PO SECURITY

Automate least-privileged database access with precisely scoped, Just-in-Time controls

Databases are present in almost every internet-facing application. Modern enterprises run critical workloads on managed cloud database services, open-source database engines and cloud data warehouses.

Yet access to these sensitive systems often rely on overly permissive, standing access that creates significant risk, audit gaps and operational drag. Teams are left managing credentials and manual workflows rather than protecting access to their crown jewels.

PO's approach to database access

PO Security redefines privileged access for cloud databases by removing shared DB users and credential-based access. Replacing standing privilege with fine-grained, JIT access. No vaults, bastions or proxies required.

PO delivers least-privilege database access that is provisioned on demand, automatically revoked when no longer needed and fully auditable by design.

1. Enforce just-enough and Just-in-Time access, replacing static credentials and standing privilege
2. Provision access to a user's IdP-native identity, removing shared accounts and manual workflows
3. Simplify operations with API-led orchestration, no added infrastructure to deploy or manage

All with seamless developer experiences via Slack/Teams, web console or the PO CLI.

Capabilities

- Enforce least-privilege by binding requests to specific tasks, DB roles (e.g. read-only, stats-reader, admin-lite) and even individual queries
- Dynamically provision and revoke access with automated expiration windows
- Tie all session logs to the accountable end-user identity by leveraging the target system's authentication method
- Automate policy-based request routing to designated approvers (owners, managers, DBAs) for faster access

Outcomes

- **Zero standing privileges:** All access is tied to a verified IdP identity, granted JIT and automatically revoked after use
- **Granular and flexible control:** Roles and policies are scoped to allow access based on dynamic needs and DB roles
- **Global governance at enterprise scale:** Manage servers, databases, K8s, cloud and agentic apps under a single policy framework and consistent developer UX
- **Fully auditable, tamper-evident logs:** Sessions are tied to the accountable human user – never a shared account
- **Fast, infrastructure free deployment:** Integrates with your existing setup, no proxies or agents needed

JIT PostgreSQL access

Getting started with P0 Security for database access

Setting up access to your databases takes just a few steps. **For this example, we will look at managed instances of PostgreSQL.**

Simply connect your cloud account in the P0 dashboard, set up your database integration and link P0 to your existing identity provider. Once configured, team members securely request access to PostgreSQL roles or to execute PostgreSQL queries.

You can also define routing and approval policies to direct access workflows based on roles or queries, ensuring the right people get the right access through approved paths. The entire process takes under an hour and scales automatically as your environment grows. Users can make requests seamlessly via the P0 CLI, web console or Slack/Teams. P0 will temporarily provision the relevant database permissions and automatically revokes access on expiry.

Sample workflow: PostgreSQL Database in Google CloudSQL

1

A developer requests access to a PostgreSQL role or to execute a specific query, requesting time-bound access from their existing tool stack like Slack or the P0 CLI.

2

The request is validated against organizational policies that define roles, environments, justifications and approval workflows.

3

Once approved, P0 provisions JIT database access and the user runs a native auth flow (e.g., SSO profile) to connect. No secondary credentials, no shared accounts.

4

The user directly connects from within the P0 CLI and P0 uses its PostgreSQL integration to securely apply and later revoke task-specific role grants.

5

P0 records all access activity under the accountable user identity, providing full visibility and on demand audit trails with session level playback.

6

When the session ends or the approval window expires, access is automatically revoked, **ensuring zero standing privilege** across your cloud environment.

JIT Snowflake access

Getting started with P0 for database access

Setting up access to your databases takes just a few steps. **For this example, we will look at Snowflake data warehouses.**

Simply connect your cloud account in the P0 dashboard, set up your Snowflake integration and link P0 to your existing identity provider. Once configured, team members securely request access to Snowflake roles or execute queries directly from P0.

You can also define routing and approval policies to direct access workflows based on roles or queries, ensuring the right people get the right access through approved paths. The entire setup takes under an hour and scales automatically as your Snowflake footprint grows. Users can make requests seamlessly via the P0 CLI, web console or Slack/Teams. P0 will temporarily provision the relevant Snowflake permissions and automatically revokes access on expiry.

Sample workflow: Snowflake data warehouses

1. A developer with zero standing privilege requests access to a role, or to execute a query, for a preset duration via an approved channel such as Slack, the P0 CLI or console.
2. The request is validated against organizational policies that define roles, environments, justifications and approval workflows.
3. Once approved, P0 provisions JIT Snowflake access and the user runs a native auth flow (e.g., SSO profile) to connect. No secondary credentials, no shared accounts.
4. The user directly connects from within the P0 CLI and P0 uses its Snowflake integration to securely apply and later revoke task-specific role grants.
5. P0 records all access activity under the accountable user identity, providing full visibility and on demand audit trails with session level playback.
6. When the session ends or the approval window expires, access is automatically revoked, **ensuring zero standing privilege** across your cloud environment.