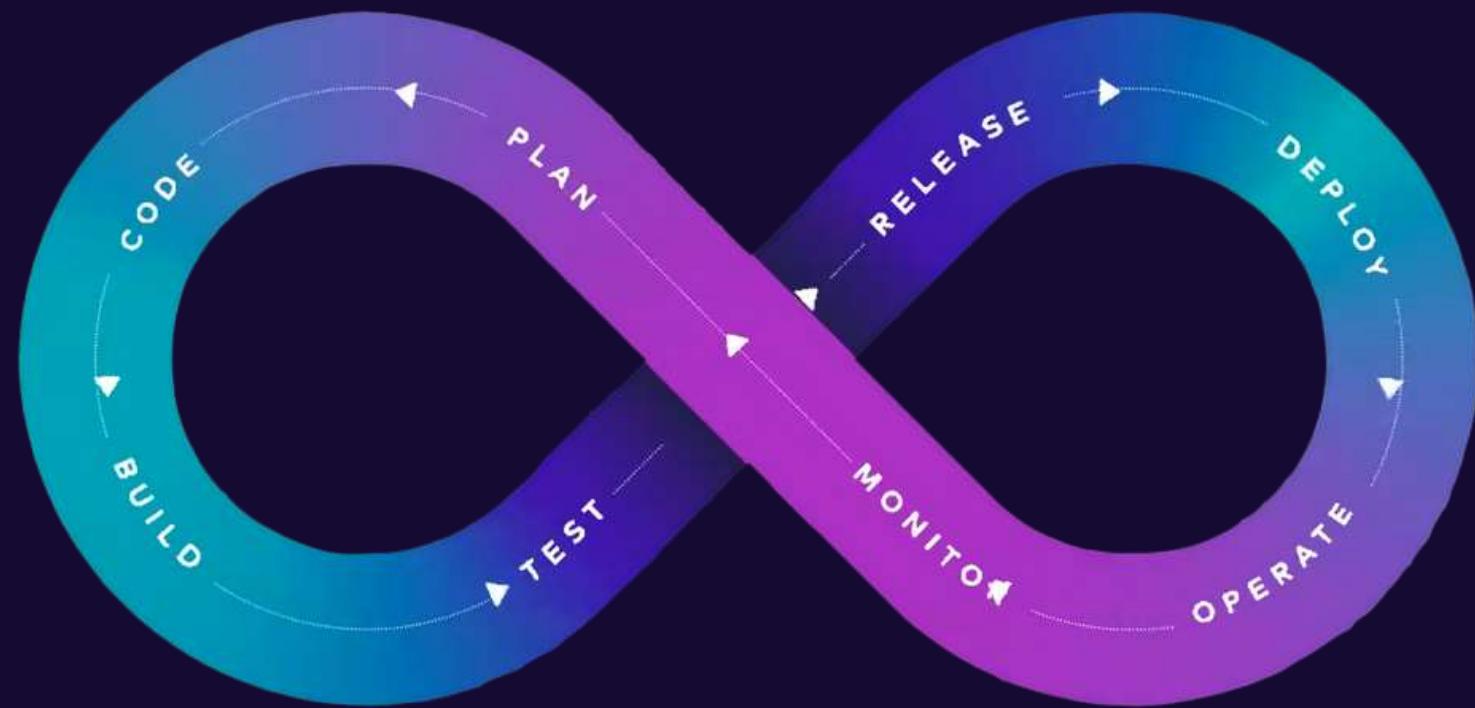




Solution de sécurité pour
développeurs



Maël Chevalier
Léo Charreau

IDE



Snyk Open Source



Snyk Code

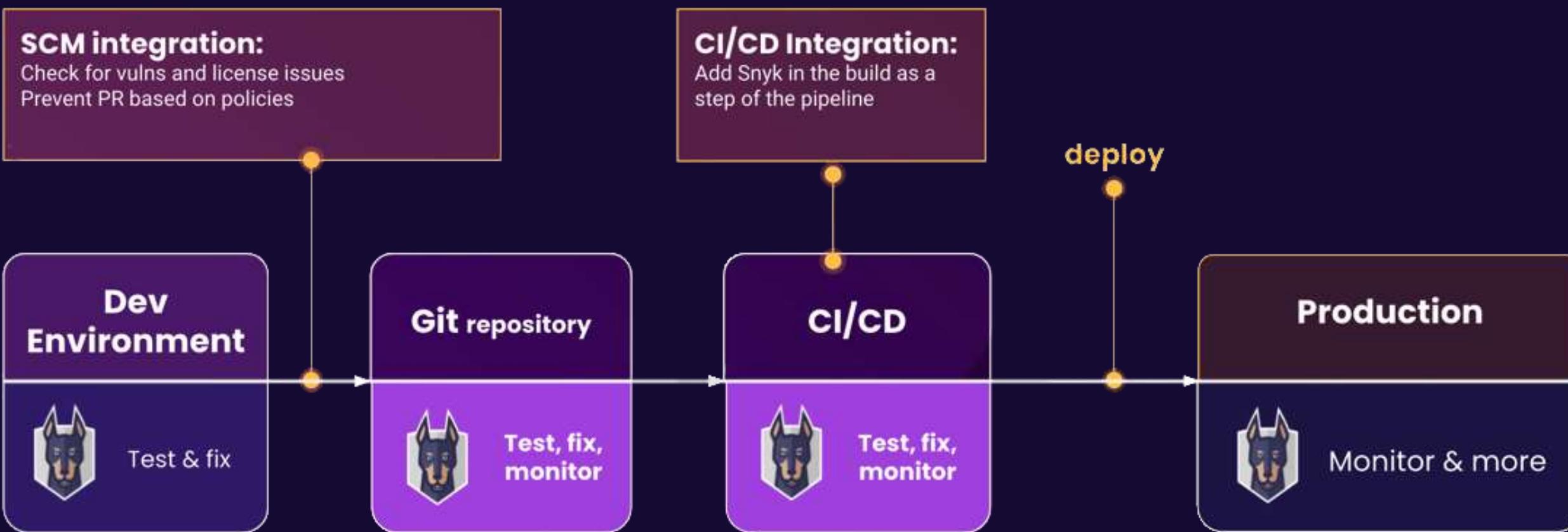


Snyk IaC



Snyk Container

Intégration



Correction auto.

The screenshot shows the Snyk Report interface. At the top left is a blue hexagonal icon featuring a black Doberman Pinscher head. Below it, the word "Rapport" is displayed in a large, light blue font. The main area is a white card titled "snyk". It includes sections for "IMPORTED BY", "PROJECTS OWNER", "ENVIRONMENT", and "BUSINESS CRITICALITY". A "Issues" button is present. On the right side of the card is a green button labeled "OPEN FIX REQUEST" with a gear icon. Below the card, there's a search bar and a "FIX THIS VULNERABILITY" button. The main content area lists vulnerabilities:

- MySQL.Data.EntityFrameworkCore** (Reachable, Score: 625) - A modal window is open over this item, also labeled "FIX THIS VULNERABILITY".
- org.apache.struts:struts2-core** (Reachable, Score: 435)

On the left, there are filters for "REACHABLE VULS" (checkboxes for "Reachable" and "Exploit Maturity") and a "SEARCH..." input field.

This screenshot shows the Snyk Fixes interface. It features a large green button at the top labeled "OPEN FIX PR" with a gear icon. Below it is a section titled "ISSUES WITH A FIX" with a sub-section "AN UPGRADE OR PATCH IS AVAILABLE TO FIX THESE ISSUES". There are five small red icons with "H" and "P" symbols. To the right is a dark sidebar with horizontal bars.



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<?> new ResourceRef<?>("jakarta.el.ELProcessor", null, null);
```

Code injection

- Code injection: the basics
- Code injection in action
- Code injection under the hood
- Code injection mitigation

Intégration TER

25 of 25 issues

Sort by highest priority score ▾

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

VULNERABILITY | CWE-94 ⓘ | CVE-2022-23221 ⓘ | CVSS 9.8 ⓘ | CRITICAL | SNYK-JAVA-COMH2DATABASE-2348247 ⓘ

SCORE 811

Introduced through com.h2database:h2@1.4.200
Fixed in com.h2database:h2@2.1.210

Show more detail ↴ [Learn about this type of vulnerability ⓘ](#)

[Ignore](#) [Partially fix this vulnerability](#)

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

VULNERABILITY | CWE-285 ⓘ | CVE-2022-22978 ⓘ | CVSS 8.2 ⓘ | HIGH | SNYK-JAVA-ORGSPRINGFRAMEWORKSECURITY-2833359 ⓘ

SCORE 731

Introduced through org.springframework.boot:spring-boot-starter-security@2.6.7
Fixed in org.springframework.security:spring-security-web@5.5.7, @5.6.4

Show more detail ↴ [Learn about this type of vulnerability ⓘ](#)

[Ignore](#) [Partially fix this vulnerability](#)

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

VULNERABILITY | CWE-94 ⓘ | CVE-2022-23221 ⓘ | CVSS 9.8 ⓘ | CRITICAL | SNYK-JAVA-COMH2DATABASE-2348247 ⓘ

SCORE 726



comptegithubfac/term1	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 ⚙️

Merci pour votre attention.



Maël Chevalier
Léo Charreau