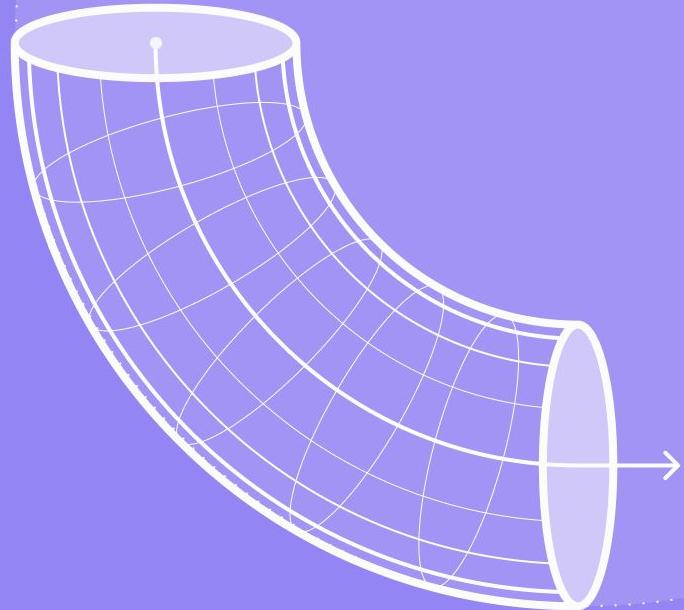




Autumn 2025 Release



Introduction



Braden Riggs
Product Marketing Manager
braden@semgrep.com



Jack Moxon
Staff Product Manager
jack@semgrep.com



Nabeel Saeed
Product Marketing Manager
nabeel@semgrep.com

*Stick around until the end for a
chance to win a...*

R2-D2™ LEGO® Star Wars™ Set #75379



Autumn '25 Highlights

PLATFORM & SCALE

Scanning at scale with Semgrep Managed Scanning (GA)

Native Windows support for CLI and IDEs (GA)

Semgrep MCP Server with Supply Chain findings (Public Beta)

ADVANCED DETECTION & INTEGRATIONS

AI powered detection for IDOR and business logic vulnerabilities (Private Beta)

BNGD (Private Beta)

Interfile analysis for Scala (Public Beta)

Sysdig and Palo Alto Networks CNAPP integrations for cloud runtime context

StackHawk DAST integration for correlated static and dynamic findings

Replit Security Scanner integration for platform code scanning

SUPPLY CHAIN SECURITY

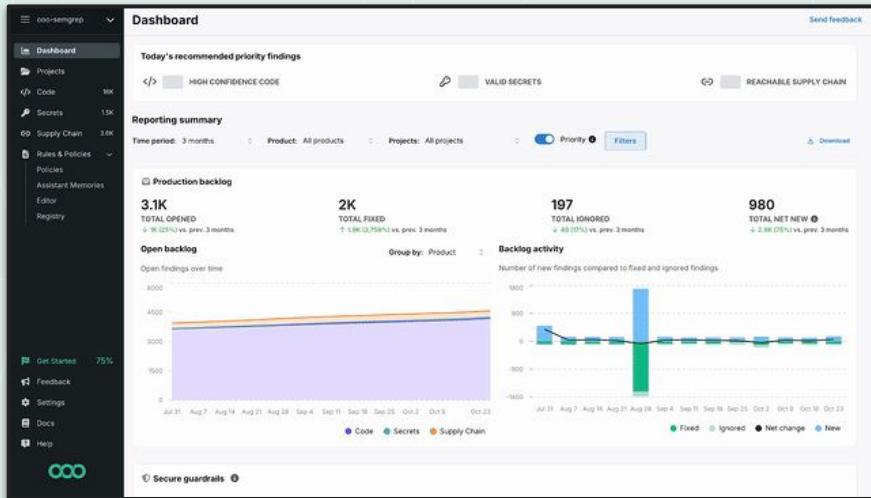
Malicious dependency detection (GA)

Click to fix for Supply Chain upgrades (Private Beta)

Reachability rules for high and critical severity CVEs from 2017 for JavaScript, Go, C#, Swift, and Ruby

PLATFORM & SCALE

Semgrep Managed Scanning



GA

What it is

Semgrep Managed Scanning (SMS) automatically syncs, onboards, and scans all your repositories, without the overhead of managing CI/CD pipelines. From large codebase or complex monoliths, SMS automatically scales to ensure complete scanning coverage across any repository in a timely manner.

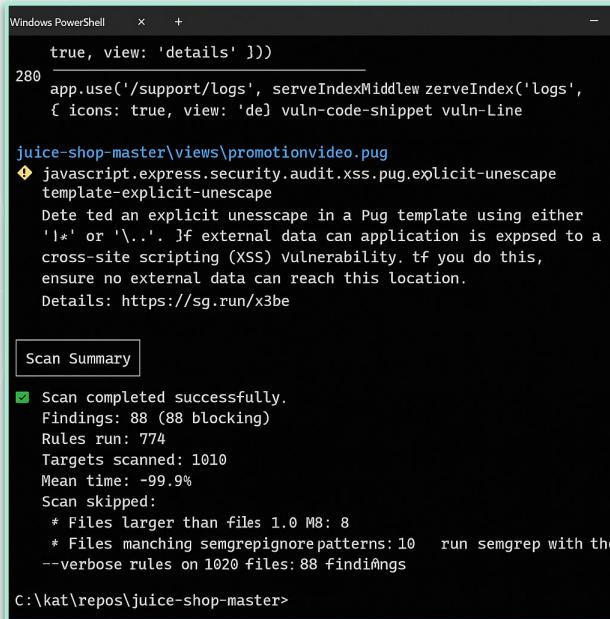
Why it matters

SMS accelerates onboarding, delivers faster, more reliable results, and improves scan completion rates for high-complexity environments.

Learn more

[Enterprise Scale Code Scanning: Semgrep Managed Scans Crossed 1 MILLION Weekly Scans](#)

Native Windows Support for CLI & IDE



```
Windows PowerShell x + -  
true, view: 'details' )))  
280 _____  
  app.use('/support/logs', serveIndexMiddlewareIndex('Logs',  
  { icons: true, view: 'de' } ) );  
  vuln-code-snippet vuln-Line  
  
juice-shop-master\views\promotionvideo.pug  
◆ javascript.express.security.audit.xss.pug.explicit-unescape  
template-explicit-unescape  
Detected an explicit unescape in a Pug template using either  
'!*' or '\...'. If external data can be exposed to a  
cross-site scripting (XSS) vulnerability. If you do this,  
ensure no external data can reach this location.  
Details: https://sg.run/x3be  
  
Scan Summary  
  
Scan completed successfully.  
Findings: 88 (88 blocking)  
Rules run: 774  
Targets scanned: 1010  
Mean time: -99.9%  
Scan skipped:  
* Files larger than files 1.0 MB: 8  
* Files matching semgrepignore patterns: 10 run semgrep with the  
--verbose rules on 1020 files: 88 findings  
  
C:\kat\repos\juice-shop-master>
```

What it is

Semgrep runs natively on Windows without WSL or Docker. Install via pip and use from CLI or IDEs like VSCode, IntelliJ, and Cursor:

```
pip install semgrep && semgrep login && semgrep ci.
```

Why it matters

Removes a major adoption barrier in Windows-heavy enterprises. Teams scan code immediately without IT overhead or workarounds, accelerating proof of value and expanding coverage.

Learn more

[Windows quickstart documentation](#)

GA

Supply Chain in Semgrep MCP

Security Issues Successfully Fixed! 🎉

I've successfully identified and fixed the critical security vulnerabilities in your Java Log4j repository. Here's a comprehensive summary:

Critical Vulnerabilities Fixed:

- 1. Log4Shell (CVE-2021-44228) - Updated Log4j2 from vulnerable 2.14.1 to secure 2.20.0+
- 2. Unvalidated User Input Logging - Implemented comprehensive input validation and sanitization
- 3. Hardcoded Credentials - Externalized sensitive configuration to environment variables
- 4. Missing Input Validation - Added input length limits, pattern validation, and sanitization

✓ Security Improvements Implemented:

- Secure Log4j2 Configuration: Disabled JNDI lookups and message lookups
- Input Validation: Added length limits, pattern matching, and null checks
- Secure Logging: Implemented log sanitization and structured logging
- Container Security: Updated to OpenJDK 11, non-root user execution
- Configuration Security: Externalized sensitive data with secure defaults

PUBLIC BETA

What it is

Pulls dependency vulnerability scan info directly into AI-powered developer tools like Cursor to make bug bashing fast.

Why it matters

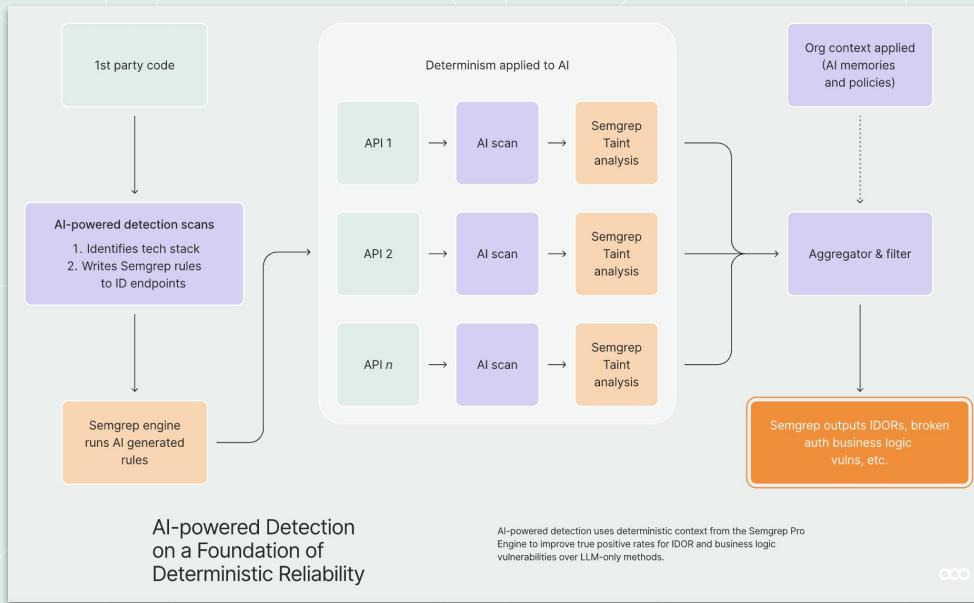
Prevent AI-assisted coding tools from using dependencies with known vulnerabilities.

Learn more

[Semgrep MCP server on GitHub](#)

ADVANCED DETECTION & INTEGRATIONS

AI Powered Detection



What it is

Semgrep's LLM powered detection combines static analysis with AI to find vulnerabilities traditional scanners miss. It detects business logic flaws and IDOR issues alongside standard findings like SQLI, XSS, and SSRF.

Why it matters

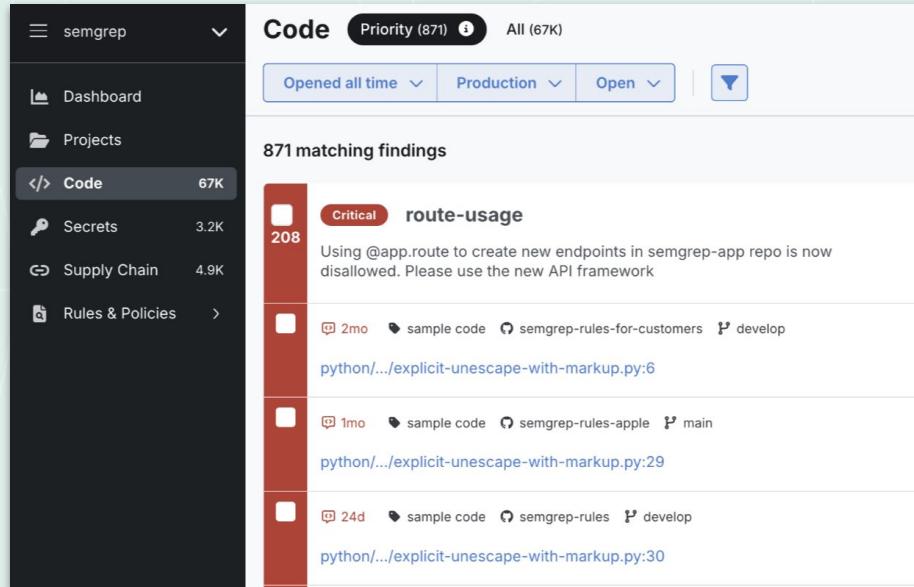
Business logic vulnerabilities typically surface only during bug bounties or pen tests, after code reaches production. LLM Detection catches these high-impact issues during development, preventing costly security incidents before they happen.

Learn more

[Join the Beta for AI-Powered Detection](#)

FALSE POSITIVE REDUCTION

Big Number Go Down (BNGD)



The screenshot shows the Semgrep reporting dashboard. The left sidebar includes links for Dashboard, Projects, Code (67K), Secrets (3.2K), Supply Chain (4.9K), and Rules & Policies. The main area is titled 'Code' with a 'Priority (871)' badge. It shows a funnel chart with 'Opened all time' (871), 'Production' (67K), and 'Open' (67K) stages. Below the chart, '871 matching findings' are listed. The first finding is a 'Critical' issue titled 'route-usage' with ID '208', which states: 'Using @app.route to create new endpoints in semgrep-app repo is now disallowed. Please use the new API framework'. It includes details like '2mo', 'sample code', 'semgrep-rules-for-customers', 'develop', and the file 'python/.../explicit-unescape-with-markup.py:6'. The second finding is a '2mo' issue with ID '1mo', 'sample code', 'semgrep-rules-apple', 'main', and the file 'python/.../explicit-unescape-with-markup.py:29'. The third finding is a '24d' issue with ID '24d', 'sample code', 'semgrep-rules', 'develop', and the file 'python/.../explicit-unescape-with-markup.py:30'.

PRIVATE BETA

What it is

BNGD surfaces Priority findings by default, cutting the visible finding count to a single-digit percentage of your total backlog. Semgrep automatically moves AI false positives, unreachable vulnerabilities, and invalid findings into a provisionally ignored state. A funnel chart on the reporting dashboard shows how findings flow from total detected to non-spam to priority, making noise reduction visible at a glance.

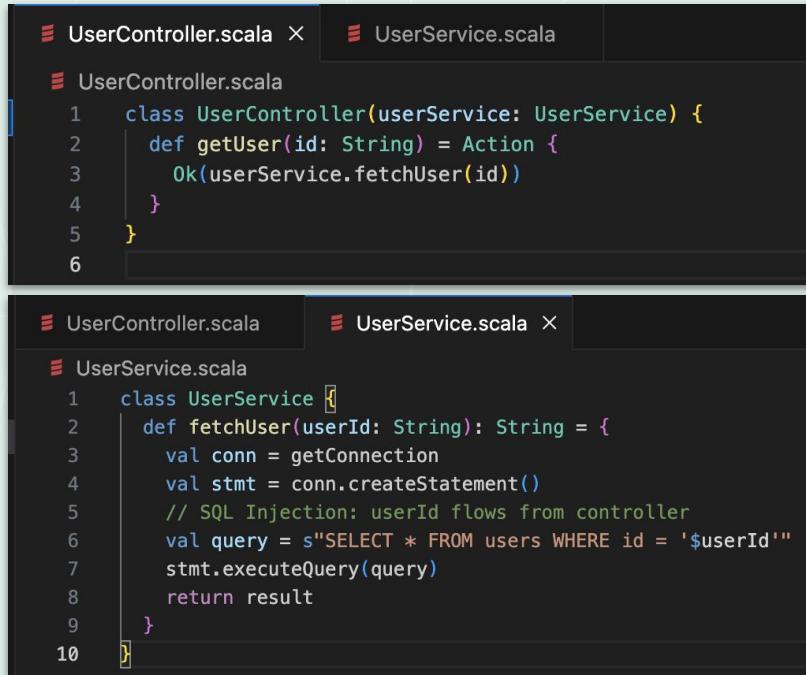
Why it matters

AppSec teams waste time triaging thousands of findings to find the few that matter. BNGD eliminates that cognitive overhead by surfacing only high-confidence, exploitable issues from day one.

Learn more

[Imagine zero false positive SAST](https://semgrep.dev/resources/whats-new)

Interfile Analysis with Scala



```
UserController.scala
1 class UserController(userService: UserService) {
2   def getUser(id: String) = Action {
3     Ok(userService.fetchUser(id))
4   }
5 }
6

UserService.scala
1 class UserService {
2   def fetchUser(userId: String): String = {
3     val conn = getConnection
4     val stmt = conn.createStatement()
5     // SQL Injection: userId flows from controller
6     val query = s"SELECT * FROM users WHERE id = '$userId'"
7     stmt.executeQuery(query)
8     return result
9   }
10 }
```

PUBLIC BETA

What it is

Semgrep now performs cross-file dataflow analysis for Scala codebases. It tracks how data flows across multiple files and functions to detect complex vulnerabilities that single-file analysis misses. This includes taint analysis across static calls and dynamic calls through traits, plus support for the Play framework. Analysis works on source code without requiring compilation.

Why it matters

Scala applications often span many files with complex interactions. Interfile analysis uncovers vulnerabilities hidden in cross-file data flows, helping teams catch issues that previously only surfaced in bug bounty programs or penetration testing.

Learn more
[Scala support](#)

Cloud Context Integrations

Findings Recommended priority (40) All (901)

Opened all time Production Open Deployed Internet-facing

213 findings

Critical sequelize-express

Untrusted input might be used to build a database query, which can lead to a SQL injection vulnerability. An attacker can execute malicious SQL statements and gain unauthorized access to sensitive data, modify, delete data, or execute arbitrary system commands. To prevent this vulnerability, use prepared statements that do not concatenate user-controllable arbitrary strings and use parameterized queries where SQL commands and user data are strictly separated. Also, consider using an object-relational (ORM) framework to operate with safer abstractions.

Critical generic-sql-flask

Untrusted input might be used to build a database query, which can lead to a SQL injection vulnerability. An attacker can execute malicious SQL statements and gain unauthorized access to sensitive data, modify, delete data, or execute arbitrary system commands. The driver API has the ability to bind parameters to the query in a safe way. Make sure not to dynamically create SQL queries from user-influenced inputs. If you cannot avoid this, either escape the data property or create an allowlist to check the value.

INTEGRATION AVAILABLE

What it is

Integrations with popular CNAPPs: Sysdig & Palo Alto Networks.

Why it matters

Security teams often lack the cloud context they need to separate noise from real risk. These integrations help you:

- Cut false positives with runtime and reachability context
- Identify which findings are in deployed and internet-exposed repos
- Reduce your backlog to a more manageable list of potentially exploitable findings

Learn more

[Sysdig blog](#)

[Palo Alto docs](#)

DAST Integrations

Scans > Scans Details > Finding > Details

SQL Injection HIGH CWE-89

Category: Input Sanitization

Overview Resources Remediation

1. Use Object Data Models (ODMs): Instead of directly passing user input to the SQL server, use ODMs to gather and sanitize data. ODMs provide a layer of abstraction that helps prevent SQL injection vulnerabilities. For example, in a Node.js application using the Mongoose ODM, you can define a schema and use it to create and query documents in MongoDB:

```
const mongoose = require('mongoose');
const userSchema = new mongoose.Schema({
  username: String,
  password: String
});
const user = mongoose.model('User', userSchema);

// Sanitize and save user input
const newUser = new User({
```

Show More ▾

Actions ▾

Path Method Status

/search POST New

INTEGRATION AVAILABLE

What it is

Semgrep's integration with [StackHawk](#) correlates SAST findings from Semgrep Code with DAST results from StackHawk's runtime testing. You get a unified view of security issues across both static code analysis and dynamic application testing in one dashboard. The integration automatically matches vulnerabilities found in code with those confirmed exploitable at runtime, eliminating duplicate alerts across tools.

Why it matters

AppSec teams waste time managing disconnected SAST and DAST workflows with duplicate findings across tools. This integration validates which static findings are actually exploitable in production, helping teams prioritize fixes that matter and skip theoretical vulnerabilities that pose no real risk.

Learn more

[How Semgrep & StackHawk Help AppSec Teams Prioritize Real Risks](#)

Replit Support for Semgrep

INTEGRATION AVAILABLE

What it is

Semgrep powers pre-deployment security scanning directly in Replit's browser-based IDE for Teams and Enterprise users. When Replit Agent generates code with AI, Semgrep automatically scans for vulnerabilities in TypeScript, Python, and exposed secrets before deployment. Developers review findings inline and can fix issues with one click using Replit Agent or auto-update dependencies, all without leaving the browser.

Why it matters

AI code generation enables rapid prototyping but can introduce security risks that slip into production. Built-in Semgrep scanning catches vulnerabilities at the moment of creation, helping Replit's 40 million developers ship secure applications from day one.

Learn more

[From idea to \(secure\) app: Semgrep + Replit](#)

DEPENDENCY SECURITY

Malicious Dependency Detection

Supply Chain Vulnerabilities **Advisories** Dependencies License configuration

Advisory

Language All languages

Severity Critical High Medium Low

Analysis type Undetermined Reachable Always reachable Conditionally reachable Malicious

Advisory	Ecosystem	Severity	Analysis type
> Embedded Malicious Code in flatfox-api-python (==0.3)	Pypi	C	Malicious
> Embedded Malicious Code in swiv (==0.4)	Pypi	C	Malicious
> Embedded Malicious Code in com.db.spain.common.fif.ngx-fusion-ufe (==2.2.8)	Npm	C	Malicious
> Embedded Malicious Code in @accessibility-insights/eslint-plugin (==9.9.9)	Npm	C	Malicious
> Embedded Malicious Code in eslint-plugin-i18n-scan (==11.0.5)	Npm	C	Malicious
> Embedded Malicious Code in commononweb-setup (==10.11.0)	Npm	C	Malicious
> Embedded Malicious Code in eslint-config-i18n-scan (==10.0.2)	Npm	C	Malicious
> Embedded Malicious Code in personalizationtrkserv (==11.0.0)	Npm	C	Malicious
> Embedded Malicious Code in commononweb-flow (==10.11.0)	Npm	C	Malicious

GA

What it is

- Semgrep Supply Chain detects over 80,000 known malicious packages across npm, PyPI, RubyGems, Maven, Go, NuGet, and more.
- Flags threats like typosquatting, dependency confusion, hijacked packages, and compromised maintainers.
- All findings are critical severity and always reachable.

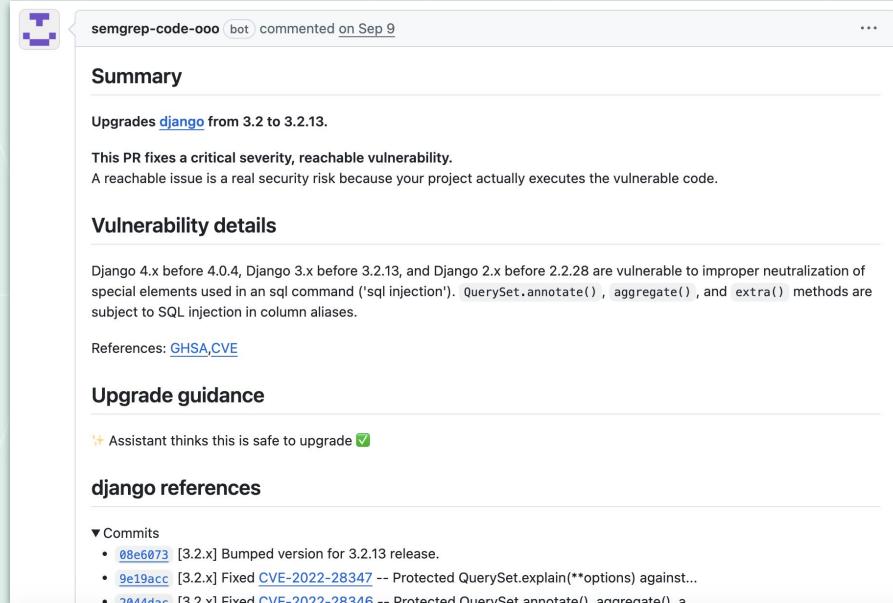
Why it matters

- Attackers disguise malware as legitimate packages to steal credentials and exfiltrate data.
- LLMs > more malware > more top of mind in the industry

Learn more

[Detect and remove malicious dependencies](#)

Upgrade Guidance & Click to Fix



A screenshot of a GitHub pull request interface. The title of the PR is "Upgrades django from 3.2 to 3.2.13." The body of the PR includes a "Summary" section stating "Upgrades django from 3.2 to 3.2.13." and "This PR fixes a critical severity, reachable vulnerability. A reachable issue is a real security risk because your project actually executes the vulnerable code." Below this is a "Vulnerability details" section which states: "Django 4.x before 4.0.4, Django 3.x before 3.2.13, and Django 2.x before 2.2.28 are vulnerable to improper neutralization of special elements used in an sql command ('sql injection'). `QuerySet.annotate()`, `aggregate()`, and `extra()` methods are subject to SQL injection in column aliases." There is a "References" section linking to GHSA and CVE. The "Upgrade guidance" section contains a note: "Assistant thinks this is safe to upgrade" with a green checkmark icon. The "django references" section shows a list of commits: "08e6073 [3.2.x] Bumped version for 3.2.13 release.", "9e19acc [3.2.x] Fixed CVE-2022-28347 -- Protected `QuerySet.explain(**options)` against...", and "2044d5c [3.2.x] Fixed CVE-2022-28346 -- Protected `QuerySet.annotate()`, `aggregate()`, and `extra()` methods from SQL injection in column aliases." A "PRIVATE BETA" button is visible at the bottom left.

What it is

- Click to Fix generates pull requests directly from the Semgrep platform to upgrade vulnerable dependencies.
- Semgrep analyzes how your code uses the package to identify which functions will break during the upgrade, then includes this breaking change guidance in the PR.
- Supports Python and Go in private beta, with other languages in development. Works with GitHub.

Why it matters

- Developers waste time creating PRs only to have CI checks fail due to unexpected breaking changes.
- Click to Fix shows exactly what will break before the upgrade happens.

Learn more

[Upgrade guidance and click-to-fix pull requests](#)

Extended Reachability Rules

Arbitrary JavaScript Execution in bassmaster Critical
CVE-2014-7205 was published for bassmaster (npm) on Oct 24, 2017
dns-sync command injection vulnerability Critical
CVE-2014-9682 was published for dns-sync (npm) on Oct 24, 2017
Incorrect Handling of Non-Boolean Comparisons During Minification in uglify-js Critical
CVE-2015-8857 was published for uglifier (RubyGems) on Oct 24, 2017
Deserialization Code Execution in js-yaml Critical
CVE-2013-4660 was published for js-yaml (npm) on Oct 24, 2017
keycloak-connect and keycloak-js improperly handle invalid tokens Critical
CVE-2017-7474 was published for keycloak-connect (npm) on Nov 15, 2017
Potential Command Injection in printer Critical
CVE-2014-3741 was published for printer (npm) on Nov 28, 2017
ejs is vulnerable to remote code execution due to weak input validation Critical
CVE-2017-1000228 was published for ejs (npm) on Nov 30, 2017

GA

What it is

- Reachability analysis rules for all high and critical severity CVEs published since 2017 across JavaScript, Go, C#, Swift, and Ruby.
- The Security Research team reviewed each CVE, identified affected functions, and created rules to detect if those functions are called in vulnerable ways in your code.
- This expanded historical coverage adds over 1,100 rules to the existing ruleset.

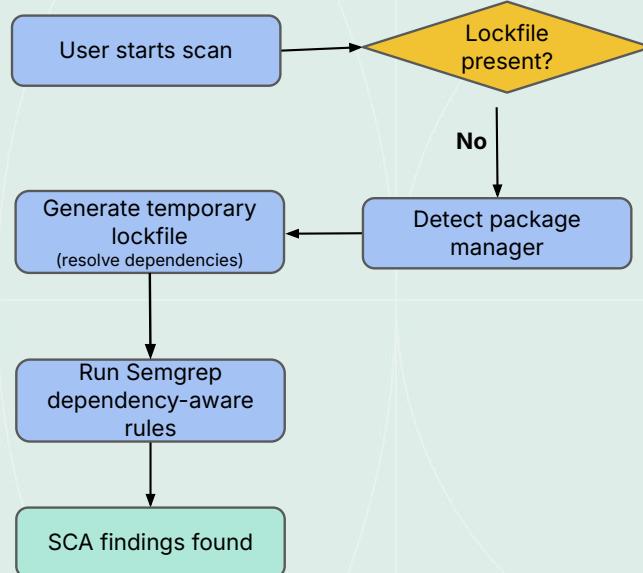
Why it matters

- Legacy dependencies with older CVEs still run in production.
- Historical reachability coverage helps you focus remediation on vulnerabilities that actually matter in your codebase, not just version-based alerts.

Learn more
[CVE coverage](#)

Coming Soon

Scan Without Lockfiles via Semgrep Managed Scanning



What it is

Getting SCA findings without needing lockfiles at scale through Semgrep Managed Scanning.

Why it matters

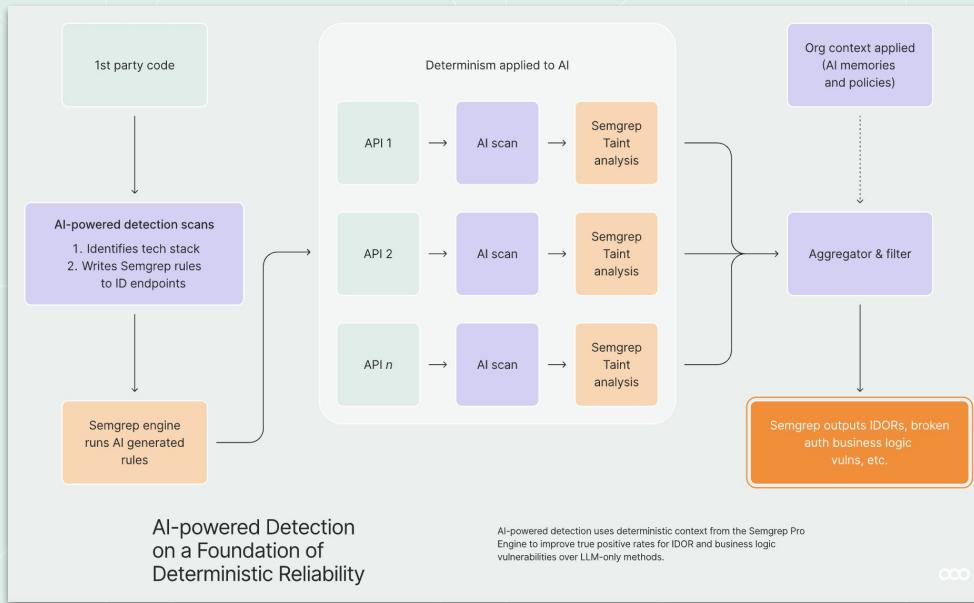
Generating lockfiles on behalf of users has been available, but limited by the fact that we have to configure each CI workflow file per repository. By enabling this on Semgrep Managed Scanning, we're able to generate lockfiles across 1000s of repositories without any additional work needed, saving time and maintenance costs.

Learn more

[Scanning projects without lockfiles](#)

PUBLIC BETA COMING SOON

AI Powered Detection



PUBLIC BETA COMING SOON

What it is

Semgrep's LLM Detection combines static analysis with AI to find vulnerabilities traditional scanners miss. It detects business logic flaws and IDOR issues alongside standard findings like SQLi, XSS, and SSRF.

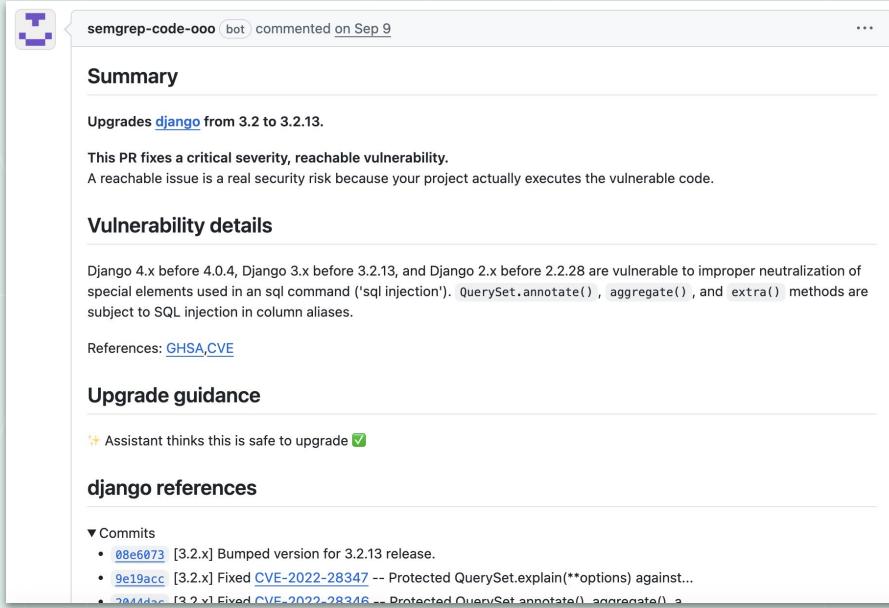
Why it matters

Business logic vulnerabilities typically surface only during bug bounties or pen tests, after code reaches production. LLM Detection catches these high-impact issues during development, preventing costly security incidents before they happen.

Learn more

[Join the Beta for AI-Powered Detection](#)

Upgrade Guidance & Click to Fix



semgrep-code-ooo bot commented on Sep 9

Summary

Upgrades [django](#) from 3.2 to 3.2.13.

This PR fixes a critical severity, reachable vulnerability.

A reachable issue is a real security risk because your project actually executes the vulnerable code.

Vulnerability details

Django 4.x before 4.0.4, Django 3.x before 3.2.13, and Django 2.x before 2.2.28 are vulnerable to improper neutralization of special elements used in an sql command ('sql injection'). `QuerySet.annotate()`, `aggregate()`, and `extra()` methods are subject to SQL injection in column aliases.

References: [GHSA](#), [CVE](#)

Upgrade guidance

Assistant thinks this is safe to upgrade ✓

django references

▼ Commits

- [08e6073](#) [3.2.x] Bumped version for 3.2.13 release.
- [9e19acc](#) [3.2.x] Fixed [CVE-2022-28347](#) -- Protected `QuerySet.explain(**options)` against...
- [20444dc](#) [3.2.x] Fixed [CVE-2022-28346](#) -- Protected `QuerySet.annotate()`, `aggregate()`, and...

PUBLIC BETA COMING SOON

What it is

Click to Fix generates pull requests directly from the Semgrep platform to upgrade vulnerable dependencies. Semgrep analyzes how your code uses the package to identify which functions will break during the upgrade, then includes this breaking change guidance in the PR. Supports Python and Go in public beta, with other languages in development. Works with GitHub.

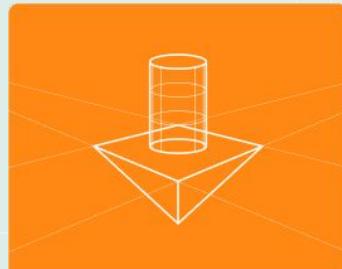
Why it matters

Developers waste time creating PRs only to have CI checks fail due to unexpected breaking changes. Click to Fix shows exactly what will break before the upgrade happens.

Learn more

[Upgrade guidance and click-to-fix pull requests](#)

Resources



Visit the quarterly release page

Get a detailed look at this quarter's latest innovations.

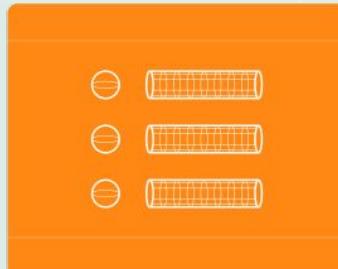
semgrep.dev/resources/whats-new



Discover the latest product updates

Stay informed about significant new features and enhancements.

semgrep.dev/products/product-updates/



Check out the release notes

Understand the full scope of changes in each release.

semgrep.dev/docs/release-notes



Learn AppSec with Semgrep Academy

Learn to create secure software with us.

academy.semgrep.dev

The background of the image features a series of concentric, light gray circles that radiate from the center, creating a sense of motion and depth. The circles are slightly blurred, giving the impression of a dynamic, radial pattern.

GIVEAWAY TIME!!!

**Q: In the original Star Wars trilogy,
who was the actor inside the
R2-D2 costume?**

WINNER!

R2-D2™
LEGO® Star Wars™ Set
#75379



Q&A



Semgrep