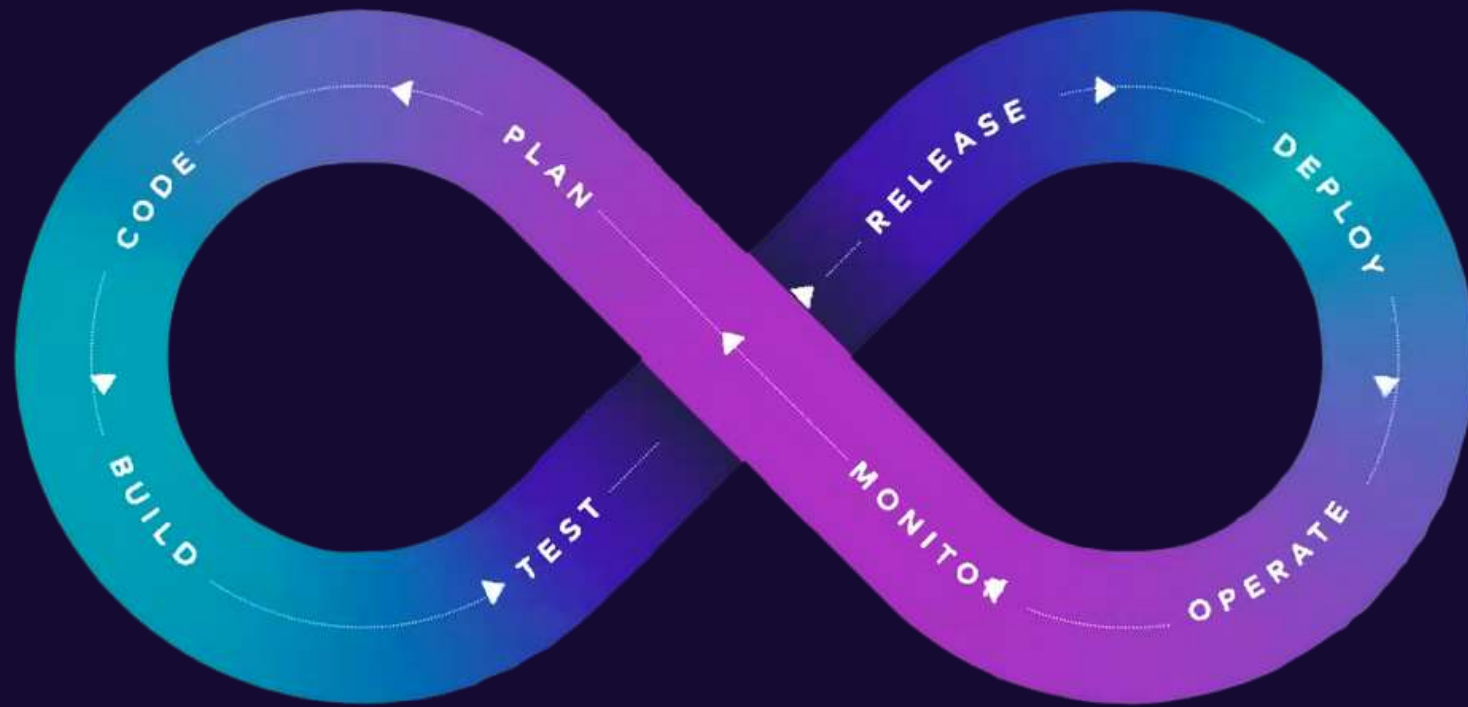


# snyk

Solution de sécurité pour  
développeurs



snyk

Maël Chevalier  
Léo Charreau



# IDE



Snyk Open Source



Snyk Code

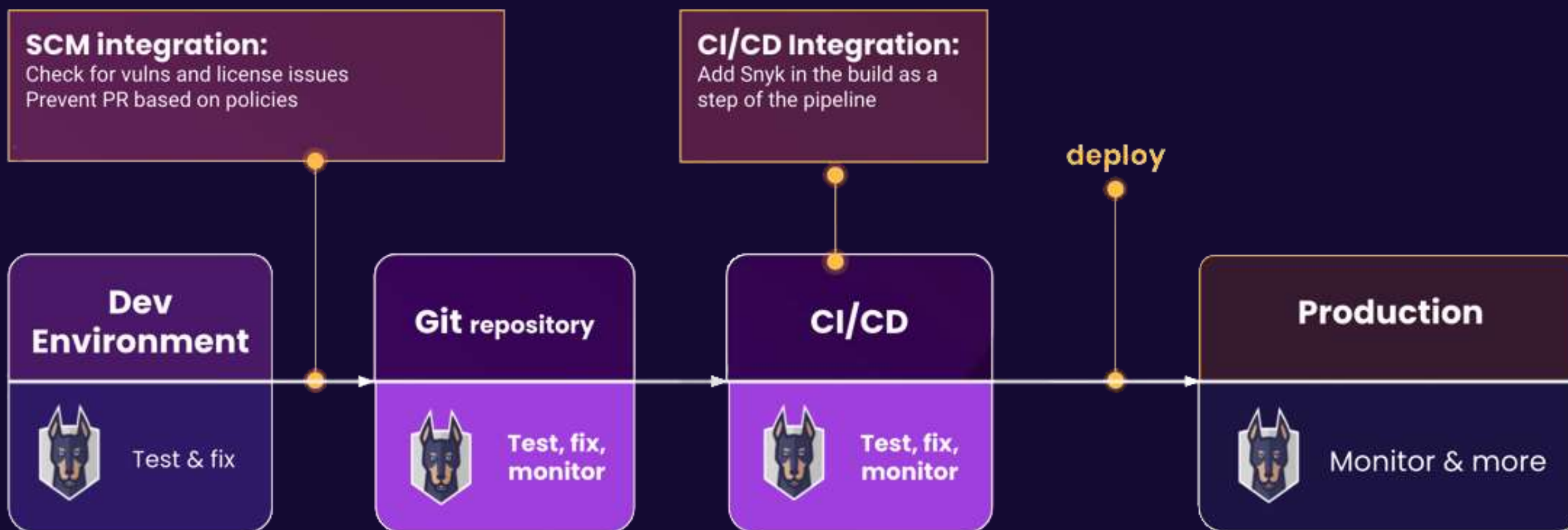


Snyk IaC



Snyk Container

## Intégration





The screenshot displays the Snyk dashboard for a project. At the top, it shows the Snyk logo and a user profile icon. Below this, a summary bar indicates the scan was created on Thursday, 16th April 2020, and is a snapshot of a recurring test performed 2 hours ago. The dashboard is divided into four main sections: IMPORTED BY (showing a user icon), PROJECTS OWNER (showing a user icon), ENVIRONMENT (showing a user icon), and BUSINESS CRITICALITY (showing a user icon). A 'Issues' tab is selected, showing a list of vulnerabilities. The first vulnerability is titled 'MySQL.Data.EntityFrameworkCore' with a score of 625. The second vulnerability is titled 'org.apache.struts:struts2-core' with a score of 435. A 'REACHABLE VULS' sidebar on the left shows a list of vulnerabilities with checkboxes and counts. The 'EXPLOIT MATURITY' section at the bottom left shows a progress bar and a list of vulnerabilities.

**snyk**

Created Thu 16th Apr 2020 | Snapshot by recurring test 2 hours ago

IMPORTED BY PROJECTS OWNER ENVIRONMENT BUSINESS CRITICALITY

Issues

ISSUE TYPE

REACHABLE VULS

☐ Reachable 3

☐

☐

EXPLOIT MATURITY

Search...

**FIX THIS VULNERABILITY**

**H** MySQL.Data.EntityFrameworkCore

**SCORE 625**

**FIX THIS VULNERABILITY**

**M** org.apache.struts:struts2-core

**SCORE 435**





# Snyk Learn

snyk Learn

Lessons

Learning paths

Learning progress

## JNDI injection

To exploit an example application vulnerable to JNDI injection we first create a malicious RMI server:

```
1 import java.rmi.registry.*;
2 import com.sun.jndi.rmi.registry.*;
3 import javax.naming.*;
4 import org.apache.naming.ResourceRef;
5
6 public class MaliciousRMIServer {
7     public static void main(String[] args) throws Exception {
8         //create our malicious RMI registry on port 1097 of our hosting server")
9         Registry registry = LocateRegistry.createRegistry(1097);
10
11         //this payload exploits unsafe reflection in org.apache.naming.factory.
12         BeanFactory
13         // NOTE: class namespace changed to jakarta.el.ELProcessor since Java 9
14         ResourceRef resourceRef = new ResourceRef("com.sun.el.ELProcessor", null, "XX", "XX");
```



### Code injection



Code injection: the basics



Code injection in action






Code injection under the hood




Code injection mitigation



▼  2 <b>comptegithubfac/term1</b>	1 C 7 H 15 M 6 L	⊕
 Code analysis	0 C 1 H 1 M 2 L Tested 2 days ago	⚙️
 pom.xml	1 C 6 H 14 M 4 L Tested 2 hours ago	⚙️

25 of 25 issues

Sort by highest priority score ▼

 **com.h2database:h2** - Remote Code Execution (RCE)

SCORE  
811

VULNERABILITY

CWE-94 ⓘ

CVE-2022-23221 ⓘ

CVSS 9.8 ⓘ

CRITICAL

SNYK-JAVA-COMH2DATABASE-2348247 ⓘ

Introduced through

com.h2database:h2@1.4.200

Exploit maturity


PROOF OF CONCEPT

Fixed in

com.h2database:h2@2.1.210

Show more detail ▼

NEW

 Learn about this type of vulnerability ⓘ

🗑️ Ignore

🔧 Partially fix this vulnerability

 **org.springframework.security:spring-security-web** - Authorization Bypass

SCORE  
731

VULNERABILITY

CWE-285 ⓘ

CVE-2022-22978 ⓘ

CVSS 8.2 ⓘ

HIGH

SNYK-JAVA-ORGSPRINGFRAMEWORKSECURITY-2833359 ⓘ

Introduced through

org.springframework.boot:spring-boot-starter-security@2.6.7

Exploit maturity

PROOF OF CONCEPT


Fixed in

org.springframework.security:spring-security-web@5.5.7,@5.6.4

Show more detail ▼

🗑️ Ignore

🔧 Partially fix this vulnerability

 **com.h2database:h2** - Remote Code Execution (RCE)

SCORE  
726

# Intégration TER

Merci pour votre attention.



**snyk**

Maël Chevalier  
Léo Charreau