

PRD: Cloud Security Posture & Remediation Suite

Name: SOUMA CHATTERJEE

My Background: BTech (Electronics & Communication Engineering) | MBA (Tech Management & Marketing)

Role: Product Manager Trainee

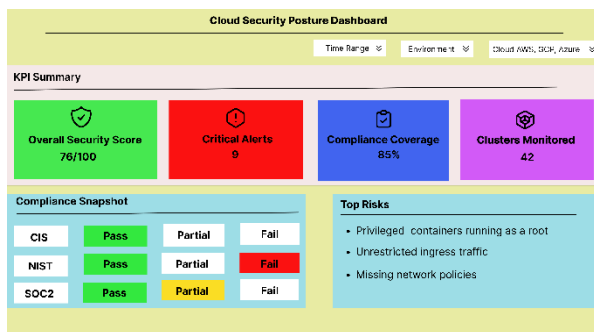
1. What is the problem?

Modern security teams are overwhelmed. If you are managing accounts across AWS, Google Cloud, and Azure, the notifications flood you. What happens is what we refer to as "Alert Fatigue." The real problem isn't just that they see the alerts; it's that the engineers don't really know which of them is dangerous or how to fix them fast. I wanted to build something that doesn't just show "red flags" but actually helps people solve them.

2. For Whom?

- Managers: The CISO/Lead need to understand the "big picture": Are we safe? Are we compliant with regulations and/or standards - e.g., CIS or SOC2?
- Engineers: These will be the ones in the trenches. This should be a simple list that says what to fix first.

3. My Solution: A 3-Step Workflow I decided to put the Dashboard and the Alert Workflow together because they work better that way.



Step 1: The Main Dashboard (Visibility)

This table lists security alerts and vulnerabilities, allowing for filtering by severity (Critical, High, Medium) and cluster. The table includes columns for Alert, Severity, Resource, Category, Status, and Last Detected. Alerts are color-coded by severity: Critical (red), High (orange), and Medium (green). Each alert has a 'View Details' link.

Alert	Severity	Resource	Category	Status	Last Detected
Container running as root	Critical	Prod-cluster-1	Runtime	Open	View Details →
Exposed sensitive key in environment variable	Critical	Web namespace	Configuration	Open	View Details →
Open port 22	High	Prod-cluster-2	Network	Open	View Details →
No network policy applied	High	Prod-cluster-3	Network	Open	View Details →
Container with high CPU usage	High	Staging-cluster	Network	Open	View Details →
Container with cluster	Medium	dev-cluster	Network	Open	View Details →

Step 2: The Alert List (Prioritization)

4. Why I built it this way

- Safety First:** I coloured the "Critical" alerts in bright red. Clearly, if the problem causes the system to crash or data to leak out, it should be presented immediately.
- Easy Filtering:** I have added filters for different clouds like AWS/GCP. If a company is huge, they have to easily sort through the data.
- User Flow:** I actually combined visibility and fix into one because, obviously, the dashboard is not going to be useful to anyone if they don't take action.

This page provides detailed information about a specific alert, including its impact, affected clusters, and compliance violations. It also lists recommended fixes for the issue.

Details
Impact: Elevation of privileges and potential container compromise
Affected Clusters: Prod-cluster-1, staging-cluster
Compliance Violation: CIS Control 5.3 - Ensure containers do not run as root
Recommended Fix: <ul style="list-style-type: none">Disable root accessApply pod security policies

Step 3: The Fix (Remediation)

5. How we'll know it's working

- Primary Goal (MTTR):** We'd like the time it takes to fix a bug to decrease.
- Success Metric:** Are more of our cloud accounts passing their security checks?
- The "Check" (Counter Metric):** I'll pay attention to the "Re-open Rate." If an alert re-appears again and again despite an engineer "fixing" it, our instructions are probably not clear enough.
- Speed Check:** I want to make sure that the dashboard does not slow down the actual systems while gathering information.

6. Bonus: Next steps for the Dev Team

In order to build this, what I would discuss with the engineers is:

- Connecting Data:** We must get the live data from AWS and Azure and display it in the dashboard.
- Shared Status:** When someone starts to work on the alert, it should be reflected as 'In Progress' for all others so that we're not duplicating the work.
- Smooth Feel:** The filters should work immediately without the whole page having to reload.
- Direct Links:** We can add a button that will take the engineer to where in the AWS interface he can correct the trouble or error.