

Secure by Default

**Building Confidence in
AI-Driven Delivery**



\$ whoami



Dewan Ahmed

Principal Developer Advocate



AI Native



Modular Platform

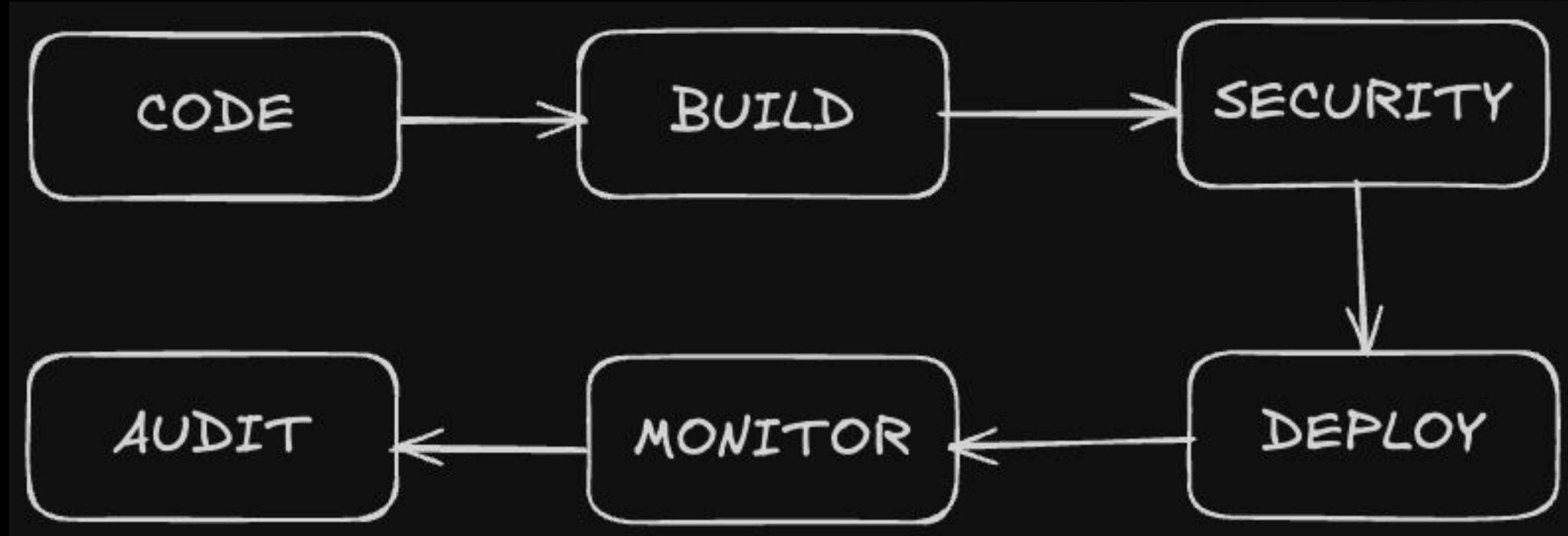


Developer First

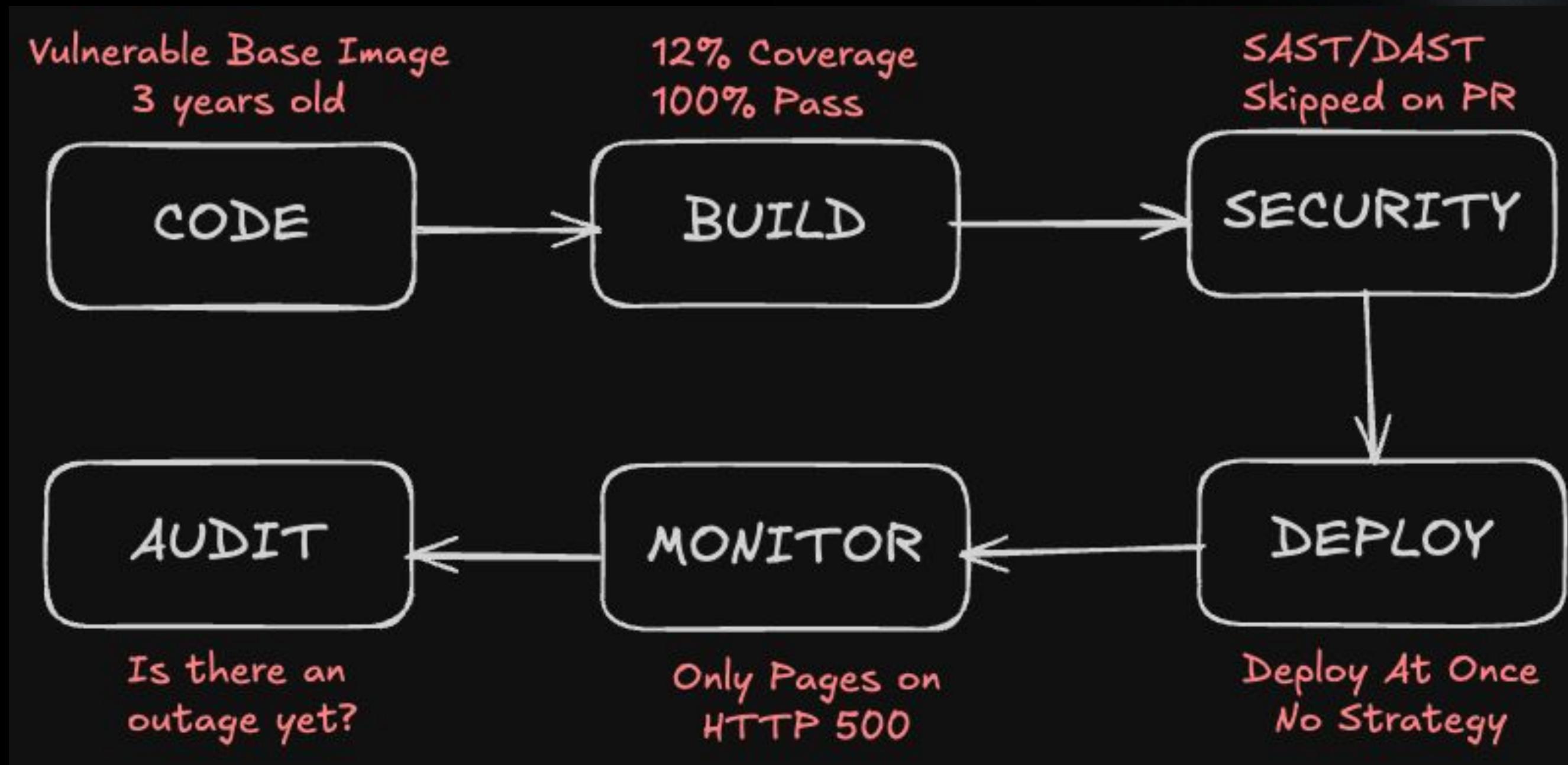


Embedded
Governance

What's stopping you from **DevSecOps-ing** like this?



What's stopping you from DevSecOps-ing like this?





We spent a decade automating delivery.

Then we added AI.

But we forgot to ask: what happens when
we automate **insecurity** at scale?

76%

of enterprises have already experienced an LLM prompt injection incident.

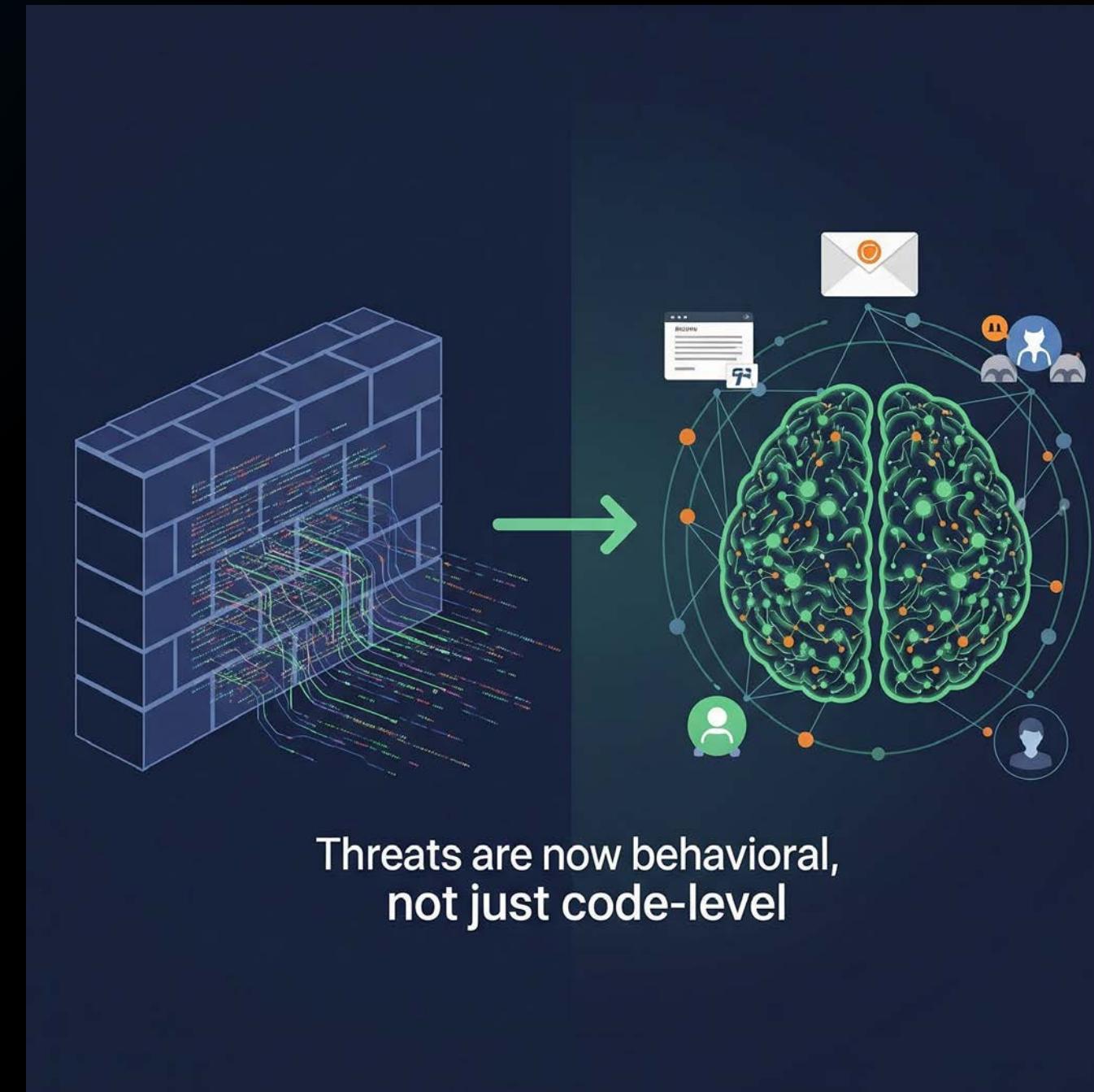
66%

have experienced an incident involving
vulnerable LLM code.

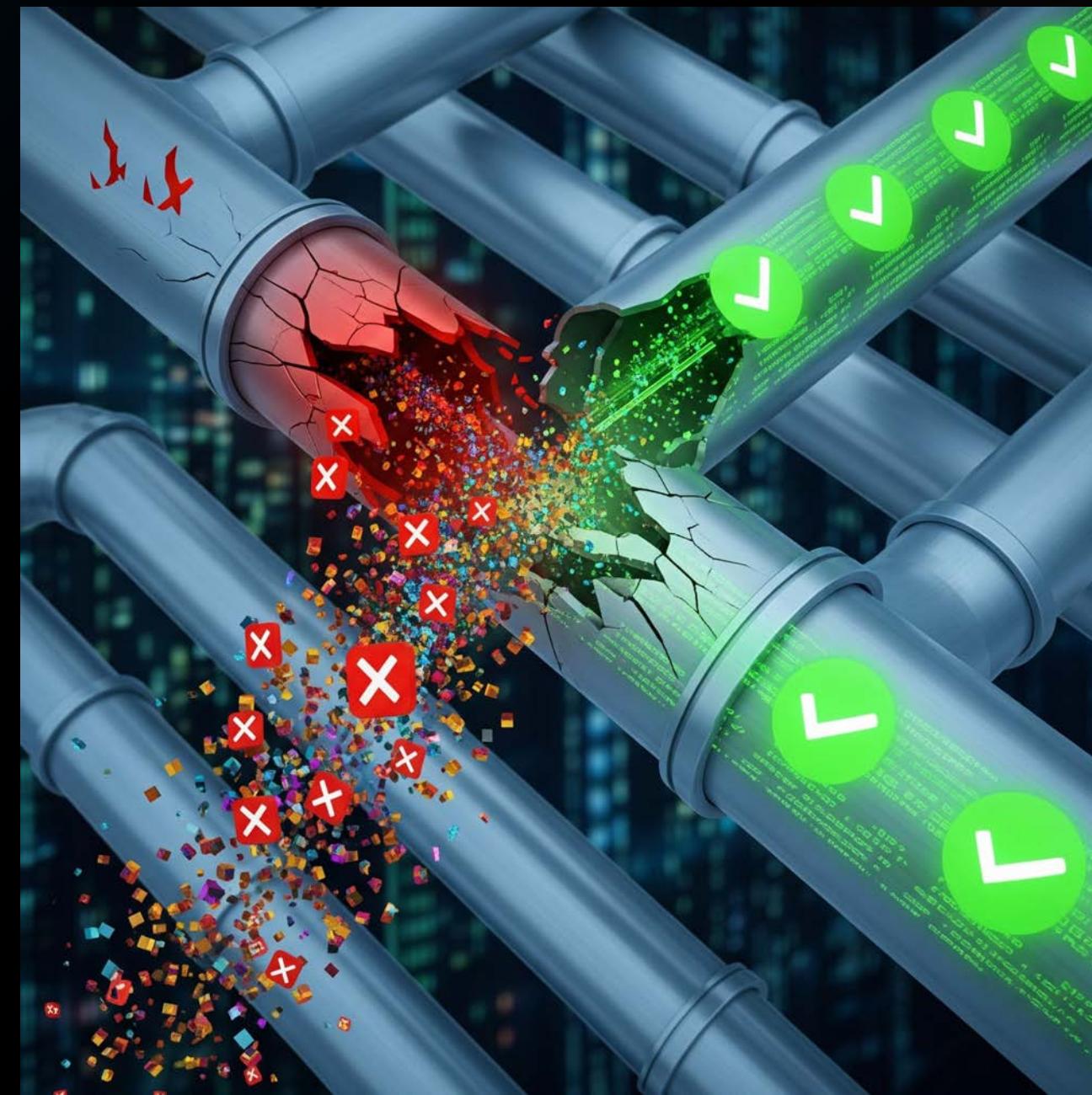
65%

have experienced LLM jailbreaking.

AI Has Changed Delivery. Threats Changed Faster.



Pipelines automate mistakes... as fast as they automate features



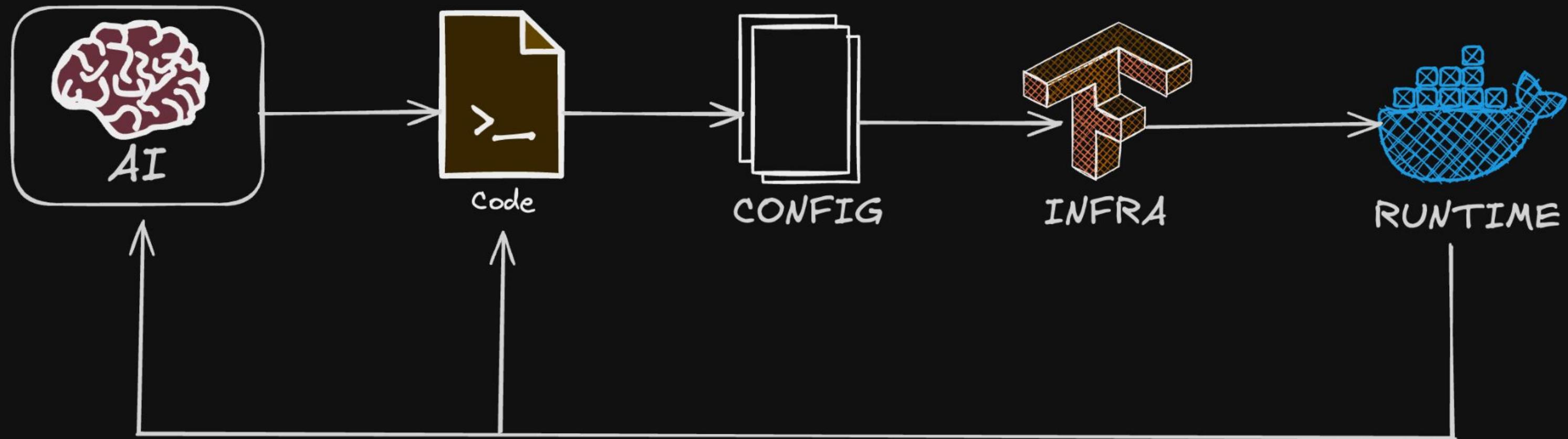
Teams are shipping AI-integrated systems... without new security patterns



Delivery Got Faster. Security Didn't Evolve.

- Delivery cycles shortened
- AI increased speed again
- Security practices stayed static
- Misconfigurations became “automated hazards”

AI-Native Delivery Lifecycle



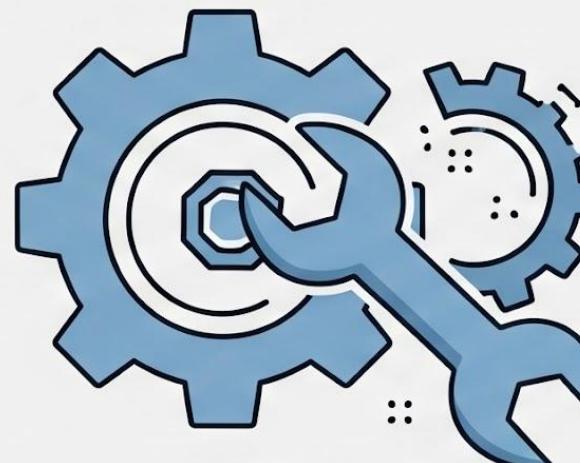
The New Attack Surface



PROMPT INJECTION



SUPPLY CHAIN



MISCONFIG



MODEL EXPLOITS

Developers Aren't Set Up For Success

43%

say developers
build security into
AI-native apps.

74%

see security as a
blocker.

62%

lack AI security
training.

What “Secure by Default” Actually Means



Safe defaults



**Automated
detection**



**Guardrails,
not gates**



**Intelligent
remediation**

How AI Can Fix the AI Problem

A pipeline that knows when something “looks wrong”

Understands provenance

Understands context

Understands behavioral drift

The Framework: Secure-by-Default DevSecOps



The Framework: Contextual intelligence



- Understands **what changed**, not just that something changed
- Connects code, config, infra, and model updates into **one unified story**
- Correlates deployment behavior with **historical patterns and risk signals**
- Detects when developer intent and system outcome **don't match**

The Framework: Automatic Verification



- Confirms artifact **provenance and integrity** before deployment
- Verifies config, environment, and infrastructure **match the source of truth**
- Ensures every pipeline step is **signed, trusted, and reproducible**
- Blocks or warns when deployments don't meet **predefined safety conditions**

The Framework: Behavior-Based Anomaly Detection



- Detects abnormal behavior across code, configs, infrastructure, and model outputs
- Identifies early indicators of drift, misconfigurations, or manipulation
- Learns normal delivery patterns and flags deviations in real time

The Framework: Continuous Learning Loops



- Improves security posture automatically based on **real-world pipeline data**
- Adapts verification rules as your delivery patterns evolve
- Strengthens guardrails with every deployment, rollback, or incident
- Turns every failure into **system-level learning**, not human rework

Where Harness Fits (1/3): Intelligent CI/CD

qa-testbot-generate-step (Execution Id: 1) Add a Note 1
rohan gupta

Pipeline Inputs Policy Evaluations Artifacts Supply Chain Vulnerabilities Resilience Tests Console View

deployment custom

deployment
Started at: 10/31/2025, 2:10:21 PM | Duration: 1m 21s

Service Infrastructure Resource Constraint Rollout Deployment

Error Summary
Outcome with name log is already saved

Step: Rollout Deployment

Details Input Output Execution Context

Analyze Error AI

Started at: 10/31/2025, 2:10:47 PM
Ended at: 10/31/2025, 2:11:19 PM
Duration: 32s
Timeout: 10m
Failure Strategy Applied: USER_MARKED_FAIL_ALL
Applied By: rohan.gupta@harness.io (10/31/2025, 2:11:19 PM)
Delegates: helm-delegate for Helm Values Fetch Task (Most Recent Delegate Selection Logs)
kubernetes-delegate-idp for K8s Task : Rolling Deploy (Most Recent Delegate Selection Logs)

Step Logs Console View

- Fetch Files
- Initialize
- Prepare
- Apply
- Wait for Steady State
- Wrap Up

Ask AI

Harness AI

Hello rohan gupta, How can I help you?

rohan gupta RG

Analyze the error for the pipeline execution

Harness AI

Thought Process

Let me create the follow-up prompts as requested:

Error Categorization
Error Type: configuration_error

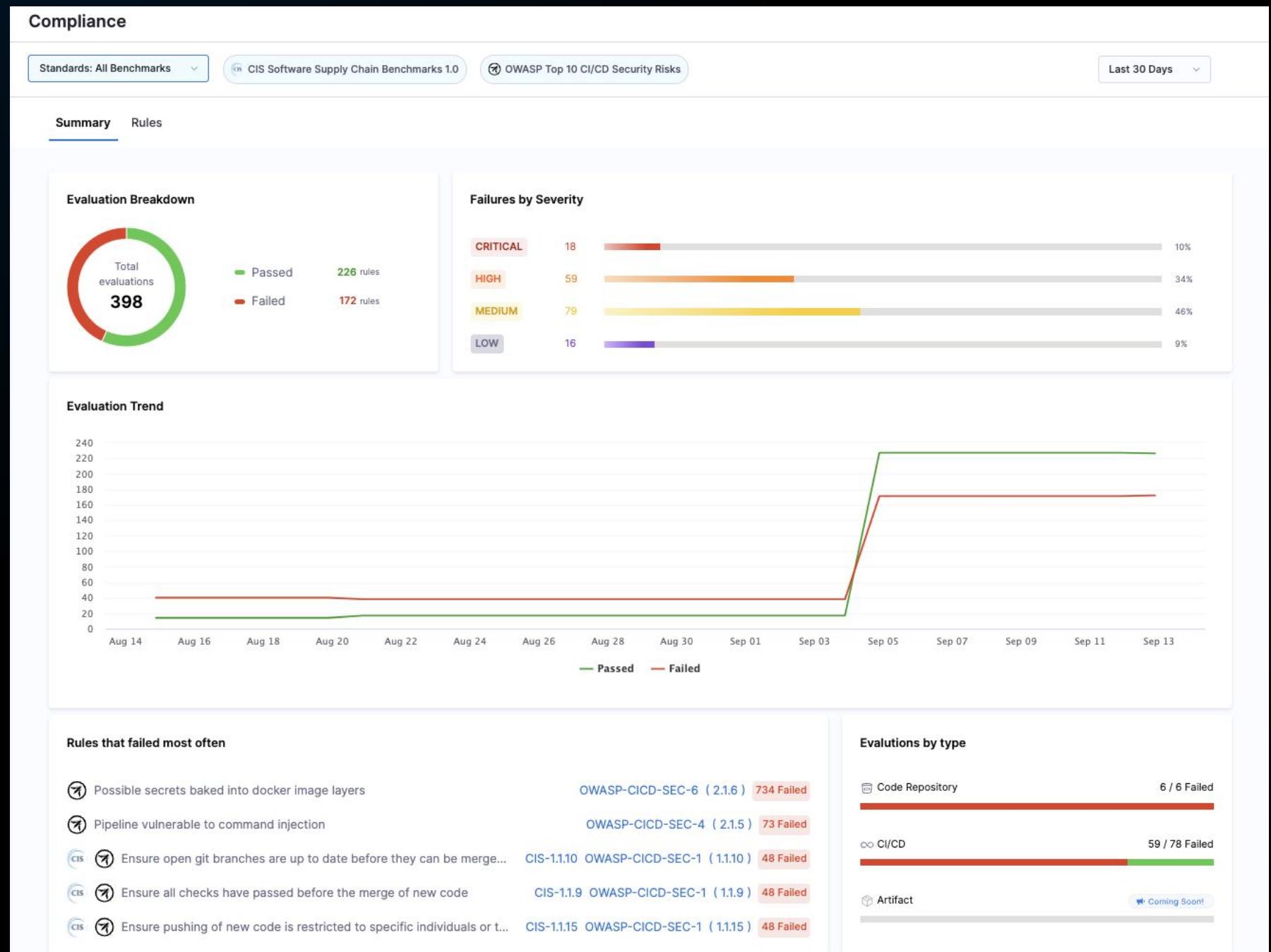
Root Cause Analysis

Summary
The pipeline execution failed due to a duplicate outcome name conflict. The error message "Outcome with name log is already saved" indicates that the pipeline is attempting to save an outcome variable with the name "log" that has already been defined or saved in the current execution context. This is a configuration issue where multiple steps or stages are trying to use the same outcome variable name.

Evidence

Error message: "Outcome with name log is already saved"
Failure type: USER_MARKED_FAILURE with ENGINE_OUTCOME_EXCEPTION
Pipeline execution ID: crH57seuSU28d01n6QKfAQ
Failed at stage: deployment (rolloutDeployment step)
Pipeline contains multiple stages with potential for outcome variable conflicts

Where Harness Fits (2/3): Security built in, not bolted on



Pipeline vulnerable to command injection

ID: (2.1.5) Severity **HIGH**

Failed 1 minute ago

About

Pipelines are vulnerable to command injection attacks, in which arbitrary commands and scripts are passed through user controlled inputs such as pull request title, commit description etc., which are referenced by the pipeline configuration file and executed in the pipeline job. When a pipeline job runs, the poisoned controlled inputs can be interpreted as code, which is then executed on the runner, enabling remote code execution attacks to take place in order to access stored secrets or to read or modify the source code.

Reason

Detected potential use of unsanitized or untrusted inputs in pipeline, increasing the risk of command injection attacks.

General Remediation

To mitigate the risk of command injection attacks, it is crucial to ensure that pipelines do not execute unsanitized or untrusted input. The recommended approach is to avoid referencing user-defined input in pipelines whenever possible. If necessary, use Harness trigger payload conditions to allow only text and numbers, thereby reducing the risk of command injection. For more information on trigger payload conditions, refer here - <https://developer.harness.io/docs/platform/triggers/triggers-reference/#payload-conditions>

1 Failure Occurrences

Occurrence 1

Evidence Snippet

```
31 echo "<+trigger.prTitle>"
```

Location

https://app.harness.io/ng/account/vpCkHKsDSxK9_KYfjCTMKA/module/sca/orgs/default/projects/Security_Test_Scans/pipelines/unscripteddemo_pipeline/pipeline-studio?#L31

Where Harness Fits (3/3): AI-Native Delivery Platform

Let's Recap: The Future of DevSecOps

AI-native delivery expands the attack surface into code, configuration, infrastructure, and model behavior.

Old security patterns cannot keep pace with AI-accelerated pipelines.

The future of DevSecOps is secure-by-default systems that detect, verify, and learn automatically.

A minute to provide feedback.



Thank You

