

Problem Overview

The core problem is that cloud permissions grow messy and risky over time, and security teams lack a clear, safe way to clean them up. Roles and users often receive “temporary” or overly broad access that is never removed, leading to unused permissions, wildcard access, and too many admin-level identities. This creates a large attack surface and audit findings, but manually reading policies and logs is slow and error-prone. The experience therefore needs to surface the riskiest identities first, show exactly which permissions drive that risk, and make it obvious how often those permissions are actually used.

My understanding is that solving this is not just about visibility, but about enabling confident remediation. Analysts need help turning raw findings into concrete, least-privilege fixes while still feeling in control. That is why the flow I designed moves from high-level KPIs and a prioritized entity list, to a detailed role view with recommended fixes, and finally to a confirmation screen that compares current vs. proposed policy and explains impact. The prioritization focuses on triage (where to start), explanation (why this is risky), and safe action (what will change), so security teams can reduce risk quickly without accidentally breaking critical workloads.

Target User Persona

Who the user is

The user is a cloud security engineer or IAM (Identity and Access Management) specialist. They work at a company that uses cloud platforms like AWS, Azure, or GCP and have to make sure people only have the access they really need.

- They are comfortable reading things like roles, policies, and actions (`ec2:StartInstances`, `s3:GetObject`, etc.).
- They are usually part of a security or DevOps team and often get pulled into audits, incidents, and compliance checks.
- They are busy and don’t have time to manually read long JSON policies or check logs for every single permission.

Their main goals are:

- Keep the company’s cloud environment secure by avoiding over-privileged access.
- Quickly find which users/roles are risky and fix them with confidence.
- Avoid breaking production systems while tightening permissions.

What they care about

This person cares about three things the most:

- **Clarity:** They want to immediately see “what’s wrong and where” instead of digging through raw policy files.
- **Confidence:** Before making a change, they want to understand the impact so they don’t accidentally break a critical app.
- **Speed:** They need to handle many issues in a day, so they appreciate guided flows, clear recommendations, and fewer manual steps.

Representative User Scenario

Asha, the cloud security engineer

Imagine Asha, a cloud security engineer at a fintech startup:

- The company has grown quickly and over the last two years many engineers have been given “temporary” admin permissions that were never taken back.
- Asha gets a request from her CISO: “Find our riskiest IAM roles and clean them up before the next audit in two weeks.”

Using the screens you designed:

1. On Screen 1 (IAM Permissions Explorer), Asha sees KPIs like “High-Risk Entities” and a table of risky roles. She quickly spots DevOps-Admin with a High risk score and opens it.
2. On Screen 2, she sees a detailed list of permissions, when they were last used, and clear recommended fixes (remove unused permissions, restrict S3 to read-only, etc.). This saves her from manually comparing logs and policies.
3. She clicks Apply Recommended Fixes, which takes her to Screen 3. Here, she can clearly see:
 - Current Policy
 - What will change
 - The Proposed Least Privilege Policy and new (lower) risk levelThis lets her double-check that important actions are kept while risky, unused ones are removed.

Because the flow is clear and transparent, Asha can confidently apply the fix, document the change, and move on to the next risky role. The product has helped her be faster, safer, and more confident in her decisions.