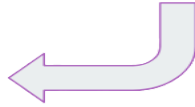


Level Up Your CI/CD: Building a secure pipeline with OSS

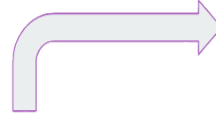


Slides



Repo

Fork it and clone it



<https://github.com/unicrons/secure-pipeline-workshop>

Level Up Your CI/CD: Building a secure pipeline with OSS



About us

Paco

Platform Engineer @ [flywire](#)

 @sanchezpaco

 pacosanchezlopez

Andoni

Junior Career Breaker @ \$HOME

 @andoniaf

 andoniaf



And he's Edu



Agenda

1. Introduction

- What is CI / CD / CS ?
- The need for secure pipeline

2. The "perfect pipeline" overview

3. Workshop:

- Brief explanation of the step
- Integrate OSS tooling
- Fix the issue



What is CI / CD / CS?

Continuous Integration (CI)

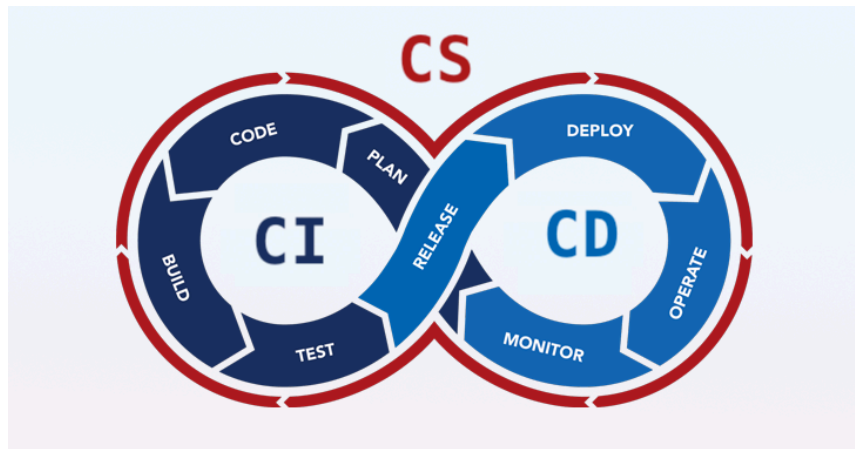
- Developers merge code multiple times a day
- Every merge triggers an automated build and test

Continuous Delivery (CD)

- Every change can be deployed to production
- "Release day" no longer exists

Continuous Security (CS)

- Embedding security into the process
- Automating security controls and tests throughout the pipeline (like we do for QA)
- **Find vulnerabilities when they are cheapest and easiest to fix**



Why is security in the pipeline important?

Why add security steps to the pipeline?

- Prevents supply chain compromises (e.g., malware injection)
- Enables early threat detection (SAST/DAST scans, secret checks)
- Reduces risks in automated environments (leaked keys, backdoors)
- Ensures compliance & builds trust



And why is security of the pipelines important?

We will talk more in the Pipeline Security step, but let's mention some recent examples:

MALWARE
Threat Brief: Widespread Impact of the Axios Supply Chain Attack

Trivy Compromised: Everything You Need to Know about the Latest Supply Chain Attack

 Rami McCarthy
March 20, 2026 5 minute read

Security Mar 26, 2026

The LiteLLM Supply Chain Attack: What Happened, Why It Matters, and What to Do Next

The Bitwarden CLI Supply Chain Attack: What Happened and What to Do

How attackers compromised Bitwarden's CLI and enlisted the help of AI coding agents to spread a worm and harvest developer secrets.

Written by  Kiran Raj
Published on April 23, 2026
Updated on April 23, 2026
Topics: Malware Security



And why is security of the pipelines important?

hackerbot-claw: An AI-Powered Bot Actively Exploiting GitHub Actions – Microsoft, DataDog, and CNCF Projects Hit So Far

Varun Sharma 

March 1, 2026



hackerbot-claw

Attack 1: avelino/awesome-go - Token Theft via Poisoned Go Script

Attack 2: project-akri/akri - Direct Script Injection

Attack 3: microsoft/ai-discovery-agent - Branch Name Injection

Attack 4: DataDog/datadog-iac-scanner - Filename Injection

Attack 5: ambient-code/platform - AI Prompt Injection

Attack 6: aquasecurity/trivy - Evidence Cleared

Attack 7: RustPython/RustPython — Branch Name Injection with Base64 Payload



Why is pipeline security important?

But my repo is private...

...until is not.

Twitch breach leads to leak of source code and streamer earnings data

More than 20GB of Intel source code and proprietary data dumped online

"Exconfidential Lake" leak includes docs Intel provided under NDA as recently as May.
DAN CODDIN AND JIM SALTER - AUG 6, 2020 4:59 PM 150

Toyota source code exposed for five years, impacts 300,000 drivers

Dashveenjit Kaur October 13, 2022

Mercedes-Benz Source Code at Risk: GitHub Token Mishap Sparks Major Security Concerns

Lohit January 29, 2024



More incidents...

Samsung and Nvidia are the latest companies to involuntarily go open-source leaking company secrets



MACKENZIE JACKSON
9 MAR 2022 · 4 MIN READ

Follow us on



The Microsoft source code breach may be much bigger than we thought

News

By Anthony Spadafora published March 22, 2022

Slack employee tokens stolen, GitHub repository breached

Published Jan. 5, 2023

NEWS

2 NOV 2022

Dropbox Suffers Breach, 130 GitHub Repositories Compromised

tj-actions changed-files through 45.0.7 allows remote attackers to discover secrets by reading actions logs.

High severity

GitHub Reviewed

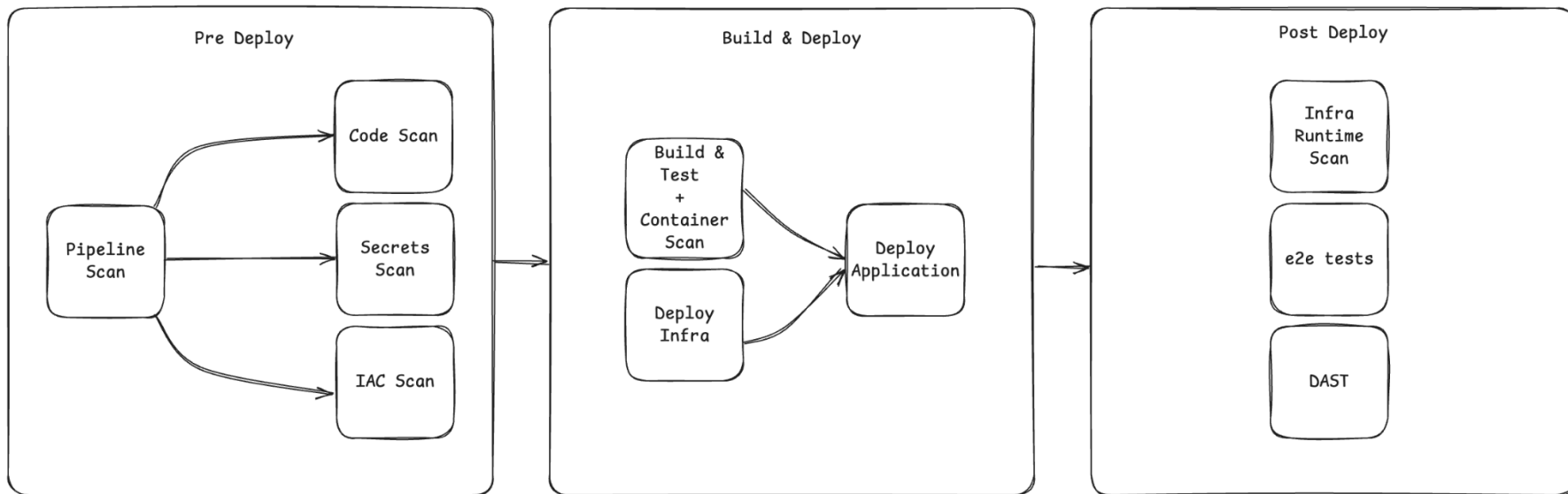
Published on Mar 14 to the GitHub Advisory Database · Updated on Mar 24

NPM BREACH EXPOSES TOPTAL GITHUB REPOSITORIES IN MAJOR JULY 2025 ATTACK

by CertPro Digest | Jul 24, 2025 | Data |



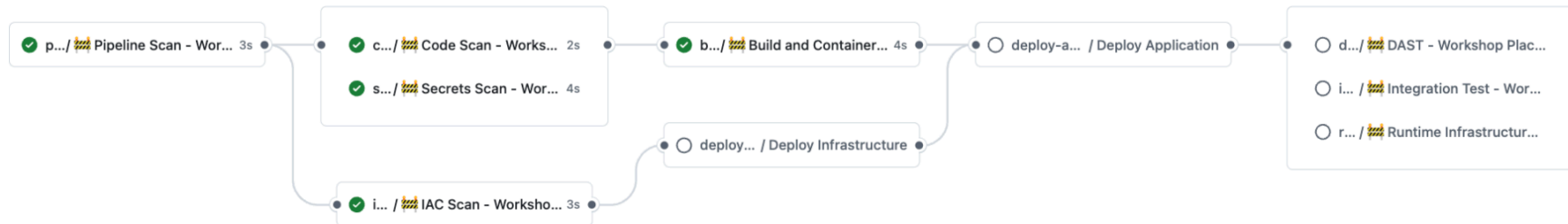
The "perfect pipeline" overview



The "perfect pipeline" overview

pipeline-orchestrator.yml

on: pull_request



Each job is a reusable workflow orchestrated by `pipeline-orchestrator.yml`.



WHERE IS THE AI?!?

There ~~is no~~ **is** AI today...

A year ago, these problems kept AI out of this talk. They haven't gone away. The industry just got much better at handling them:

- **Non-determinism** — same prompt, same code, different findings
- **Cost & rate limits** — an LLM on every commit adds up fast
- **Privacy** — your proprietary code travelling to a third-party API
- **Auditability & explainability** — *"why did you flag this?"* is still opaque
- **Prompt injection** — the code under review **is** the input



Two faces of AI in pipelines

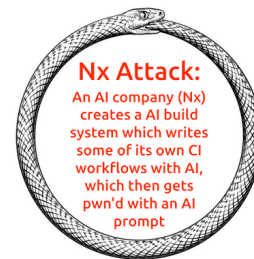
Defenders aren't the only ones shipping AI.

● **AI as the attacker** — *exploitation at machine scale*

- Autonomous bots exploiting CI/CD — *Hackerbot-Claw*
- AI reviews missing real attacks — *Nx supply chain*

● **AI as the defender** — *patterns shipping in production*

- Validated auto-fix — *Snyk, Copilot Autofix*
- AI triage on deterministic findings — *Semgrep Assistant, Aikido*
- Fine-tuned SLMs for narrow tasks
- Autonomous AI pentesting — *XBow, Aikido*



Adan Álvarez Vilchez • 1st
Principal Security Engineer | AWS Community Builder | Building Trust. Re...
3d · 🌐

Did an AI code review miss the injection vulnerability that led to the NX supply chain attack?

Analyzing the root cause of the NX incident, I immediately thought about the topic I've been discussing lately: LLMs for code reviews.

If I'm not mistaken, this PR introduced the injection vulnerability:
- <https://lnkd.in/dNgvN59J>

The PR seems to have gone through an "AI code review" but no problems were detected.

I did a quick test running this code through Semgrep GitHub Action rules and Semgrep rules do seem to catch it
- <https://lnkd.in/dnUQVeU2>

Interestingly, in the results from my tests, LLMs seemed to perform worse when analyzing things like GitHub Actions:
- <https://lnkd.in/dBm4GNeX>

As I've mentioned in previous posts, this is not about dismissing LLMs. They are powerful and improving quickly. What I keep seeing is that combining static analysis with LLMs feels like a stronger approach than relying on just one.



Where AI actually helps

A few places where it pulls its weight.

3:40 PM
40min

When LLMs Summarize Security Findings: The Tradeoffs You Can't Ignore
 Andrey Lukashenkov
AI Security Village

Axis	Scanner sees	AI sees
 Semantic Access control — IDOR, mass assignment, BOLA...	<code>user.update(req.body)</code> — valid ORM call, no SQLi, no taint sink.	"The body is unfiltered — a user can send <code>{is_admin: true}</code> and promote themselves. Same family as IDOR: scanners can't tell which fields are sensitive."
 Knowledge Up-to-date best practice	Even great scanners depend on someone keeping the ruleset current.	New patterns appear every month — <code>jwt alg: none</code> , lockfile poisoning, MCP misconfigs. The model has them baked in; your config probably doesn't.
 Explanation The "why" + a diff	<code>js/sql-injection at line 42</code> . Severity high. Now figure it out.	"Line 42 concatenates <code>req.body.email</code> . Switch to a parameterised version — here's the diff. Won't break <code>db.js</code> ."



Risks of running AI in your pipeline

And the new attack surfaces it brings:

- **⚠ Prompt injection** — OWASP LLM #1 · structural · every input is a vector
- **⚠ Slopsquatting** — ~20% of LLM-recommended packages don't exist
- **⚠ Secrets in prompts** — provider logs become an exfil channel
- **⚠ MCP attack surface** — 12 threat categories catalogued (CoSAI, Jan 2026)
- **⚠ AI step = supply chain** — pin SHAs · audit · just another third-party Action



Putting it together

Before you wire it up:



Do

- Pair AI with deterministic findings
- Treat findings as triage
- Log prompt + response on every run
- Audit the AI Action



Don't

- Block merges on AI findings alone
- Send proprietary code without evaluation
- Skip prompt-injection defences
- Auto-merge AI suggestions blindly

Baseline · OWASP Top 10 for LLM Applications



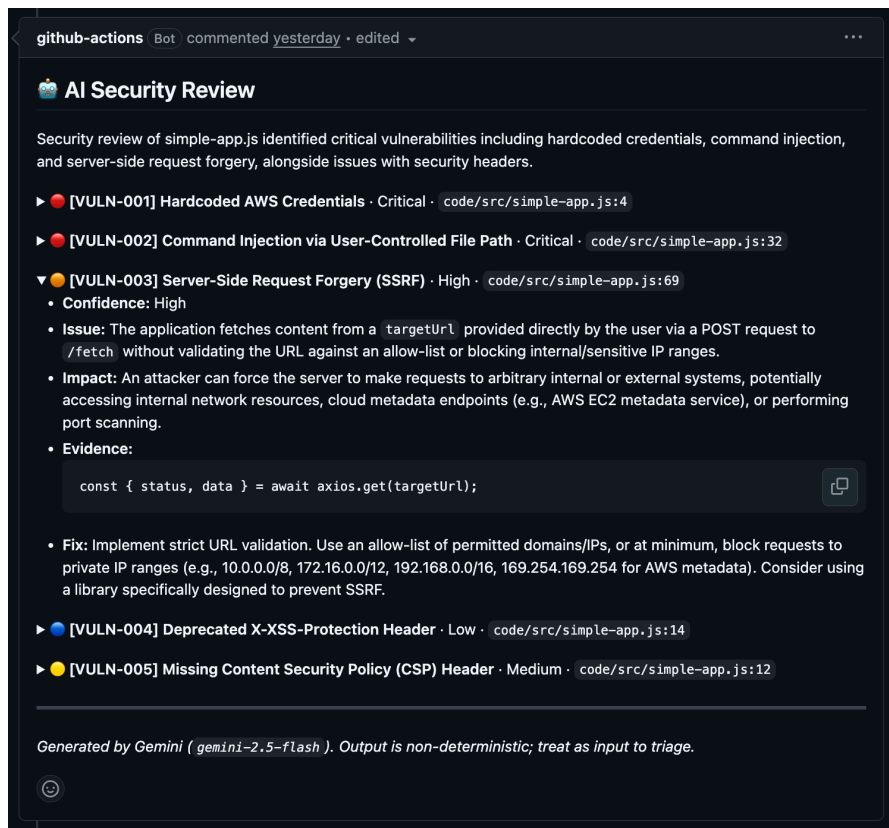
AI Example — Module 7

Workflow

- Open a PR → AI review runs automatically
- Sticky comment with structured findings: *severity · file:line · evidence · fix*
- Re-runs update the same comment in place

Backed by Gemini 2.5 Flash

- Free tier · no credit card · 250 req/day
- Model-agnostic pattern: swap for Claude, GPT, Llama via Ollama...



Important stuff that we've left out...

- pre-commit and other git hooks
- Github features deep dive
- Red Team tooling
 - Like [boostsecurityio/smokedmeat](https://github.com/boostsecurityio/smokedmeat) 
- A lot of very good tools...
- Any new vector attack that happened in the last 24h...



Before the Hands-On...

Questions?

How are we going to run the workshop?

Fork → <https://github.com/unicrons/secure-pipeline-workshop>



Setup (once)

1. Create a `workshop` branch off `main` (tip: press `.` → github.dev)
2. Enable Github Actions on the fork
3. After the first push, open **one** PR: `workshop` → your fork's `main`
4. Every later push updates that same PR

Per module

1. Brief step explanation
2. Replace the `jobs:` block in `.github/workflows/<module>-scan.yml`
3. Push → PR re-runs → fix → push again until
4. Results, questions, next module

⚠ Some modules need extra secrets/vars (`NVD_API_KEY` , `AWS_IAM_ROLE_ARN` , `GEMINI_API_KEY`) — see the repo's *Prerequisites*. Optional: enable **GitHub**

Advanced Security on your fork for SARIF in the *Code Scanning* tab.



Workshop: Pipeline Security

Why is Pipeline Security Important?

- Baseline security of our SDLC (Software Development Life Cycle)
- Elevated access:
 - to secrets
 - to code
 - to infra



Common Pipeline Security Issues

- Hardcoded Secrets
- Excessive Permissions
- Untrusted Actions
- Insecure Triggers
- Missing Security Controls



Beyond SHA Pinning:
Security for CI/CD Pipelines



CfgMgmtCamp 2026 ~ Ghent



unicrons.cloud



Workshop: Pipeline Security

Extra ball: **Warden's Ruse**

```
.../SOLVED/wardens/Wardens-Ruse-main
ls -laR
drwxrwxr-x@ - Andoni.Alonso 15 jul 14:04 .github
.rw-rw-r--@ 1,4k Andoni.Alonso 15 jul 14:04 main.tf
.rw-rw-r--@ 286 Andoni.Alonso 15 jul 14:04 README.md
.rw-rw-r--@ 174k Andoni.Alonso 15 jul 14:04 repo-visibility
.rw-rw-r--@ 465k Andoni.Alonso 15 jul 14:04 Warden.png

./github:
drwxrwxr-x@ - Andoni.Alonso 15 jul 14:04 workflows

./github/workflows:
.rw-rw-r--@ 829 Andoni.Alonso 10 ago 21:56 apply-prod.yaml

.../SOLVED/wardens/Wardens-Ruse-main
cat README.md | grep TODO
TODO: Lock up once Haisha finishes setting up dc32-wardens-treasure-prod
and <https://d2azf0l1i0s26w.cloudfront.net/>.
```

Challenge

7 Solves



Warden's Ruse

540

In the enchanted realm of Cybershire, a powerful artifact lies hidden within the mystical Vault of Secrets. This vault, guarded by the vigilant Warden, is fortified with ancient spells, allowing only the chosen ones to pass. However, a flaw in the Warden's enchantment whispers through the winds. Brave adventurers, equipped with their wits and determination, must decipher the cryptic trust sigils, bypass the Warden's watchful eye, and retrieve the secret from Vault. Only the most cunning will succeed in this quest.



Workshop: Pipeline Security

Apparently to be used for deploys, using Github Actions

Something suspicious?



```
resource "aws_iam_role" "warden" {
  name       = "warden-production"
  assume_role_policy = data.aws_iam_policy_document.warden-role.json
}

data "aws_iam_policy_document" "warden-role" {
  statement {
    actions = ["sts:AssumeRoleWithWebIdentity"]

    principals {
      identifiers = [aws_iam_openid_connect_provider.github.arn]
      type       = "Federated"
    }
  }

  condition {
    test      = "StringEquals"
    values    = ["sts.amazonaws.com"]
    variable  = "token.actions.githubusercontent.com:aud"
  }

  condition {
    test      = "StringLike"
    values    = ["repo:*/Wardens-Ruse:ref:refs/heads/endlessendurance"]
    variable  = "token.actions.githubusercontent.com:sub"
  }

  condition {
    test      = "StringEquals"
    values    = ["private"]
    variable  = "token.actions.githubusercontent.com:repository_visibility"
  }
}
```



```
condition {  
  test      = "StringEquals"  
  values    = ["sts.amazonaws.com"]  
  variable  = "token.actions.githubusercontent.com:aud"  
}
```

```
condition {  
  test      = "StringLike"  
  values    = ["repo:*/Wardens-Ruse:ref:refs/heads/endlessendurance"]  
  variable  = "token.actions.githubusercontent.com:sub"  
}
```

```
condition {  
  test      = "StringEquals"  
  values    = ["private"]  
  variable  = "token.actions.githubusercontent.com:repository_visibility"  
}
```

```
}
```

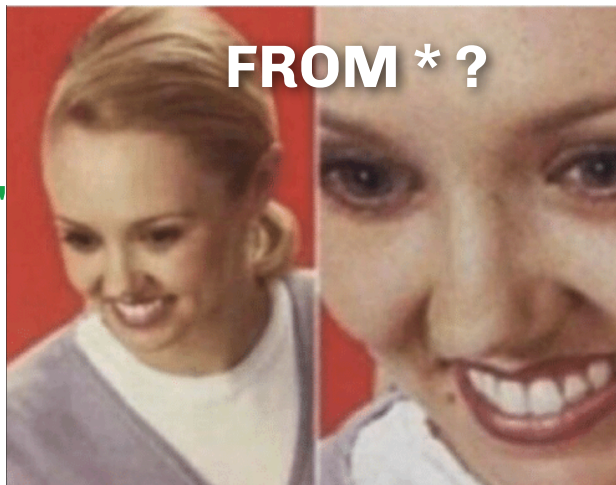
```
}
```


Workshop: Pipeline Security

```
condition {  
  test      = "StringLike"  
  values    = [ "repo:*/Wardens-Ruse:ref:refs/heads/endlessendurance" ]  
  variable = "token.actions.githubusercontent.com:sub"  
}
```

A repository

from "*"



branch "endlessendurance"



Workshop: Pipeline Security

Enable Pipeline Security Step

1. Go to `workshop/pipeline_scan`
2. Choose: `zizmor` or `claws`
3. Replace the `jobs:` block in `.github/workflows/pipeline-scan.yml` with the tool's `workflow.yml`
4. Push to your `workshop` branch — open the PR against your fork's `main` (only the first time)
5. Wait for the pipeline results
6. Try to solve them!

(Don't worry, we will show the results later.)



Workshop: Pipeline Security (Results)

Zizmor

unpinned action reference

 In pull request in `refs/pull/2/merge` 5 days ago

```
.github/workflows/pipeline-scan.yml:25
22     persist-credentials: false
23
24     - name: Run zizmor
25       uses: zizmorcore/zizmor-action@main
      unpinned action reference
      zizmor
26       with:
27         min-severity: "high"
```

steps:

- name: Checkout repository
uses: actions/checkout@11bd71901bbe5b1630ceea73d27597364c9af683 # v4.2.2
with:
 persist-credentials: false
- name: Run zizmor
uses: zizmorcore/zizmor-action@f52a838cfabf134edcbaa7c8b3677dde20045018 # v0.1.1
with:
 min-severity: "high"

Claws

 Running Claws security analysis...

Violation: UnpinnedAction on .github/workflows/pipeline-scan.yml:23

All reusable actions must be pinned to a full sha1 commit hash.

For more information:

<https://github.com/betterment/claws/blob/main/README.md#unpinnedaction>

```
- name: Set Up Ruby
>>>   uses: ruby/setup-ruby@master
      with:
        ruby-version: '3.3'
- name: Set Up Claws Config
```

- name: Set Up Ruby
uses: ruby/setup-ruby@2a7b30092b0caf9c046252510f9273b4875f3db9 #v1.254.0
with:
 ruby-version: '3.3'



Workshop: Pipeline Security

Extra ball: **Warden's Ruse**



```
./workshop/pipeline_scan/extra/WardenRuseReturns
```

This time the Admin has been studying the Github Actions OIDC token and has added some new conditions to the policy to make it more secure...

Hint: AWS does not support all claims that GitHub Actions OIDC token provides. You may need to customize the subject claim:

```
gh api /repos/<org>/secure-pipeline-workshop/actions/oidc/customization/sub -X PUT --input claims.json
```



Workshop: Pipeline Security

Extra ball: `actions/checkout` can leak Github Credentials

`./workshop/pipeline_scan/extra/checkoutLeak`

pipeline-scan / actions/checkout can leak GitHub credentials

succeeded 1 minute ago in 3s

> ✓ Set up job

> ✓ Checkout code

✓ Show ``.git/config`` content

```
1 ▶ Run cat .git/config
4 [core]
5   repositoryformatversion = 0
6   filemode = true
7   bare = false
8   logallrefupdates = true
9 [remote "origin"]
10  url = https://github.com/unicrons/github-actions-playground
11  fetch = +refs/heads/*:refs/remotes/origin/*
12 [gc]
13  auto = 0
14 [http "https://github.com/"]
15  extraheader = AUTHORIZATION: basic ***
```

> ✓ Post Checkout code

> ✓ Complete job

Repositories owned by **microsoft**
that depend on **actions/checkout**

 **1,907 Repositories**  0 Packages

Repositories owned by **coinbase** that Clear filter
depend on **actions/checkout**

 **44 Repositories**  0 Packages 

Repositories that depend on **actions/checkout**

 **14,601,495 Repositories**  0 Packages 



Workshop: Pipeline Security

Extra ball: Prowler Github Scanning

```
./workshop/pipeline_scan/extra/prowler
```



</

Showing 1 to 3 of 3 entries (filtered from 16 total entries)

Previous 1 Next

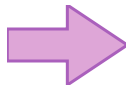


Questions?

Workshop: Secrets Scan

Why is Secrets Scan Important?

- Data breaches
- Access to our systems
- Supply Chain Attacks



- Compliance Issues
- Financial Loss
- Reputation Damage

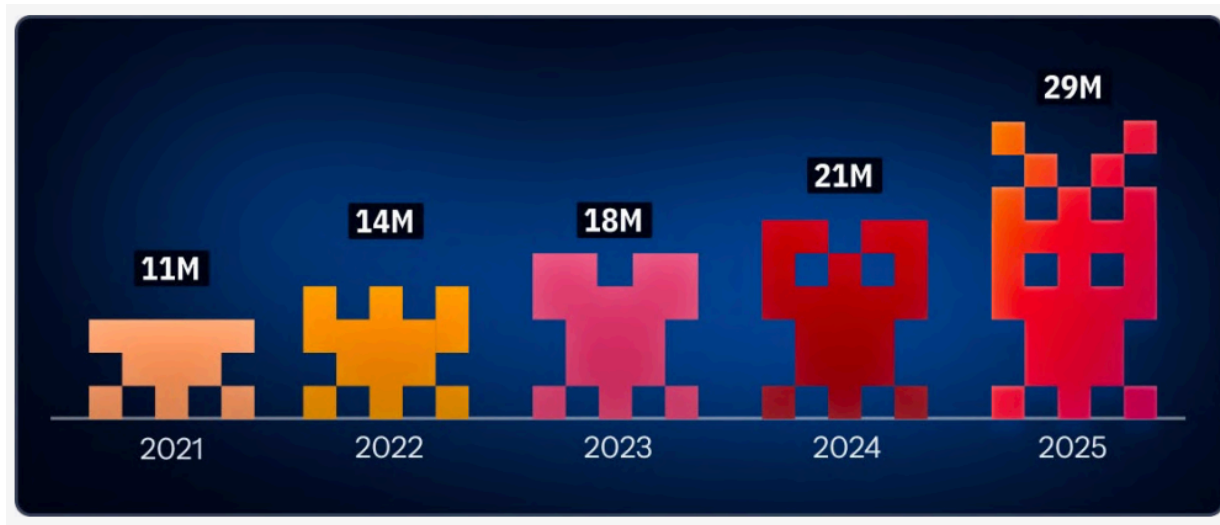
Common Leaked Secrets

- Hardcoded passwords
- Database Connection Strings
- Cloud Provider Keys
- 3rd Party API Keys (AI providers, payments, SaaS...)
- Private keys
- Service account & non-human identity (NHI) tokens



Workshop: Secrets Scan

The problem keeps growing — fast.



28.65M

new hardcoded secrets in public GitHub commits in 2025

+34% YoY

Largest single-year jump GitGuardian has ever recorded. From 11M in 2021 to ~29M in 2025.



Workshop: Secrets Scan — AI is fueling a new generation of leaks



2× — Claude Code co-authored commits leak secrets at twice the human baseline (3.2% vs 1.5%).

+81% — YoY growth in AI-services secrets (1.27M in 2025).

5× — LLM infrastructure (RAG, vector stores, orchestration) leaks faster than core model providers.

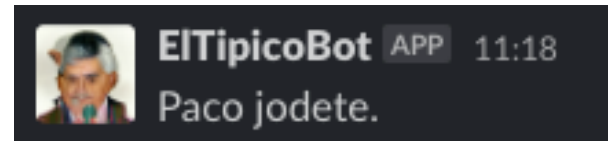
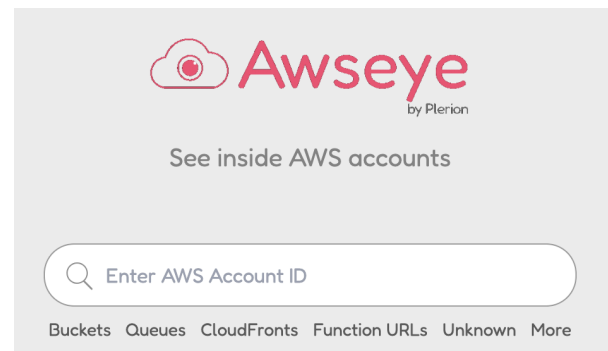
24K+ — secrets exposed in MCP configuration files in their first year.



Workshop: Secrets Scan

Other Types of Sensitive Data

- **Webhooks**
 - Expose internal domains
 - Trigger actions (like Slack Webhooks)
- **Account IDs**
 - Easier enumeration
- **Internal Documents**
- **PII**



Awseye is an open-source intelligence (OSINT) and reconnaissance service tracking publicly accessible AWS data.



Workshop: Secrets Scan

Privacy is *not* a security control

6×

Internal repos are **6×** more likely to contain a hardcoded secret than public ones (32.2% vs 5.6%).

64%

Of secrets leaked in **2022** are **still valid** in 2026. Detection isn't the bottleneck — remediation is.

80K

Secrets found on **unintentionally exposed self-hosted GitLab + Docker registries** — 10K still valid.

Recent reminders that "private" doesn't mean "safe":

- **Red Hat consulting breach** (Oct 2025): 570GB exfiltrated from internal GitLab, 28K repos, ~800 orgs affected.

- **Shai-Hulud 2** (npm supply chain): 6,943 dev workstations harvested, ~33K unique secrets — secret avg appears in 8 places per machine.



Workshop: Secrets Scan

Enable Secrets Scan step

1. Go to `workshop/secrets_scan`
2. Choose: `trufflehog` or `gitleaks`
3. Replace the `jobs:` block in `.github/workflows/secrets-scan.yml` with the tool's `workflow.yml`
4. Push to your `workshop` branch — your existing PR re-runs automatically
5. Wait for the pipeline results
6. Try to solve them!

(Don't worry, we will show the results later.)



Workshop: Secrets Detection (Results)

Trufflehog

✅ Found verified result 🦊🔑

Detector Type: AWS

Decoder Type: PLAIN

Raw result: AKIA2T2SJH6MS337PDWL

Resource_type: Access key

Account: 729780141977

Arn: arn:aws:iam::729780141977:user/canarytokens.com@@xom68iew4t07umwp9nj01uy4s

Rotation_guide: <https://howtorotate.com/docs/tutorials/aws/>

User_id: AIDA2T2SJH6M50CWJQ6T

File: code/src/simple-app.js

Line: 3

Gitleaks

```
Finding:    const AWS_SECRET_ACCESS_KEY = 'oMKFrMwcYIJB/PU7l2E0G8wg9K0fQapwVKGP4HaD'  
Secret:     oMKFrMwcYIJB/PU7l2E0G8wg9K0fQapwVKGP4HaD  
RuleID:     generic-api-key  
Entropy:    4.834184  
File:       code/src/simple-app.js  
Line:       4  
Fingerprint: code/src/simple-app.js:generic-api-key:4
```

```
Finding:    ...WS_ACCESS_KEY_ID = 'AKIA2T2SJH6MS337PDWL  
Secret:     AKIA2T2SJH6MS337PDWL  
RuleID:     aws-access-token  
Entropy:    3.921928  
File:       code/src/simple-app.js  
Line:       3  
Fingerprint: code/src/simple-app.js:aws-access-token:3
```



Questions?

Workshop: Code Scan

Why is **SAST** Code Scan Important?

- Vulnerability detection in your source code logic
- Faster feedback loop compared to DAST
- Understands vulnerabilities with code context, not only matching regexp



Common code security issues

- RCE (Remote code execution)
- Secrets leakage
- DoS (Denial of Service)
- SQL Injection



Workshop: Code Scan

Why is **SCA** Code Scan Important?

- License Risk Assessment
- Outdated/Vulnerable Component Detection
- Helps you build a Software Bill of Materials (SBOM)
- Mitigate (detect) supply chain attacks

Common SCA findings can cause:

- License issues
- Vulnerabilities in your dependencies
- Executing untrusted code



Workshop: Code Scan

Enable Code Scan step

1. Go to `workshop/code_scan`
2. Choose one **SAST** tool as a job: `CodeQL` or `semgrep`
3. Add the **SCA** tool as a job: `Dependency Check` or `OSV Scanner`
4. Replace the `jobs:` block in `.github/workflows/code-scan.yml` with both jobs
5. Push to your `workshop` branch — your existing PR re-runs automatically
6. Wait for the pipeline results
7. Try to solve them!

(Don't worry, we will show the results later.)

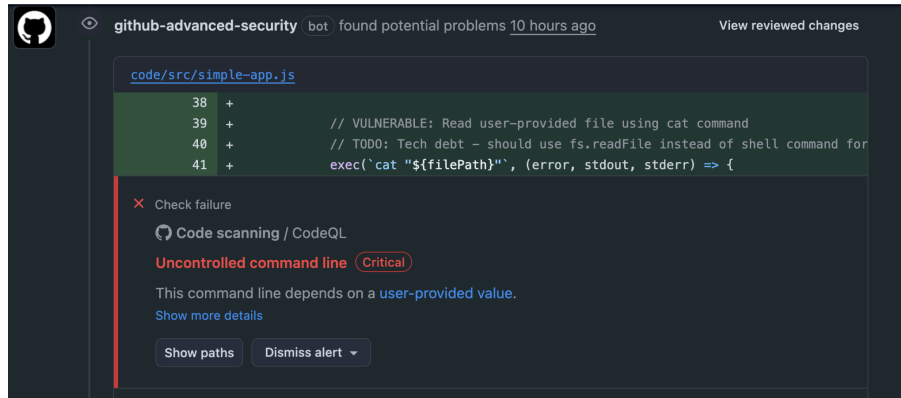
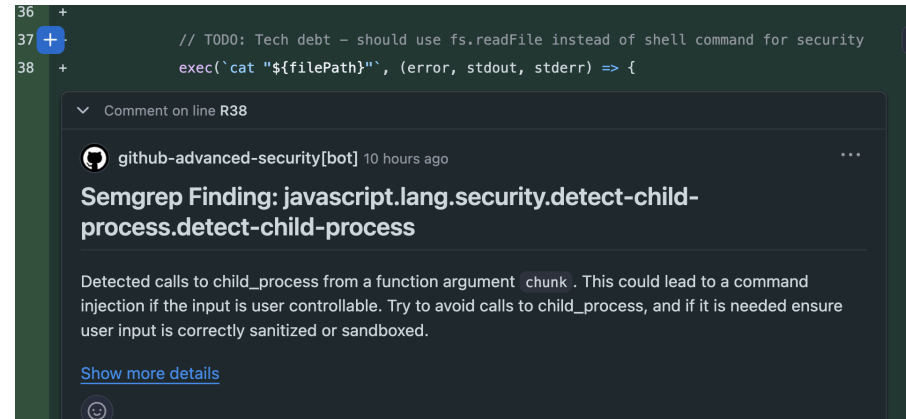
💡 Optional: set `NVD_API_KEY` in *Settings* → *Secrets* to speed up Dependency Check.



Workshop: Code Scan (Results SAST)

CodeQL

semgrep



Workshop: Code Scan (Results SCA)

Dependency Check

```
Error:
One or more dependencies were identified with vulnerabilities that have a CVSS score greater than or equal to '7.0':
package-lock.json?axios (pkg:npm/axios@1.6.0): GHSA-4hjh-wcwx-xvwj(7.5), GHSA-jr5f-v2jv-69x6(10.0), GHSA-8hc4-vh64-cxmj(10.0), GHSA-43fc-jf86-j433(7.5)
See the dependency-check report for more details.
```

OSV Scanner

```
▶ Run COUNT=$(jq '[.runs[]|results[]] | length' osv-scanner.sarif)
✗ Found 6 vulnerability finding(s):
- CVE-2025-62718 | Package 'axios@1.6.0' is vulnerable to 'CVE-2025-62718' (also known as 'GHSA-3p68-rc4w-qgx5').
  code/package-lock.json:1
- CVE-2026-25639 | Package 'axios@1.6.0' is vulnerable to 'CVE-2026-25639' (also known as 'GHSA-43fc-jf86-j433').
  code/package-lock.json:1
- CVE-2025-58754 | Package 'axios@1.6.0' is vulnerable to 'CVE-2025-58754' (also known as 'GHSA-4hjh-wcwx-xvwj').
  code/package-lock.json:1
- CVE-2024-39338 | Package 'axios@1.6.0' is vulnerable to 'CVE-2024-39338' (also known as 'GHSA-8hc4-vh64-cxmj').
  code/package-lock.json:1
- CVE-2026-40175 | Package 'axios@1.6.0' is vulnerable to 'CVE-2026-40175' (also known as 'GHSA-fvcv-3m26-pcqx').
  code/package-lock.json:1
- CVE-2025-27152 | Package 'axios@1.6.0' is vulnerable to 'CVE-2025-27152' (also known as 'GHSA-jr5f-v2jv-69x6').
  code/package-lock.json:1
```



Workshop: Code Scan (Results SCA)



Dependency-Check is an open source tool performing a best effort analysis of 3rd party dependencies; false positives and false negatives may exist in the analysis performed by the tool. Use of the tool and the reporting provided constitutes acceptance for use in an AS IS condition, and there are NO warranties, implied or otherwise, with regard to the analysis or its use. Any use of the tool and the reporting provided is at the user's risk. In no event shall the copyright holder or OWAS

[How to read the report](#) | [Suppressing false positives](#) | [Getting Help: github issues](#)

Project: code

Scan Information:

- *dependency-check version:* 12.2.2
- *Report Generated On:* Sun, 3 May 2026 18:27:15 GMT
- *Dependencies Scanned:* 2 (2 unique)
- *Vulnerable Dependencies:* 1
- *Vulnerabilities Found:* 6
- *Vulnerabilities Suppressed:* 0

Summary

Dependency	Vulnerability IDs	Package	Highest Severity	CVE Count	Confidence	Evidence Count
axios:1.6.0		pkg:npm/axios@1.6.0	HIGH	6		3

Dependencies

axios:1.6.0

File Path: /github/workspace/code/package-lock.json?axios

Referenced In Project/Scope: package-lock.json: transitive

Identifiers

- pkg:npm/axios@1.6.0 (Confidence: Highest)

Check the **Upload Scan Report** step in the SCA Job to check the HTML report with deeper info on the vulnerabilities.



Questions?

Workshop: Infra as Code Scan

Why is IaC Scan Important?

- Identify misconfigurations before deployments
- Treat IaC as any other piece of software
- Scalable security approach for the Cloud



This is supposed to be us creating "infra as code".



Workshop: Infra as Code Scan

Common IaC Issues

- **Access Control**
 - Overly Permissive IAM Policies, Publicly Accessible Resources, Missing Authentication Controls, Default or Weak Credentials
- **Encryption**
 - Unencrypted Storage at Rest, Unencrypted Backups & Snapshots, Missing TLS/In-Transit Encryption, Weak or Outdated Cipher Suites
- **Network Security**
 - Open Security Groups (0.0.0.0/0), Public-Subnet Exposure, Mis-scoped Load Balancers/Endpoints
- **Compliance & Governance**
 - Insufficient Logging & Audit Trails, Lack of Continuous Monitoring/Alerts, Missing Resource Tagging, Improper Backup Retention/Encryption



Workshop: Infra as Code Scan

Specific IaC Recommendations

- Use encrypted remote backends
- Use state locking to prevent concurrent modifications
- Mark sensitive data appropriately
- Pin provider versions
- Vet third-party modules

We recommend running IaC scans against `terraform plan` and equivalents when possible for more accurate results.



Workshop: Infra as Code Scan

Extra ball: **Terraform Plan RCE**

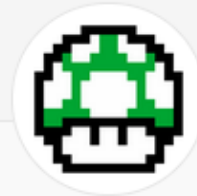
<https://alex.kaskaso.li/post/terraform-plan-rce>

Terraform and external programs. You can use the `external` data source to run arbitrary code during a `plan`.

```
data "external" "example" {
  program = ["python", "${path.module}/example-data-source.py"]

  query = {
    # arbitrary map from strings to strings, passed
    # to the external program as the data query.
    id = "abc123"
  }
}
```

The `query` will be passed as a JSON string on `stdin` to the `program`; you could use this to grab variables from Terraform.



Terraform Plan RCE

11 May 2021 • Written by alxk



Workshop: Infra as Code Scan

Enable IaC Scan step

1. Go to `workshop/iac_scan`
2. Choose: `trivy` or `checkov`
3. Replace the `jobs:` block in `.github/workflows/iac-scan.yml` with the tool's `workflow.yml`
4. Push to your `workshop` branch — your existing PR re-runs automatically
5. Wait for the pipeline results
6. Try to solve them!

(Don't worry, we will show the results later.)



Workshop: Infra as Code Scan (Results)

Checkov

```
Ensure no security groups allow ingress from 0.0.0.0/0 to port 22 #267
Open In pull request refs/pull/26/merge · 4 minutes ago

infra/main.tf:115
112 # Note from the SME team: if you need to reference the VPC CIDR use this data:
113 #   data.aws_vpc.existing.cidr_block
114
115 resource "aws_security_group" "ecs_tasks" {
116   name_prefix = "${var.app_name}-ecs-"
117   description = "Security group for ECS tasks"
118   vpc_id      = data.aws_vpc.existing.id
119
120   ingress {
121     description = "HTTP from ALB"
122     from_port   = 22
123     to_port     = 22
124     protocol    = "tcp"
125     cidr_blocks = ["0.0.0.0/0"]
126   }
127
128   # trivy:ignore:AW0-010
129   egress {
130     description = "HTTPS outbound"
131     from_port   = 443
132     to_port     = 443
133     protocol    = "tcp"
134     cidr_blocks = ["0.0.0.0/0"]
135   }
136
137   # trivy:ignore:AW0-010
138   egress {
139     description = "HTTP outbound"
140     from_port   = 80
141     to_port     = 80
142     protocol    = "tcp"
143     cidr_blocks = ["0.0.0.0/0"]
144   }
145
146   tags = {
147     Name = "${var.app_name}-ecs-sg"
148   }
149 }

Error
Ensure no security groups allow ingress from 0.0.0.0/0 to port 22
Ensure no security groups allow ingress from 0.0.0.0/0 to port 22
checkov
150
151 #
152 # ECS CLUSTER, TASK DEFINITION, AND SERVICE
```

Trivy

Security groups should not allow unrestricted ingress to SSH or RDP from any IP address. #262

Open In pull request refs/pull/25/merge · 3 minutes ago

main.tf:125

Preview unavailable
Sorry, we couldn't find this file in the repository.

Error

Security groups should not allow unrestricted ingress to SSH or RDP from any IP address.

Artifact: main.tf
Type: terraform
Vulnerability AWS-0107
Severity: HIGH
Message: Security group rule allows unrestricted ingress from any IP address.
Link: [AWS-0107](#)

Trivy

Tool	Rule ID
Trivy	AWS-0107

Misconfiguration AWS-0107

Show more



Questions?

Workshop: Container Scan

Why is Container Scan Important?

- Common way to deploy software → Common target for attackers
- Access to underlying infra/secrets

Common Container Security Issues

- Vulnerable dependencies
- Hardcoded Secrets
- Excessive Privileges (root, network, filesystem)



Workshop: Container Scan

Enable Container Scan Step

1. Go to `workshop/container_scan`
2. Choose: `trivy` or `grype`
3. Replace the `jobs:` block in `.github/workflows/container-scan.yml` with the tool's `workflow.yml`
4. Push to your `workshop` branch — your existing PR re-runs automatically
5. Wait for the pipeline results
6. Try to solve them!

(Don't worry, we will show the results later.)



Workshop: Container Scan (Results)

Trivy

☐	22 Open	6 Closed	Language	Tool	Rule	Severity	Sort
☐	☒						
☒						High	refs/pull/27/merge
#295 opened 4 minutes ago • Detected by Trivy in sanchezpaco/secure-pipeline-workshop :1							
☒						High	refs/pull/27/merge
#294 opened 4 minutes ago • Detected by Trivy in sanchezpaco/secure-pipeline-workshop :1							
☒						High	refs/pull/27/merge
#292 opened 4 minutes ago • Detected by Trivy in sanchezpaco/secure-pipeline-workshop :1							
☒						High	refs/pull/27/merge
#291 opened 4 minutes ago • Detected by Trivy in sanchezpaco/secure-pipeline-workshop :1							
☒						Medium	Library
#304 opened 4 minutes ago • Detected by Trivy in app/.../axios/package.json:1							
☒						Medium	Library
#303 opened 4 minutes ago • Detected by Trivy in app/.../axios/package.json:1							
☒						Medium	refs/pull/27/merge
#297 opened 4 minutes ago • Detected by Trivy in sanchezpaco/secure-pipeline-workshop :1							
☒						Medium	refs/pull/27/merge
#296 opened 4 minutes ago • Detected by Trivy in sanchezpaco/secure-pipeline-workshop :1							

Grype

☐	47 Open	0 Closed	Language	Tool	Rule	Severity	Sort
☒						Critical	refs/pull/28/merge
#250 opened 2 days ago • Detected by Grype in secure-pipeline-workshop/.../bin/node :1							
☒						High	refs/pull/28/merge
#248 opened 2 days ago • Detected by Grype in secure-pipeline-workshop/.../db/installed:1							
☒						High	refs/pull/28/merge
#247 opened 2 days ago • Detected by Grype in secure-pipeline-workshop/.../db/installed:1							
☒						High	refs/pull/28/merge
#243 opened 2 days ago • Detected by Grype in secure-pipeline-workshop/.../bin/node :1							
☒						High	refs/pull/28/merge
#242 opened 2 days ago • Detected by Grype in secure-pipeline-workshop/.../bin/node :1							
☒						High	refs/pull/28/merge
#236 opened 2 days ago • Detected by Grype in secure-pipeline-workshop/.../bin/node :1							
☒						High	refs/pull/28/merge
#235 opened 2 days ago • Detected by Grype in secure-pipeline-workshop/.../db/installed:1							
☒						High	refs/pull/28/merge
#234 opened 2 days ago • Detected by Grype in secure-pipeline-workshop/.../db/installed:1							
☒						High	refs/pull/28/merge
#230 opened 2 days ago • Detected by Grype in secure-pipeline-workshop/.../db/installed:1							
☒						High	refs/pull/28/merge
#229 opened 2 days ago • Detected by Grype in secure-pipeline-workshop/.../db/installed:1							



Questions?

Workshop: Runtime Infra Scan

Why is Runtime Infra Scan Important?

- We can't catch everything in previous steps
- Manual changes exist
- It's the final source of truth

Common Runtime Infra Security Issues

- Access Control
- Encryption
- Network Security
- Compliance & Governance
- Drift



Workshop: Runtime Infra Scan

Enable Runtime Infra Scan Step

1. Go to `workshop/runtime_infra_scan`
2. Choose: `prowler` or `steampipe`
3. Replace the `jobs:` block in `.github/workflows/runtime-infra-scan.yml` with the tool's `workflow.yml`
4. Configure secrets/vars: `AWS_IAM_ROLE_ARN` (*secret*), `AWS_REGION` + `AWS_IAM_ROLE_SESSION_DURATION` (*vars*)
5. Push to your `workshop` branch — your existing PR re-runs automatically
6. Review the report (if you're using your own account 🤖)

⚠️ **No bait here.** Unlike the other modules, this one scans a **live AWS account** — the job stays green even when findings are reported. Just review the report.



Workshop: Runtime Infra Scan (Results)

Prowler

Steampipe

PROWLER

Report Information	AWS Assessment Summary	AWS Credentials	Assessment Overview
Version: 3.10.0	AWS Account: REDACTED	User Id: AR0AR042EVA5W6V45AANC:PowerSession	Total Findings: 99
Parameters used: aws --service iam s3 -> output-format.html 2	AWS-CLI Profile: default	Caller Identity ARN: arn:aws:sts:REDACTED:assumed-role:ProwlerRole:ProwlerSession	Passed: 62
Date: 2025-08-08T00:38:23.134664	Audited Regions: All Regions		Passed (Muted): 0
			Failed: 37
			Failed (Muted): 0
			Total Resources: 21
			Search: <input type="text"/>

Filters (2)		Show	100	entries										
Status	Severity	Service Name	Region	Check ID	Check Title	Resource ID	Resource Type	Status Extended	Risk	Recommendation	Compliance			
PASS	high	s3	us-east-1	s3_bucket_shadow_resource_vulnerability	Check for S3 buckets vulnerable to Shadow Resource Hijacking (Bucket Monopoly)	arn:aws:s3::warden-easter-egg-c4ba76z	S3	Bucket warden-easter-egg-c4ba76z is not a known shadow resource.	An attacker can pre-create S3 read more...	Ensure that all S3 buckets are read more...				
FAIL	medium	s3	us-east-1	s3_bucket_no_mfa_delete	Check if S3 Bucket MFA Delete is not enabled.	arn:aws:s3::warden-easter-egg-c4ba76z	S3	Bucket warden-easter-egg-c4ba76z has MFA Delete disabled.	Your security credentials are read more...	Adding MFA delete to an S3 bucket read more...	-MITRE-ATTACK: T1485 -Prowler read more...			
FAIL	low	s3	us-east-1	s3_bucket_lifecycle_enabled	Check if S3 buckets have a Lifecycle configuration enabled	arn:aws:s3::warden-easter-egg-c4ba76z	S3	Bucket warden-easter-egg-c4ba76z does not have a lifecycle configuration enabled.	The risks of not having lifecycle read more...	Enable lifecycle policies on S3 buckets read more...	-ISO27001-2022: A.8.10 -PCI read more...			
PASS	medium	s3	us-east-1	s3_bucket_acl_prohibited	Check if S3 buckets have ACLs enabled	arn:aws:s3::warden-easter-egg-c4ba76z	S3	Bucket warden-easter-egg-c4ba76z has Bucket ACLs disabled.	S3 ACLs are a legacy access control read more...	Ensure that S3 ACLs are disabled read more...	-PCI-4.0: 7.2.1.24, 7.2.2.24 read more...			
PASS	medium	s3	us-east-1	s3_bucket_default_encryption	Check if S3 buckets have default encryption (SSE) enabled or use a bucket policy to enforce it.	arn:aws:s3::warden-easter-egg-c4ba76z	S3	Bucket warden-easter-egg-c4ba76z has Server Side Encryption with AES256.	Amazon S3 default encryption p read more...	Ensure that S3 buckets have on read more...	-ISO27001-2022: A.8.11, A.8.2 read more...			
FAIL	medium	s3	us-east-1	s3_bucket_event_notifications_enabled	Check if S3 buckets have event notifications enabled.	arn:aws:s3::warden-easter-egg-c4ba76z	S3	Bucket warden-easter-egg-c4ba76z does not have event notifications enabled.	Without event notifications, I read more...	Enable event notifications for read more...	-PCI-4.0: 7.1.5.2.5, 7.6.1.5 read more...			
FAIL	medium	s3	us-east-1	s3_bucket_kms_encryption	Check if S3 buckets have KMS encryption enabled.	arn:aws:s3::warden-easter-egg-c4ba76z	Server Side Encryption	Server Side Encryption is not configured with kms for S3 Bucket warden-easter-egg-c4ba76z.	Amazon S3 KMS encryption provided read more...	Ensure that S3 buckets have on read more...	-ISO27001-2022: A.8.11, A.8.2 read more...			

1.5 Ensure MFA is enabled for the 'root' user account

The 'root' user account is the most privileged user in an AWS account. Multi-factor Authentication (MFA) adds an extra layer of protection on top of a username and password. With MFA enabled, when a user signs in to an AWS website, they will be prompted for their username and password as well as for an authentication code from their AWS MFA device.

OK	Skip	Info	Alarm	Error	Total
1	0	0	0	0	1

Reason		Dimensions
✓	MFA enabled for root account.	REDACTED

1.6 Ensure hardware MFA is enabled for the 'root' user account

The 'root' user account is the most privileged user in an AWS account. MFA adds an extra layer of protection on top of a user name and password. With MFA enabled, when a user signs in to an AWS website, they will be prompted for their user name and password as well as for an authentication code from their AWS MFA device. For Level 2, it is recommended that the root user account be protected with a hardware MFA.

OK	Skip	Info	Alarm	Error	Total
0	0	0	1	0	1

Reason		Dimensions
✗	MFA enabled for root account, but the MFA associated is a virtual device.	REDACTED

1.7 Eliminate use of the 'root' user for administrative and daily tasks

With the creation of an AWS account, a 'root user' is created that cannot be disabled or deleted. That user has unrestricted access to and control over all resources in the AWS account. It is highly recommended that the use of this account be avoided for everyday tasks.

OK	Skip	Info	Alarm	Error	Total
0	0	0	1	0	1

Reason		Dimensions
✗	Root password used 05-Aug-2025 (2 days). Access Key 1 never used. Access Key 2 never used.	REDACTED



Questions?

Thanks for coming!

Paco

 sanchezpaco

 pacosanchezlopez

Andoni

  andoniaf

 @andoniaf_

 andoniaf.unicrons.cloud

unicrons

 unicrons

 @unicrons_cloud

 unicrons.cloud



Feel free to contact us, or open an issue in the repo.
Any feedback would be greatly appreciated.

<https://github.com/unicrons/secure-pipeline-workshop>

Find this slides and other talks at
unicrons.cloud/en/talks

