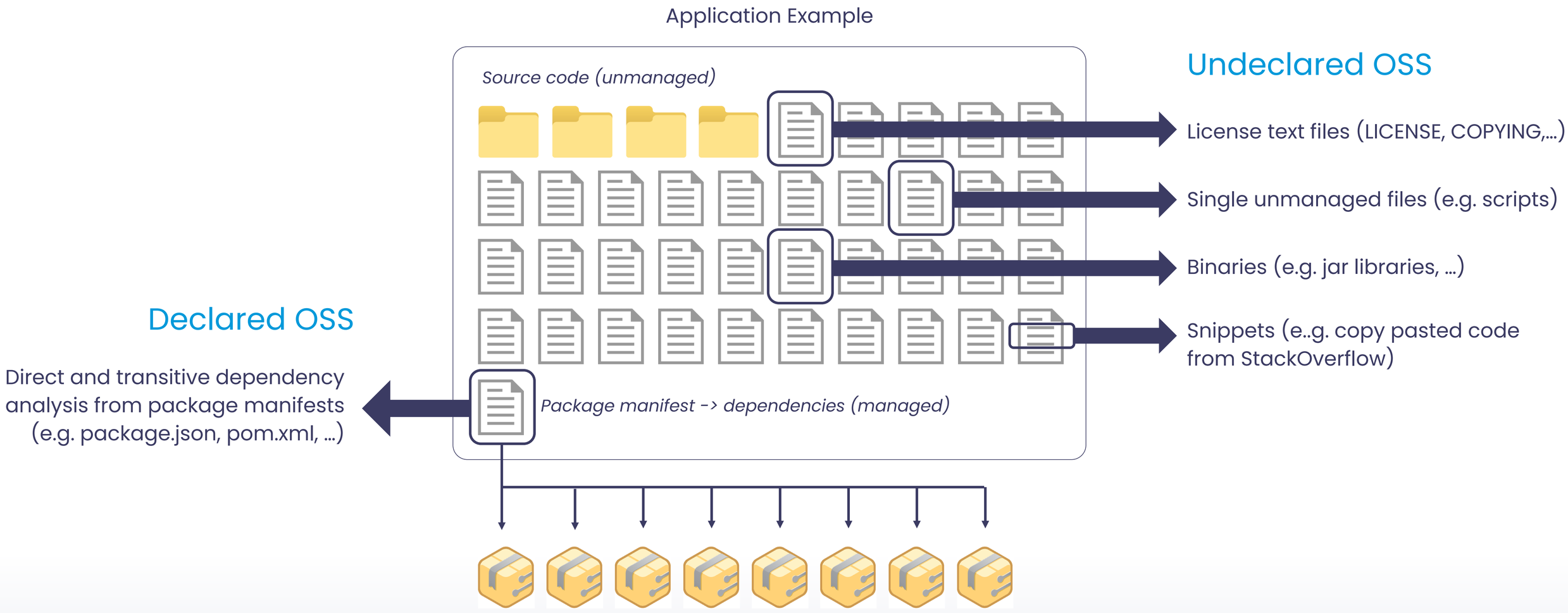
These software components are released under 2000+ different open source licenses.

Security Risks

Only about 1M components have known vulnerabilities and exposures (CVEs)

FOSSID

Types of SCA Declared VS Undeclared OSS



FOSSID

Open Source is everywhere As is the need to properly manage it

97%

of codebases leverage Free Open Source Software

81%

of codebases have security vulnerability issues

53%

of codebases have license compliance issues

Source: 2022 Open Source Security & Risk Analysis Report (OSSRA)

FOSSID

FossID Key Strength – Detection Capabilities

With our tools, you should find more (compared to competition)

- Ability to **identify more** software
- Ability to identify in **higher level of detail**
- Ability to identify with **less false positives**

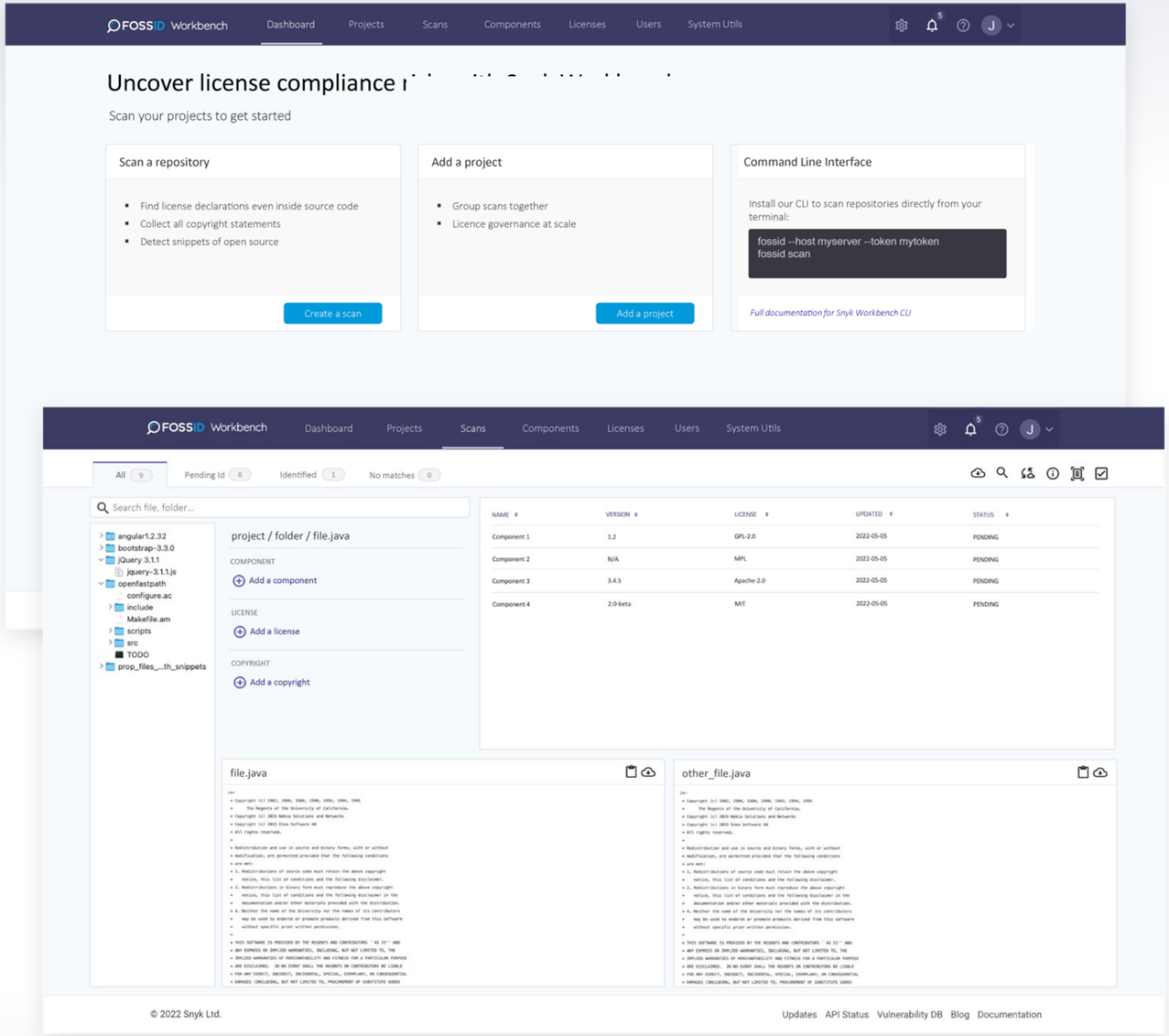
FOSSID

PRODUCTS

Workbench Platform

One interface to manage all 3rd party sw:

- Review scan results:
  - See OSS component, file & snippet matches
  - Review license and copyright information
- Generate reports
  - SBOM (SPDX),
  - Notice files, etc…
- Policy management
  - Implement and enforce company wide or project based OSS policies
  - Approve/reject OSS based on policies
  - Follow up through JIRA
- Catalog all 3rd party software used in your products
- Create customized user roles based on privileges



FOSSID

Knowledge Base OSS Intelligence

Comprehensive Open-Source Software (OSS) coverage

FossID products and services are powered by FossID's industry-leading OSS intelligence database.

Best Coverage

Maintained by a dedicated research team, it covers over 150M OSS components coming from 60+ public sources and user contribution sites such as GitHub and StackOverflow.

Maximum Accuracy

FossID's scanning technology detects from whole OSS components and binaries to small snippets of code (as little as 6 lines of code).

Highest Confidentiality

FossID never accesses or transfers your source code, scans cryptographic hashes instead. Alternatively, FossID allows for fully on-prem air-gapped deployments.

FOSSID

Knowledge Base Detection Capabilities

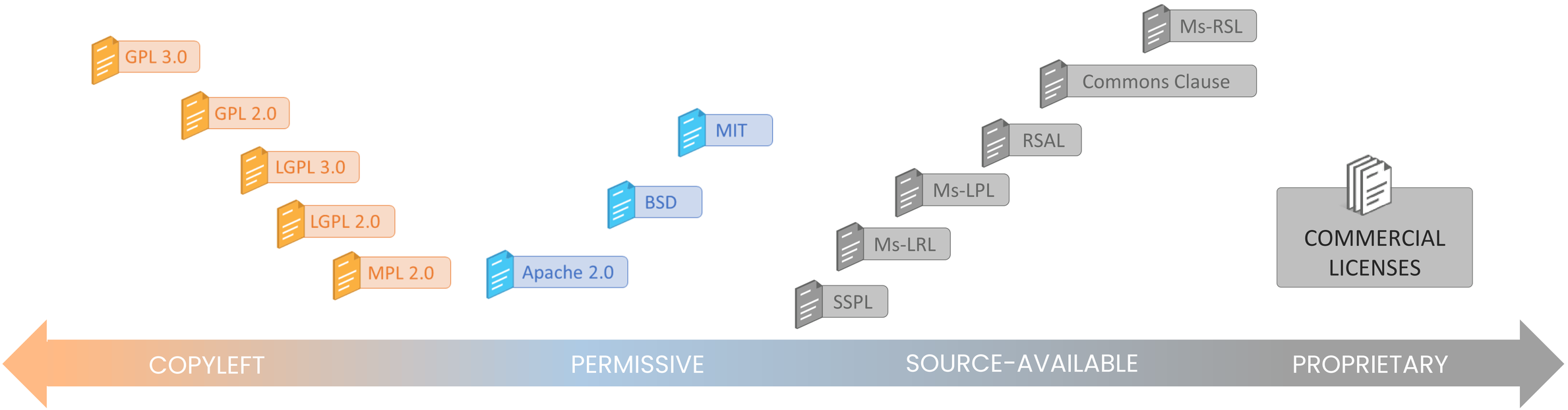
Quickly identify folders, libraries, archives or binaries

Detect full files in your code base even if they are modified

Identify smaller open source footprints like copy pasted code

FOSSID

Types of SCA Source-available licenses



FOSSID

Automation Package

Integrate FossID Workbench in your SDLC:

- Command Line Interface
  - BareMetal scanning functionality
  - Linux, Windows and Mac OS
  - Can be used as stand alone (independently from Workbench)
- API
  - Access all Workbench functionality
  - Upload target code, trigger scans, generate reports, etc…

```
MacBook-Pro-6:CLI fossi$ fossid-cli --config fossid.cfg EXAMPLE_PROJECT_WebApp/prop_files_with_snippets/sample_copy.c --pretty --fields minimal
{
    "component": {
        "artifact": "kernel",
        "author": "linux",
        "license": "GPL-2.0-only",
        "url": "https://www.kernel.org/pub/linux/kernel/v3.x/linux-3.8.6.tar.gz",
        "version": "3.8.6"
    },
    "local_path": "EXAMPLE_PROJECT_WebApp/prop_files_with_snippets/sample_copy.c"
}
{
    "component": {
        "artifact": "kernel",
        "author": "kernel",
        "license": "GPL-2.0-only",
        "url": "https://cdn.kernel.org/pub/linux/kernel/v3.x/linux-3.8.6.tar.bz2",
        "version": "3.8.6"
    },
    "local_path": "EXAMPLE_PROJECT_WebApp/prop_files_with_snippets/sample_copy.c"
}
{
```

```
        inode = mqueue_get_inode(sb, ns, S_IFDIR | S_ISVTX | S_IRWXUGO, NULL);
        if (IS_ERR(inode))
                return PTR_ERR(inode);

        sb->s_root = d_make_root(inode);
        if (!sb->s_root)
                return -ENOMEM;
        return 0;
}

==>static struct dentry *mqueue_mount(struct file_system_type *fs_type,
==>                     int flags, const char *dev_name,
==>                     void *data)
==>{
==>     if (!(flags & MS_KERNMOUNT)) {
==>             struct ipc_namespace *ns = current->nsproxy->ipc_ns;
==>             /* Don't allow mounting unless the caller has CAP_SYS_ADMIN
==>              * over the ipc namespace.
==>              */
==>             if (!ns_capable(ns->user_ns, CAP_SYS_ADMIN))
==>                     return ERR_PTR(-EPERM);
==>
==>             data = ns;
==>     }
==>     return mount_ns(fs_type, flags, data, mqueue_fill_super);
}

static void init_once(void *foo)
{
        struct mqueue_inode_info *p = (struct mqueue_inode_info *) foo;
```

FOSSID

VulnSnippet Finder Package

Snippet detection for vulnerable OSS snippets

FossID's Knowledge Base snippet detection capabilities have been extended to include special detection of vulnerable OSS snippets.

Extensive Coverage

Over 200k vulnerable snippets curated by experts.

Risks in Undeclared OSS

Knowing the exact lines of code that introduce vulnerabilities is crucial when dealing with undeclared OSS.

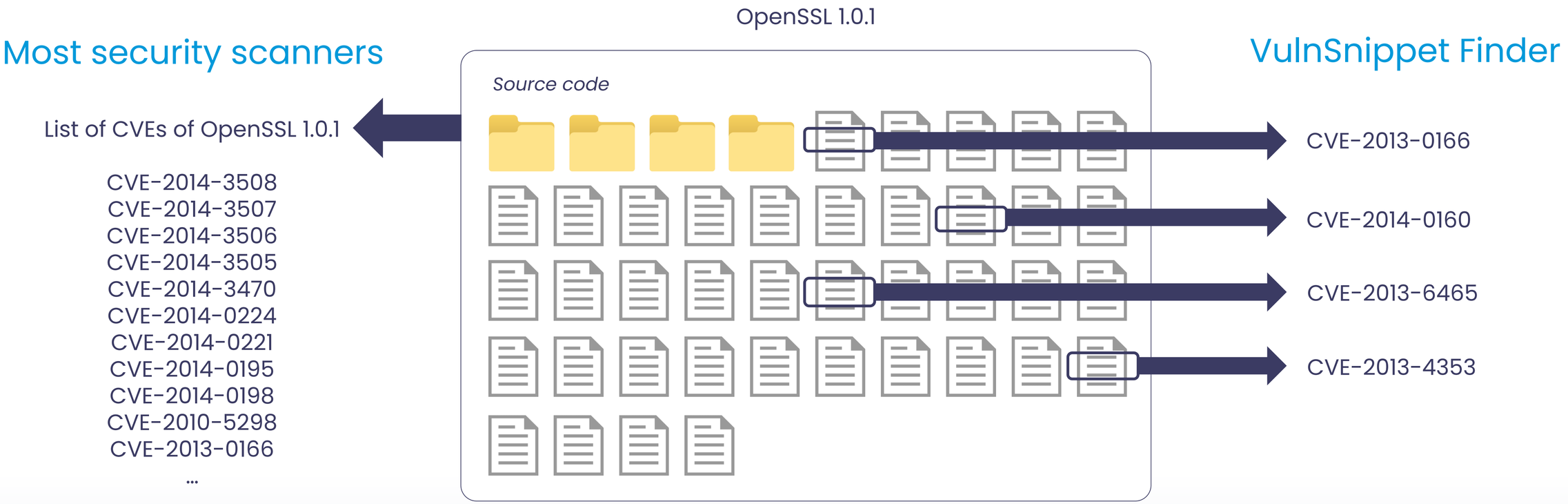
Designed for automation

Only trigger build fails for vulnerabilities you know you have in your source code.

FOSSID

VulnSnippet Finder Package

Most security scanners assume OSS vulnerabilities based on component/version while VulnSnippet Finder searches for the exact lines of code/snippets.



FOSSID

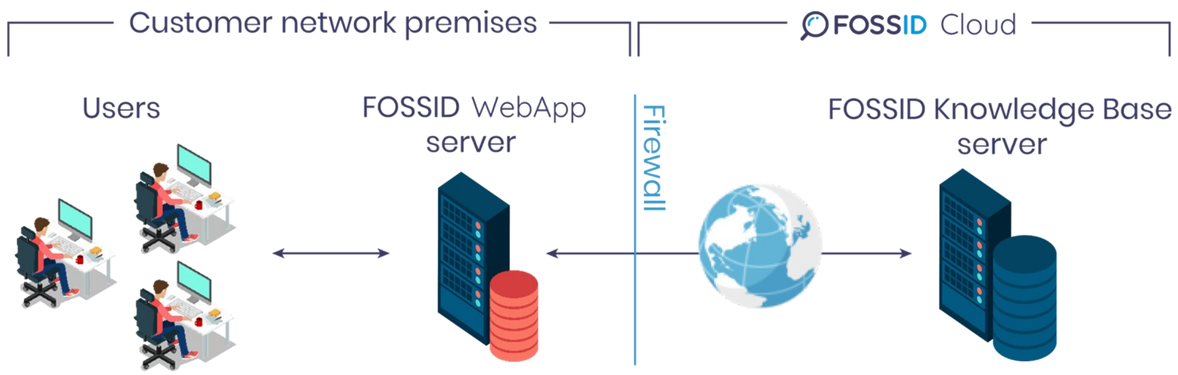
DEPLOYMENT OPTIONS

Deployment Options Hybrid & On-prem

Hybrid Deployment

- Knowledge Base in the cloud
- No source code is ever transmitted

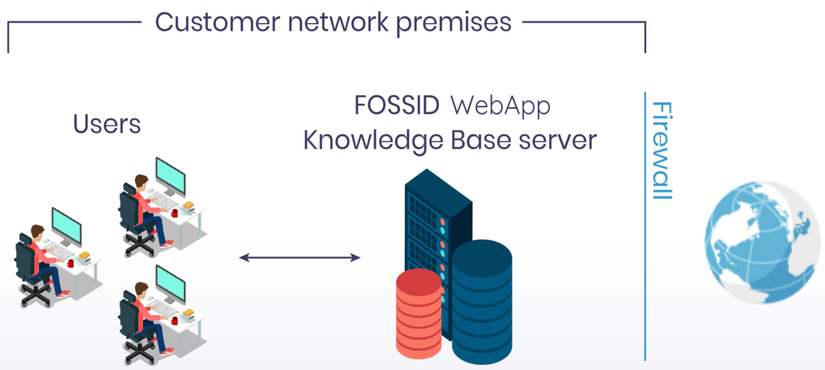
  Only digital signatures of source code are used to query the FOSSID Knowledge Base.
- Continuous updates



On-prem Deployment

- Knowledge Base locally deployed
- No external network traffic involved

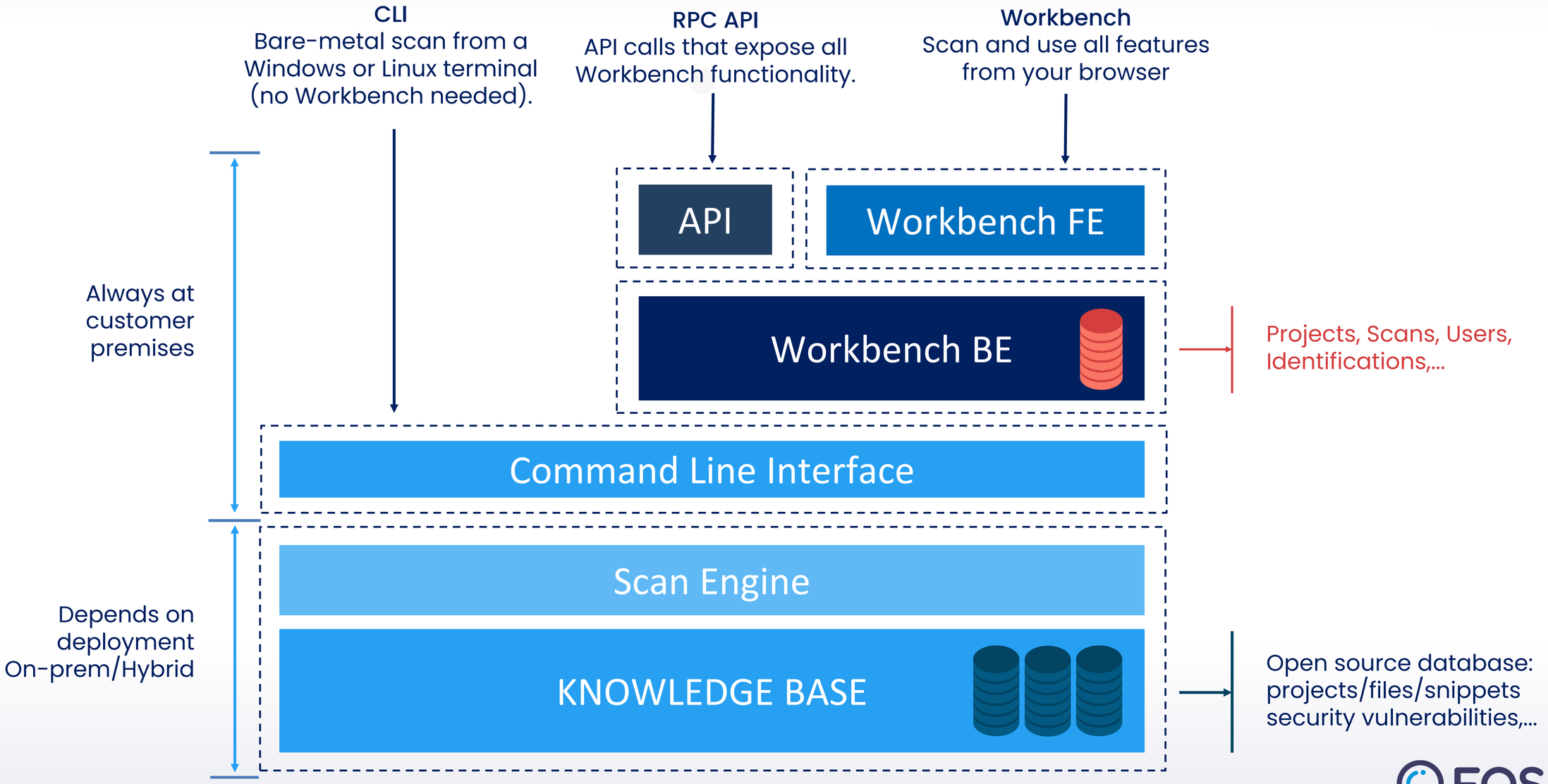
  Performing scans does not involve any network traffic outside your network premises.
- Monthly and weekly updates



FOSSID

ARCHITECTURE

FossID Technology Architectural View



FOSSID

OSBC

Thanks

snyk

Develop fast.

Stay secure.

snyk

Developer Security Platform

Snyk Code

Snyk Open Source

Snyk Container

Snyk IaC

Snyk Cloud

Developer Experience

Snyk Application intelligence

Snyk Learn Security Education

Snyk Intel Security intelligence

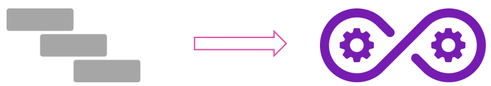
Empowerment

Extensibility

Governance

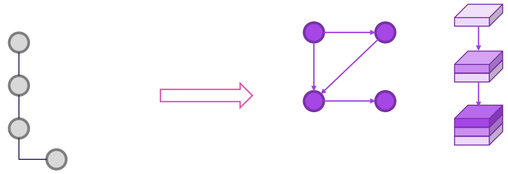
Development Has Changed

More development, faster



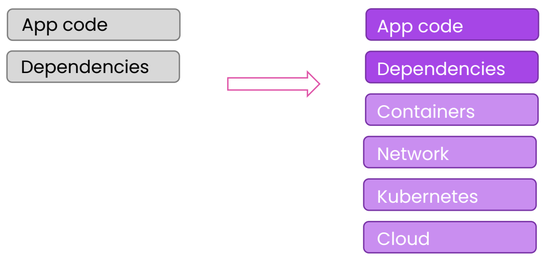
Need developers who understand and love to create secure applications

Software supply chain more complex



Require guidance and visibility into the entire software supply chain

The cloud is part of the code



Need to understand full scope of apps from code to cloud, and back to code

A Modern Approach to Security is Required

Traditional App Sec

Testing after development

Audit Based

Code and Infrastructure Secured Independently

Dev-First

Continuous Testing

Fix Based

Holistic Cloud Native Application Context

snyk

Snyk empowers secure developers

All your developers

App devs

Platform devs

Throughout their code

- Code
- Open Source Dependencies
- Containers
- Infrastructure as Code

In their natural workflow

Store code in SCM → Develop features & fixes → PR code review → Automated build & test → Store artifacts → Deploy & Run → Feedback & improve

Security Education

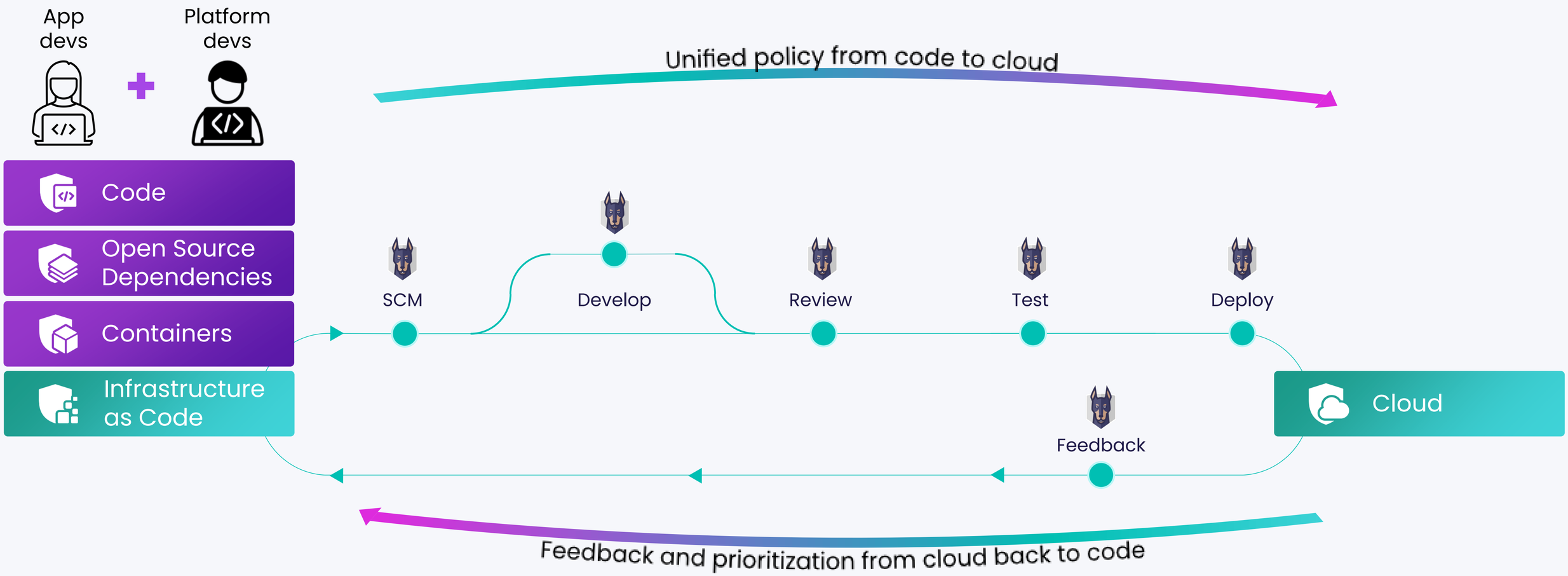
Snyk Provides Visibility and Developer Guidance In Your Software Supply Chain

Security policies to automate security at scale

- Open Source Dependencies
- Containers

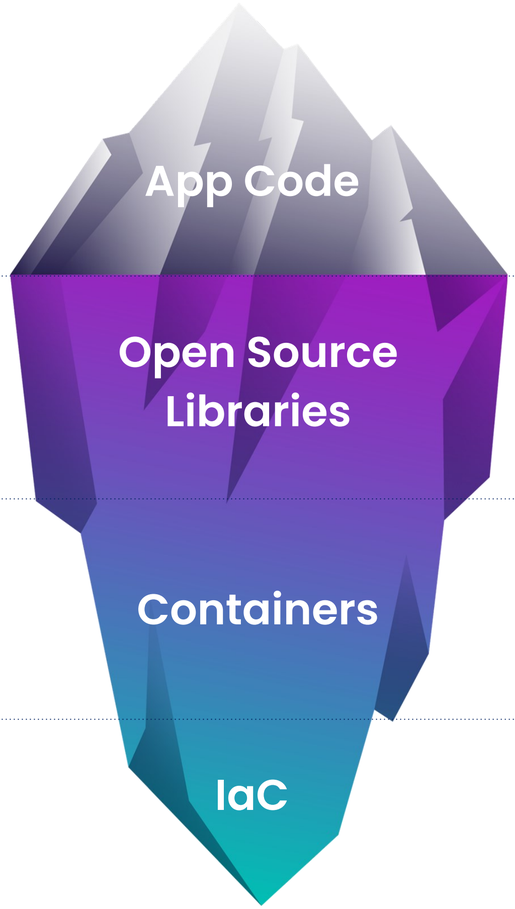
Choose safe packages and containers → Identify and track all dependencies → Fix vulnerabilities → Govern releases → Test continuously

Snyk empowers secure cloud developers



Snyk 소개 | 개요

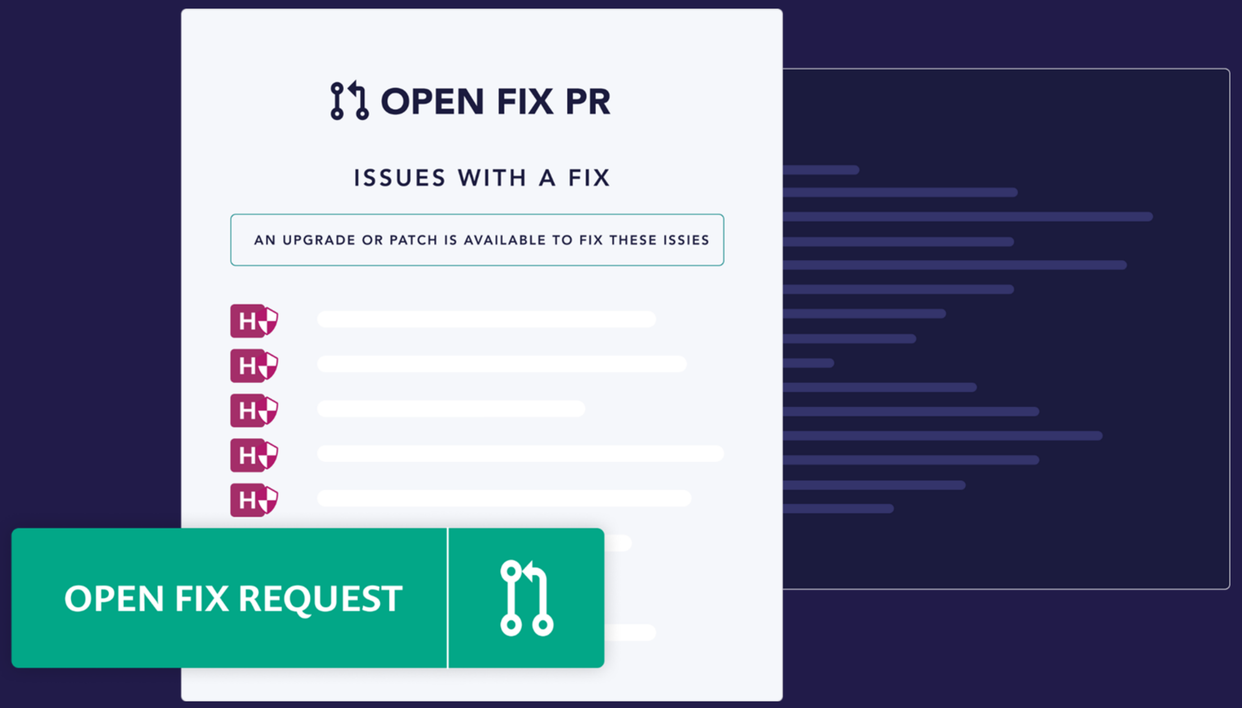
Snyk은 DevSecOps 환경에서 **개발 프로세스 지연 없이 지속적으로** 오픈소스의 보안취약점 및 라이선스 이슈를 식별, 모니터링 할 수 있도록 지원하는 **개발자 친화적인** 차세대 관리도구입니다.



Code	매일 발생하는 배포에 대응하기 위해 빠른 속도로 코드를 스캔하여 취약점 이슈를 확인할 수 있습니다.
Open Source	프로젝트에 포함된 오픈소스와 디펜던시를 빠르게 식별할 수 있으며 보안취약점의 80%가 발견되는 간접(Indirect) 디펜던시도 손쉽게 확인하여 모니터링 할 수 있습니다.
Containers	컨테이너 이미지의 가시성을 확보하여 포함된 소프트웨어 및 그 취약점을 빠르게 식별하고 수정할 수 있게 합니다.
Infrastructure as Code	네트워크 액세스, 스토리지, 서버를 코드 형식으로 관리하는 IaC 환경에서 잘못된 구성 설정으로 인해 발생하는 취약점을 빠르게 식별하고 수정할 수 있게 합니다.

Automated Remediation
Solving the complex fix logic

Single-click fix pull request



snyk

Be license compliant as early as coding

Start early

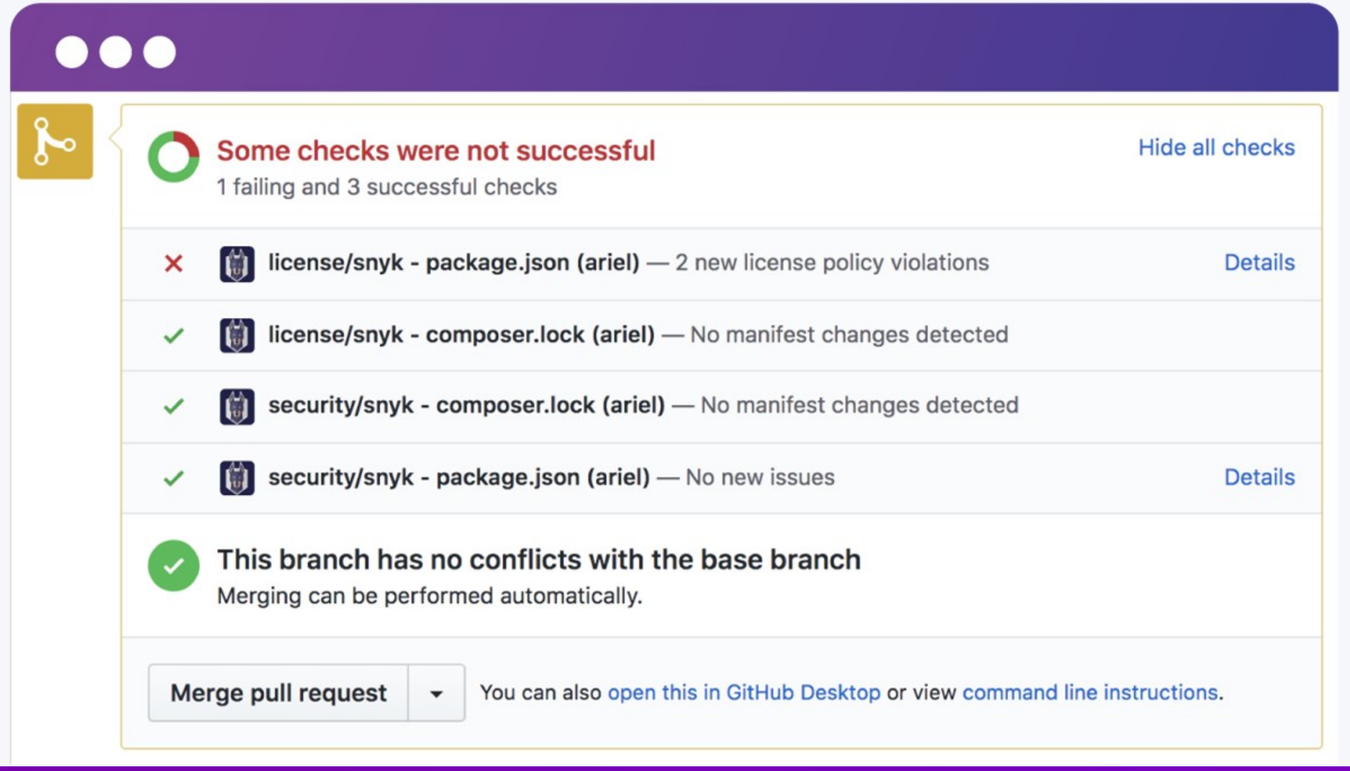
Verify compliance at every stage of development

Scan

Get visibility to all the licenses that are being used.

Comply

Define policies and take automatic actions to verify compliance.



Copyright info

BOM report

Policies

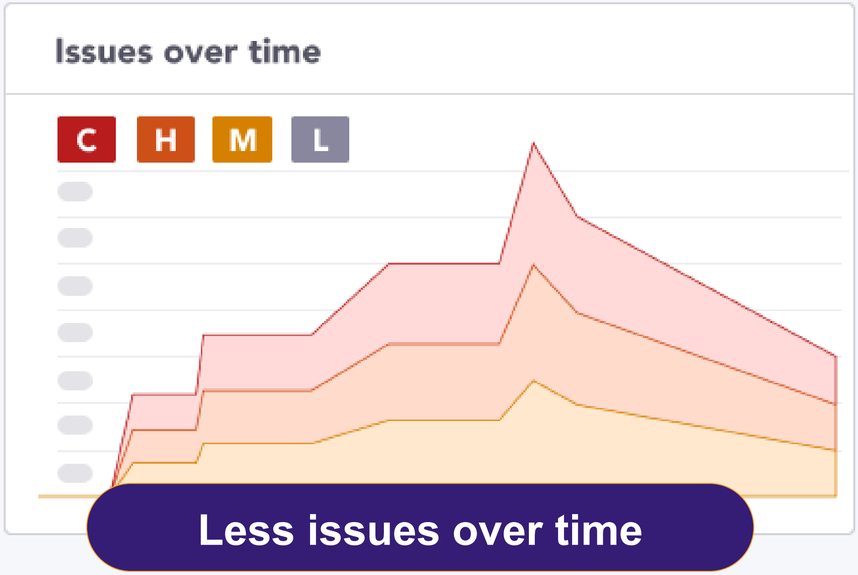
Gating non-compliant packages

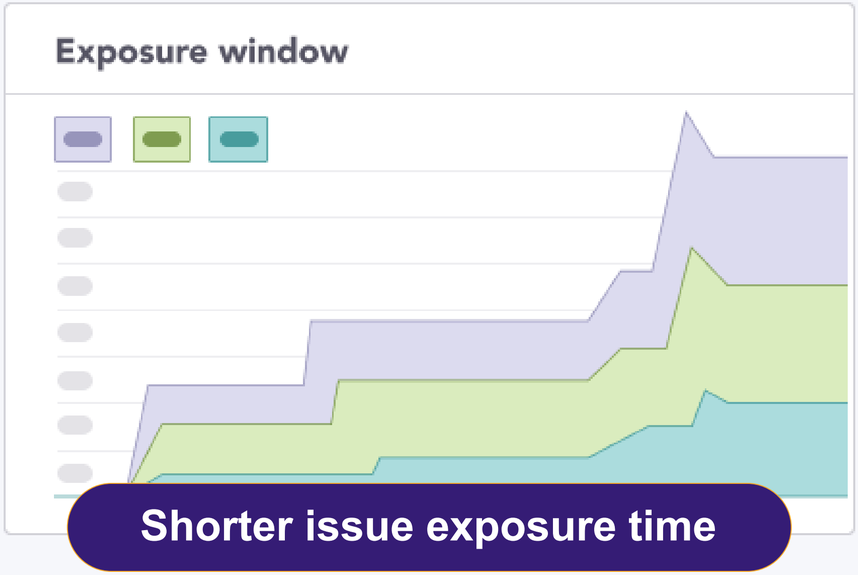
Legal team actionable instructions

snyk

Automated Remediation

Fix MORE issues, QUICKER





Accelerating MTTF (mean time to fix)

snyk

Snyk 소개 | 사용 프로세스

Snyk은 저장소와 연동된 Snyk 웹UI를 통해 저장소/프로젝트를 관리하고 실시간으로 취약점을 조회하여 자동 수정하는 간편한 프로세스를 제공합니다.

Snyk 사용 프로세스

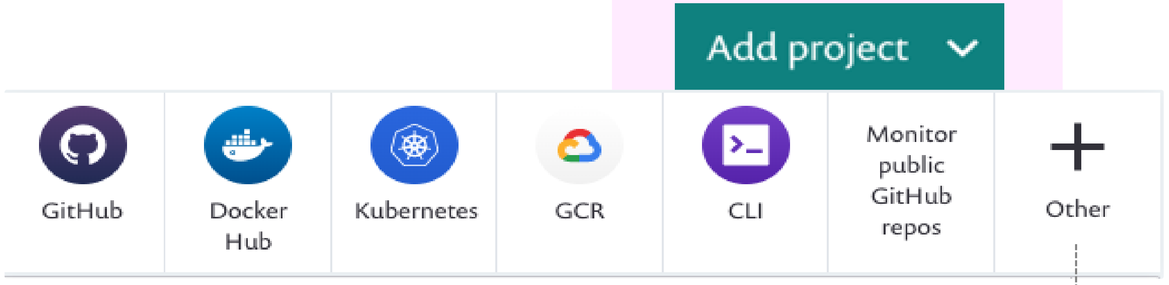
소스코드 저장소 연동 → **프로젝트 추가** → **취약점 조회** → **취약점 수정**

GitHub, GitLab, BitBucket Cloud, Azure Repos 등 연동

대시보드에서 저장소/프로젝트 선택하여 Snyk 프로젝트 생성 (또는 CLI 사용 가능)

추가된 프로젝트의 보안취약점/라이선스 취약점 조회

Pull Request 생성하여 취약점 자동 수정



GitHub, GitLab, Bitbucket, Azure Repos 등 주요 소스코드 관리도구(SCM)와의 연동을 통해 자동으로 취약점을 모니터링할 수 있습니다.

- Git 레파지토리를 Snyk으로 빠르게 추가하여 관리
- 어플리케이션과 함께 IaC를 통합하여 관리
- 로컬에서 파일을 테스트하거나 CLI에서 명령어를 사용하여 테스트 가능
- 강력한 Snyk API를 활용하여 프로젝트 가져오기 자동화 가능

Snyk 주요 기능 | 대시보드

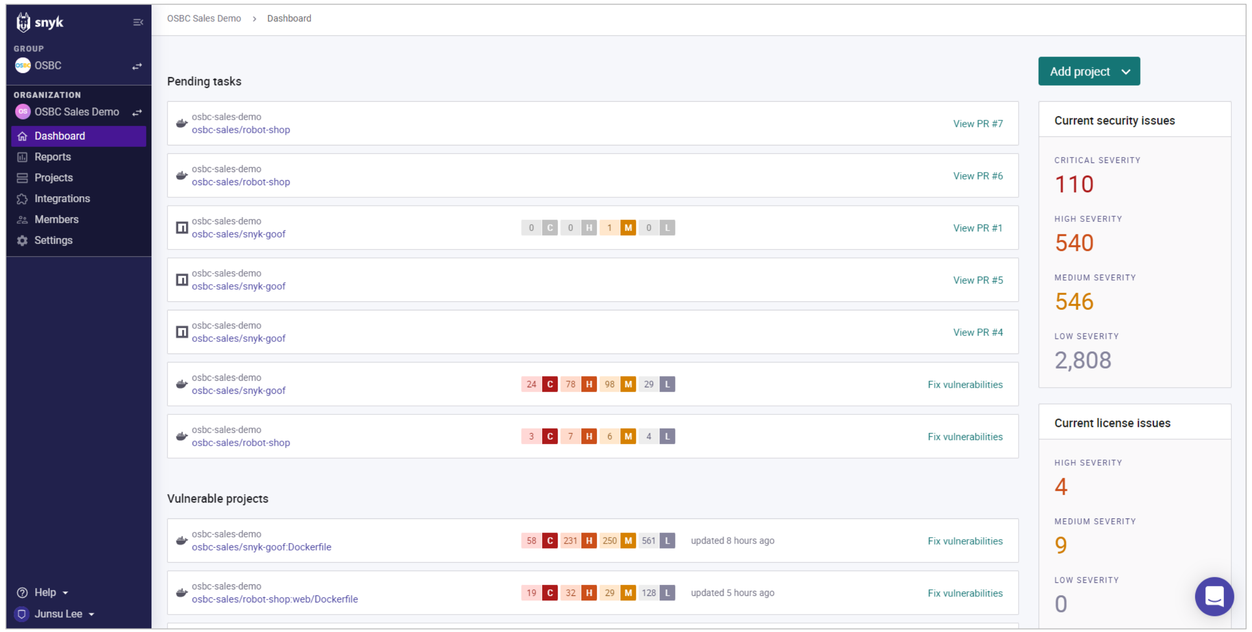
Snyk 대시보드에서는 사용자가 관리하고 있는 프로젝트 및 취약한 프로젝트의 현황을 직관적으로 표시해줍니다.

전반적인 보안취약점 상태

시간 경과에 따른 리스크, 노출 현황, 보안을 위한 조치 현황 모니터링

디펜던시 상태

전반적인 상태를 추적, 이슈 컴포넌트를 식별하고 조치 이행



실행가능성

솔루션이 존재하는 이슈들을 확인하고 자동으로 필터링

정책

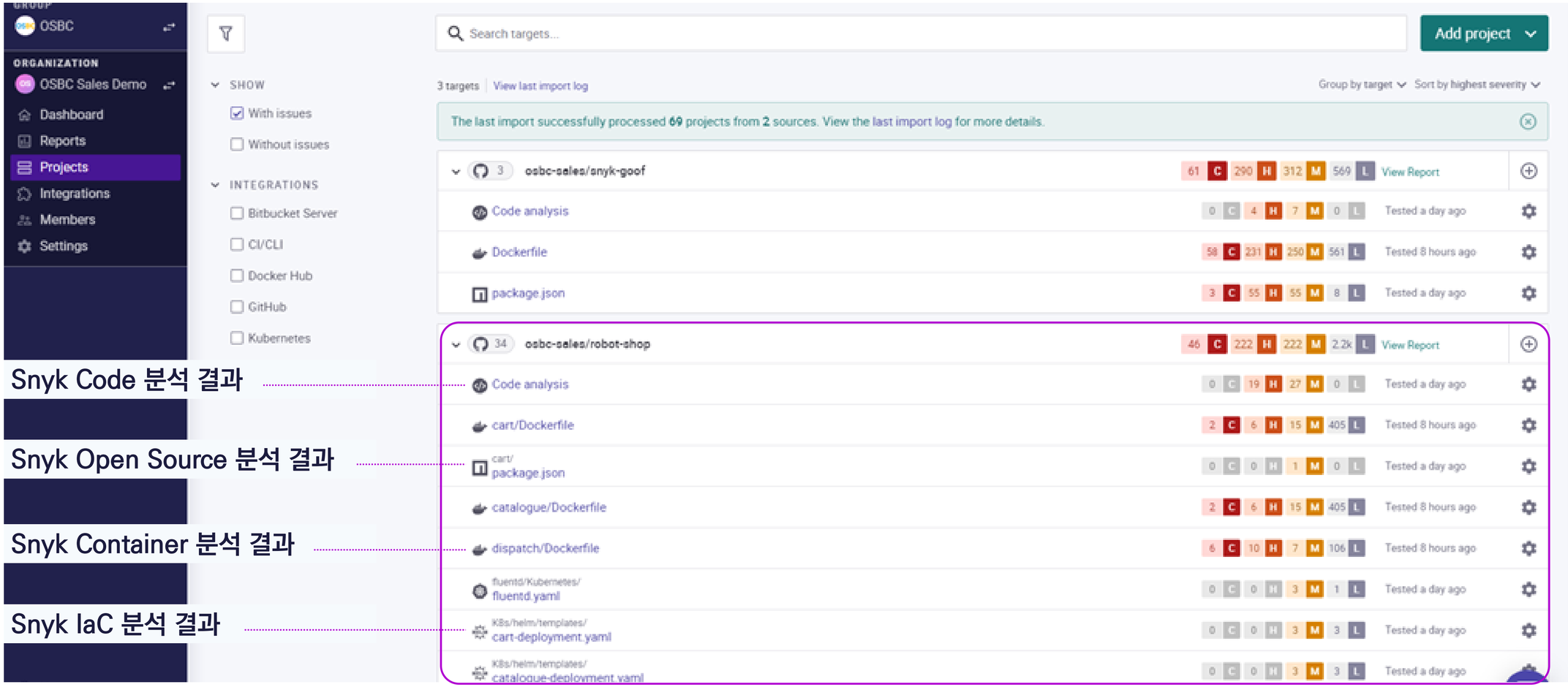
자동화 되어있고 사용자 정의 설정이 가능한 규칙을 사용하여 보안 및 라이선스 규정 준수 관리

프로젝트 관리

다수의 프로젝트에서 확인되는 보안취약점 및 라이선스 이슈를 효율적으로 관리 가능

Snyk 주요 기능 | 프로젝트 현황 관리

Snyk의 프로젝트 현황 관리 화면에서 각 프로젝트별 Target 및 위험도 현황의 개요가 표시됩니다.
사용자는 각 프로젝트에 접근하여 이슈 확인, 수정, 세부 정보 확인 등의 세부 동작을 수행합니다.



제품 종류에 따라 프로젝트 아이콘, 분석 결과 출처 표기

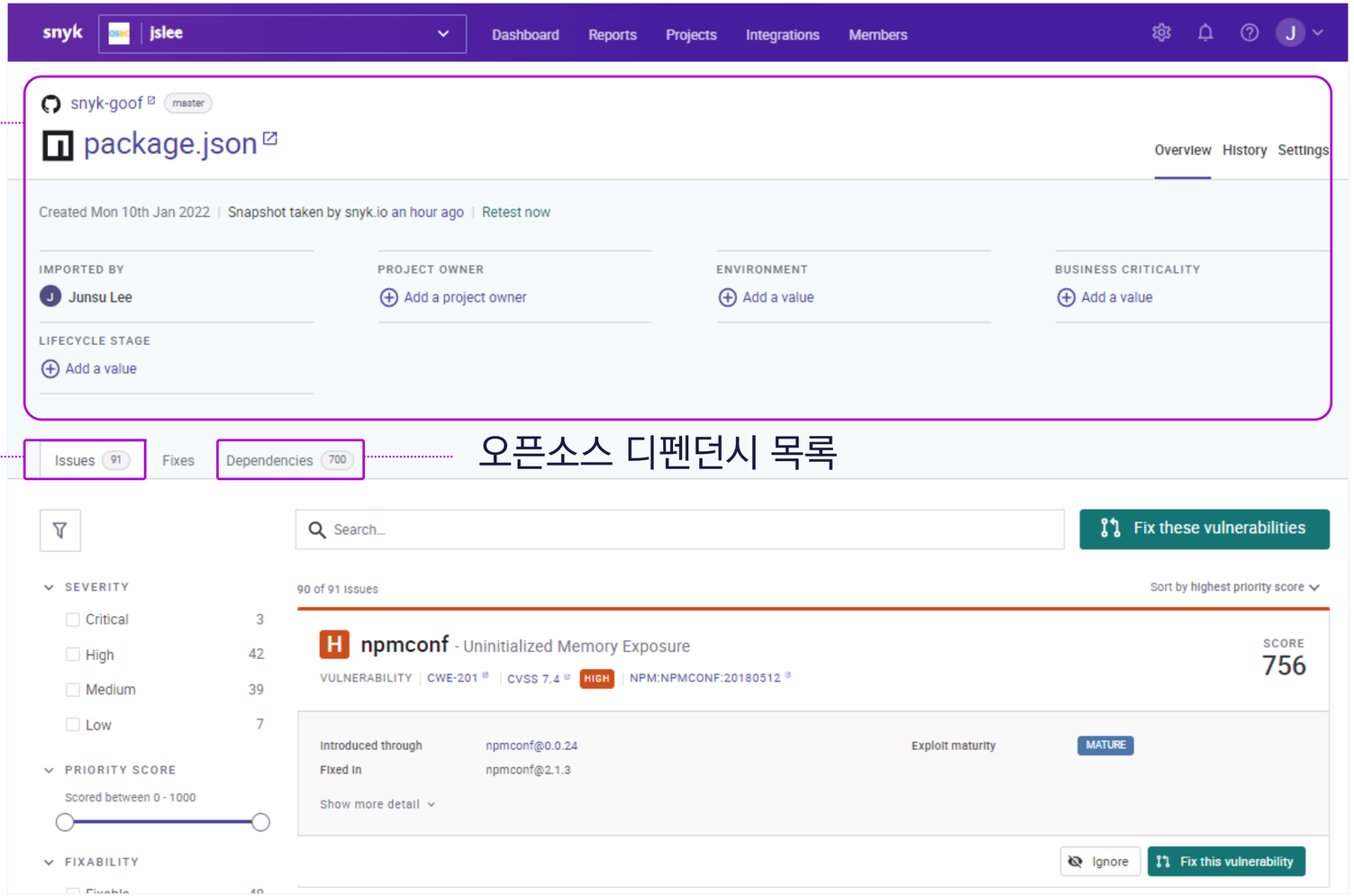
Snyk 주요 기능 | 프로젝트 세부

Snyk의 프로젝트 현황 관리 화면에서 각 프로젝트별 Target 및 위험도 현황의 개요가 표시됩니다.
사용자는 각 프로젝트에 접근하여 이슈 확인, 수정, 세부 정보 확인 등의 세부 동작을 수행합니다.

프로젝트 개요

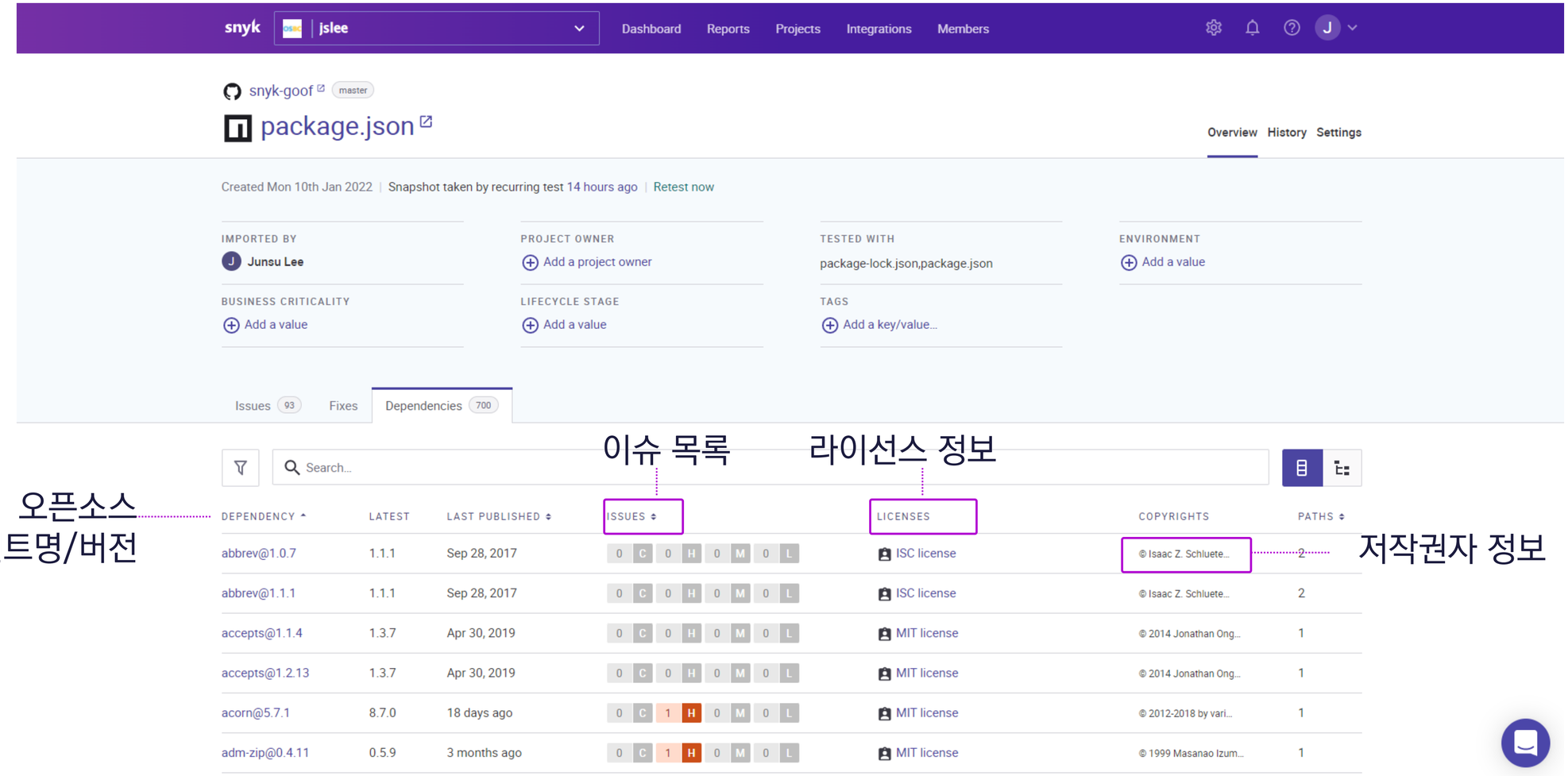
이슈 목록

오픈소스 디펜던시 목록



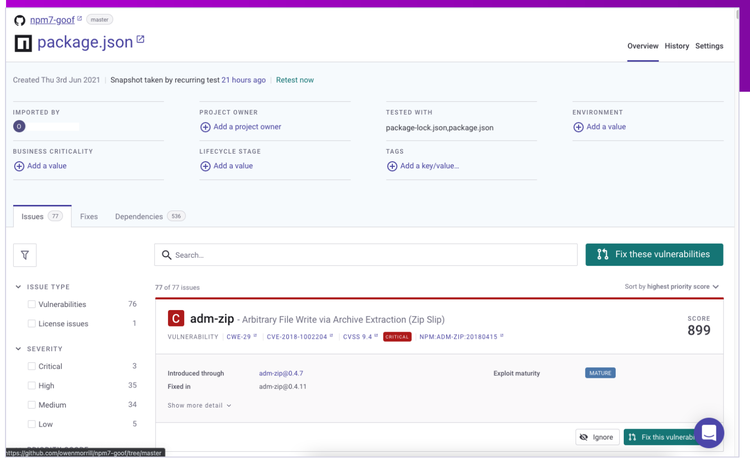
Snyk 주요 기능 | 프로젝트 세부 – Dependencies 결과화면

Dependencies 결과 화면에서 오픈소스 사용 내역(컴포넌트명, 버전 및 라이선스) 정보를 확인할 수 있습니다.
포스아이디 대비 최신버전에 대한 정보와 출시일 정보를 추가로 확인 가능합니다.

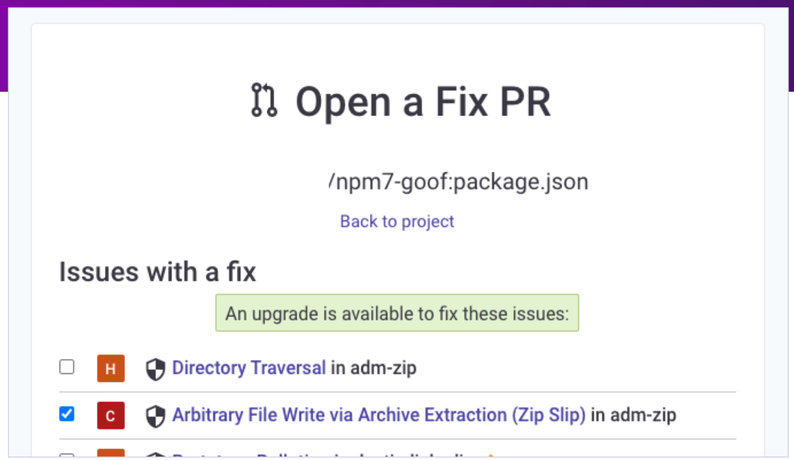


Snyk 주요 기능 | 프로젝트 세부 – Pull Request 생성을 통한 자동 조치

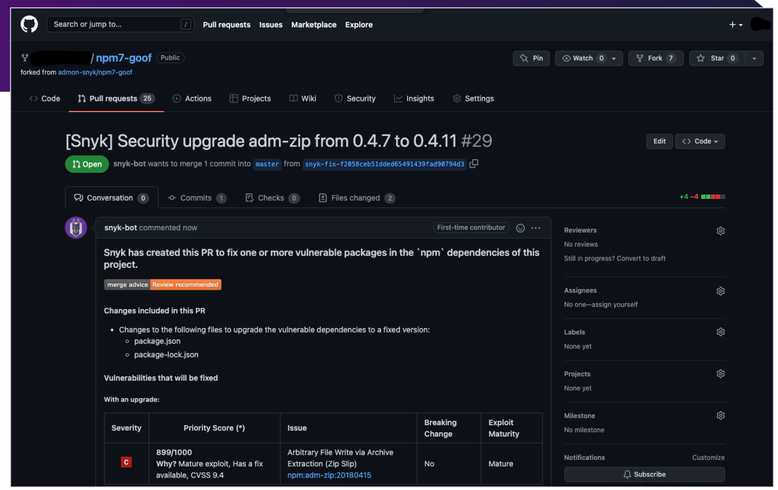
Snyk Open Source의 Auto Fix 기능을 이용해 취약한 버전의 오픈소스를 클릭 한 번에 자동 패치하여 관리할 수 있습니다.



이슈 목록에서 Fix 버튼 클릭



Fix PR(Pull Request) 생성



Approve & Merge하여 반영

오픈소스 검증 및 취약점 코드 분류 → 보안팀 확인 / 문의 → 조치 방안 확인 → 조치

자동 조치 시 생략되는 단계 (개발팀 자체 점검/조치 수행)

반복 수행

Snyk IDE 플러그인 | Snyk for Visual Studio Code 예시

SNYK

OPEN SOURCE SECURITY
- Snyk found 86 vulnerabilities
- Analysis took 6.19s, finished at 03:28 PM, 10/12/21
- package-lock.json goof - 86 vulnerabilities
  - adm-zip@0.4.7 - Arbitrary File Write via Archiv...
  - handlebars@4.0.14 - Prototype Pollution
  - kerberos@0.0.24 - DLL Injection
  - lodash@4.17.4 - Prototype Pollution
  - adm-zip@0.4.7 - Directory Traversal
  - ajv@6.10.2 - Prototype Pollution
  - ansi-regex@3.0.0 - Regular Expression Denial o...
  - bl@2.2.0 - Remote Memory Exposure

CODE SECURITY
- Snyk found 21 vulnerabilities
- Analysis took 2.96s, finished at 03:28 PM, 10/12/21
- index.js routes - 11 vulnerabilities
  - Unsanitized input from the HTTP request body ...
  - Unsanitized input from the HTTP request body ...
  - Unsanitized input from the HTTP request body ...
  - This endpoint handler performs a file system o...
  - This endpoint handler performs a file system o...
  - This endpoint handler performs a file system o...
  - This endpoint handler performs a file system o...
  - This endpoint handler performs a file system o...

CODE QUALITY
- Snyk found 4 issues
- Analysis took 2.96s, finished at 03:28 PM, 10/12/21
- index.js routes - 3 issues
- utils.js goof - 1 issue

HELP & FEEDBACK
- Help Snyk to make a better extension
- Send us feedback or report a bug
- Top 3 FAQ
  - 1. How to get the most out of Snyk's extension?
  - 2. How to ignore files and directories?
    - Add default .dcignore file to your workspace
    - Add a custom .dcignore file to your workspace

index.js 9+

routes > index.js > create > create

```
136     var time = t.slice(reminder + remindToken.length);
137     time = time.replace(/\n$/, '');
138
139     var period = hms(time);
140
141     console.log('period: ' + period);
142
143     // remove it
144     t = t.slice(0, reminder);
145     if (typeof period != 'undefined') {
146       t += ' [' + ms(period) + ']';
147     }
148   }
149   return t;
150 }
151
152 exports.create = function (req, res, next) {
153   // console.log('req.body: ' + JSON.stringify(req.body));
154
155   var item = req.body.content;
156   var imgRegex = /\!\[alt text\]\((http.*)\s\".*/;
157   if (typeof (item) == 'string' && item.match(imgRegex)) {
158     var url = item.match(imgRegex)[1];
159     console.log('found img: ' + url);
160
161     exec('identify ' + url, function (err, stdout, stderr) {
162       console.log(err);
163       if (err !== null) {
164         console.log('Error (' + err + '):' + stderr);
165       }
166     });
167
168   } else {
169     item = parse(item);
170   }
171
172   new Todo({
173     content: item,
174     updated_at: Date.now(),
175   }).save(function (err, todo, count) {
176     if (err) return next(err);
177
178     /*
179     res.setHeader('Data', todo.content.toString('base64'));
180     res.redirect('/');
181     */
182
183     res.setHeader('Location', '/');
184     res.status(302).send(todo.content.toString('base64'));
185
186     // res.redirect('/#' + todo.content.toString('base64'));
```

Snyk Code Vulnerability

High

Unsanitized input from **the HTTP request body [:155] flows [:155, :155, :157, :157, :158, :158, :158, :158, :159, :161, :161]** into **child_process.exec [:161]**, where it is used to build a shell command. This may result in a Command Injection vulnerability.

This vulnerability happens on line 161 | More info

Security | Maintenance | Synclet | Spawn | Command

This vulnerability was fixed by 54 projects. Here are 3 example fixes.

chaitin/passionfruit | Example 1/3

```
})
.post('/spawn', async ctx => {
  let pid = await state.device.spawn([ctx.request.body.bundle])
  let { device, bundle } = ctx.request.body
  let dev = await FridaUtil.getDevice(ctx.params.device)
  let pid = await dev.spawn([ctx.request.body.bundle])
  // todo: attach
  ctx.body = { status: 'ok'}
```

Support multiple synclets for a provider.
Handle errors in spawn
Refactoring log command.

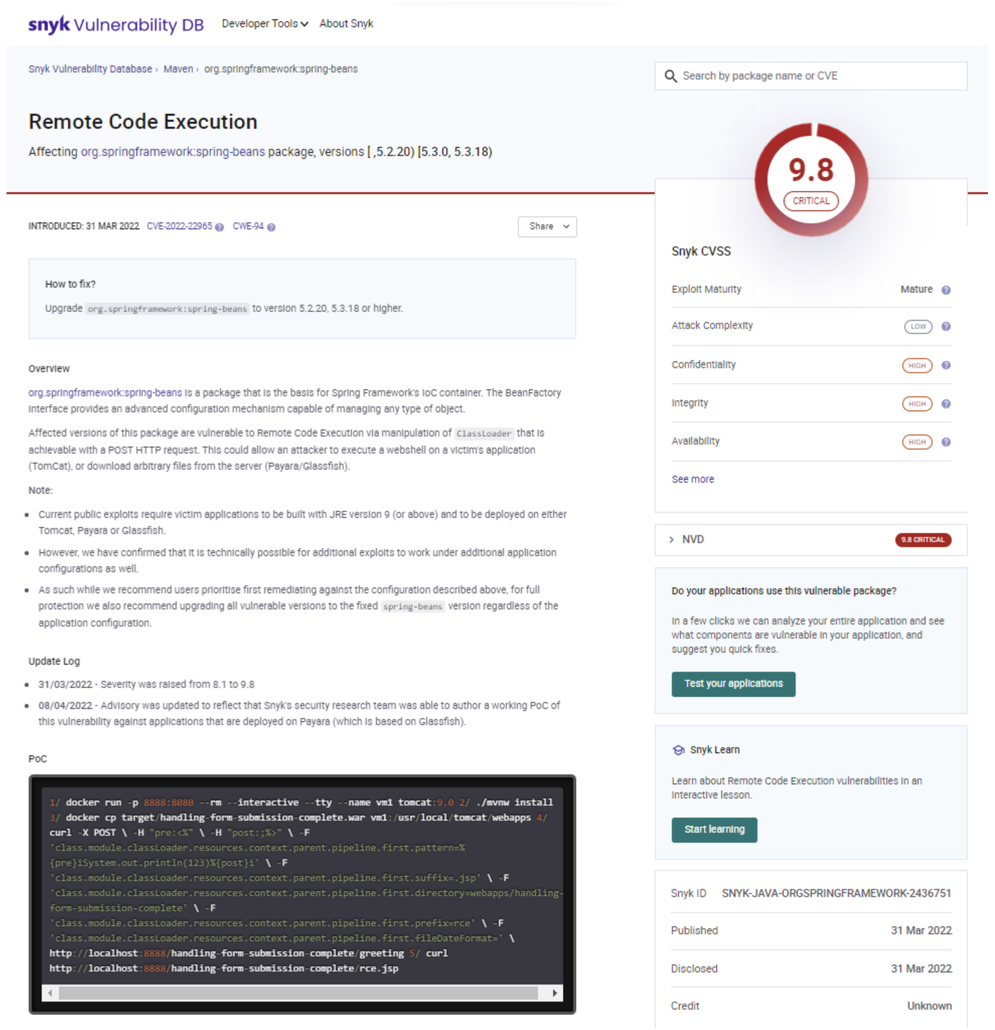
Do you want to hide this suggestion from the results?

Ignore on line 161 | Ignore in this file

master | ⊗ 5 ⚠ 17 ⓘ 3 | Snyk Vulnerability Scanner

Snyk 부가 지원 기능 | Snyk Vulnerability DB

Snyk Vulnerability DB는 가장 포괄적이고, 정확하며 시기적절한 정보를 제공하는 오픈소스 취약점 정보 DB입니다.



다양한 패키지매니저 기반 정보 제공

npm, Maven, Nuget, pip, cocoapods 등 패키지매니저와 Linux 배포판별 오픈소스 및 비정형 C/C++ 기반 오픈소스까지 폭넓게 지원

자체 전문가들이 최초 발견한 취약점 보유

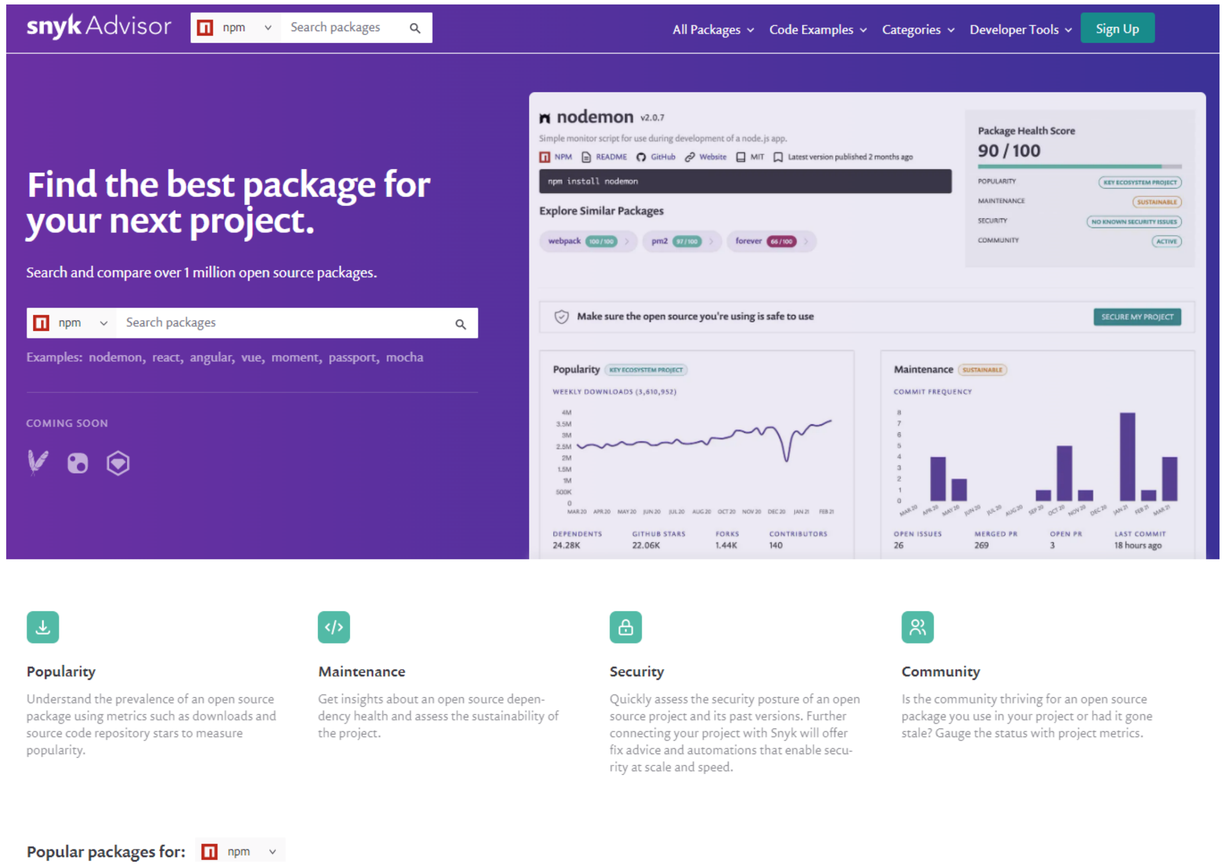
Snyk 자체 전문가들이 최초로 발견한 취약점 정보를 다량 보유하여 NVD 정보 대비 더 빠르게 정보 제공

조치 정보 제공

취약점 패치 정보 뿐만 아니라 Workaround 및 PoC 등 포괄적인 정보 제공

Snyk 부가 지원 기능 | Snyk Advisor

Snyk Advisor를 통해 오픈소스 패키지의 최신 취약점 정보를 제공합니다.



100만개 이상의 오픈소스 패키지 정보

Javascript, Python, Go 기반의 오픈소스 패키기 정보 보유

패키지 평가 정보 등 통계 정보 제공

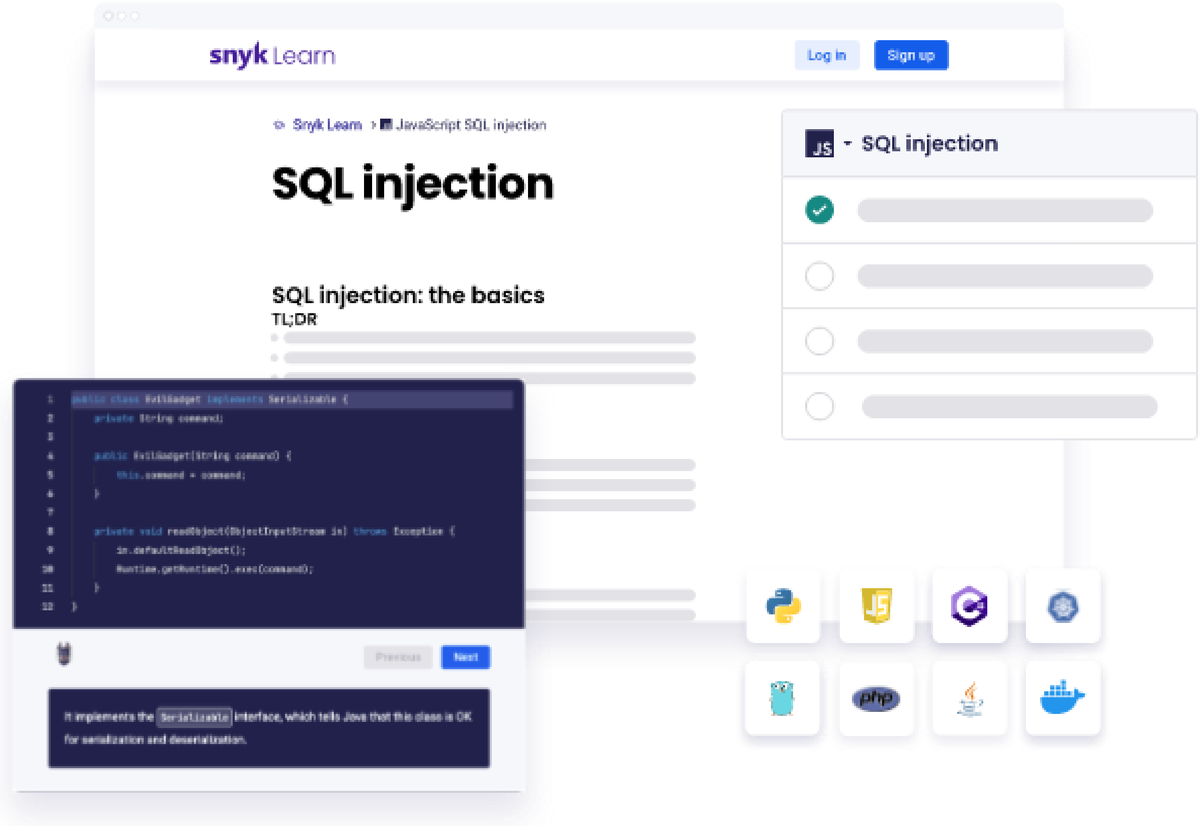
평판(인기도), 유지보수 현황, 보안취약점, 커뮤니티 활성화 정도 등 다양한 지표를 기준으로 한 평가 점수 등 다양한 통계 정보 제공

패키지별 세부 정보 제공

코드 샘플, 참고 가능한 유사 패키지, FAQ 등

Snyk 부가 지원 기능 | Snyk Learn

Snyk Learn을 통해 개발자를 위한 보안 조치 교육 프로그램을 제공합니다.



보안 전문가가 제작한 무료 교육

개발자를 위해 업계 전문가가 제작한 전문 교육으로
직관적인 온라인 교육 제공

필요한 관심 주제를 바로 학습 가능

즉석 학습 가능한 컨텐츠로 취약점 타입별로 바로
검색하여 학습할 수 있는 환경 제공

자가 테스트 가능한 환경 제공

자체 코드에서 발견된 문제를 기반으로
보안취약점을 이해하고, 수정하여 이슈 방지 가능

Snyk 특장점 | 다양한 도구와 연동체계

개발도구

- Docker
- Visual Studio
- VS Code
- WebStorm
- PyCharm
- buildah
- PhpStorm
- GoLand
- Eclipse
- IntelliJ
- RubyMine
- Snyk CLI

형상관리

- GitHub
- GitHub Enterprise
- Bitbucket Cloud
- Bitbucket Server
- Azure Repos
- GitLab

CI/CD

- CircleCI
- Snyk API
- BitBucket Pipelines
- AWS Code Pipeline
- Azure Pipelines
- TeamCity
- Concourse
- Jenkins

런타임환경

- IBM Cloud
- VMWare Tanzu
- Heroku
- Kubernetes
- Azure Functions
- Cloud Foundry
- Docker & others
- AWS Lambda
- RedHat Openshift

레지스트리

- Harbor
- Quay
- npm Enterprise Private Registry
- Amazon ECR
- Artifactory
- Google Container Registry
- Docker Hub
- Azure Container Registry

이슈관리

- Brinqua
- Fortify SSC
- Nucleus Security
- Vulcan
- Kenna Security
- RiskSense
- Snyk API
- Jira
- micro focus
- Slack

Snyk 특장점 | Snyk Vulnerability DB

Snyk Intel Vulnerability DB는 CVE 취약점 대비 441% 더 많은 취약점 정보를 포함하고 있고
NVD 대비 최대 46일 더 빠르게 취약점을 탐지하였습니다.
NVD에 기록된 JavaScript 관련 취약점의 92%는 Snyk이 먼저 탐지한 취약점들입니다.

다양한 취약점DB로부터 풍부한 데이터 확보

CVE, NVD와 같은 리소스를 통해 수집되는 데이터는 분석, 테스트 및 정제되어 데이터베이스에 포함됩니다.

새로운 취약점에 대한 독점 리서치

Snyk 보안팀은 제로데이 취약점과 같은 주요 컴포넌트의 심각한 취약점을 발견하기 위해 노력하고 있습니다.

위협 인텔리전스 시스템

보안게시판, Jira 게시판, Github 커밋 등을 통해 미보고된 취약점을 자동으로 식별합니다.

커뮤니티와의 관계

커뮤니티와 협업하고 신규 취약점에 대한 현상금 제도를 운영하여 공개커뮤니티로부터 수백개의 취약점을 수집하고 있습니다.

학계와의 협업

버클리, 버지니아공대, 워터루 등 학계의 박사급 연구진들과 파트너관계를 맺고 도구, 방법론 및 데이터를 교환하고 있고 결과는 독점적으로 공개됩니다.

Snyk 특장점 | 주요 특장점

1 개발자 우선 접근방식, DevSecOps 실현

개발자가 편리하게 사용하도록 설계되어 마찰이 없고 직관적입니다. 개발자가 현재 사용하는 도구에서 직접 문제를 쉽게 찾을 수 있을 뿐만 아니라 빠르게 수정할 수 있도록 지원합니다.

2 업계 최고의 광범위한 보안 인텔리전스 데이터베이스

Snyk의 데이터베이스는 기존 상용 데이터베이스보다 441% 더 많은 취약점 정보를 포함하고 기존 상용 취약점 대비 평균 46일 더 빠르게 취약점을 식별하였습니다.

3 자동 Fix를 통한 관리 리소스 최소화

소스코드 저장소와 연동된 UI를 통해 Pull Request를 생성하고 자동으로 Fix 하거나 CLI에서는 명령어 하나로 자동으로 취약점 없는 버전으로 Fix 할 수 있는 기능을 제공합니다.

Thank you!

snyk