A Practical Guide

Privileged Access Management

Balancing Developer Experience, Security, and Compliance

Written by Mario Kamburov, Security Professional

2026

Executive Summary

Modern SaaS companies allow engineers to access production systems daily.

They:

- SSH into servers
- Execute risky commands like `kubectl exec, psql` commands
- Connect to production databases
- Query raw customer data
- Debug live systems
- View customer data in unencrypted raw format

This access is operationally necessary.

But it also represents one of the largest structural security risks in modern organizations.

This book explains:

- Why privileged access risk is increasing
- How it conflicts with SOC 2, ISO 27001, and customer contracts
- Why logging alone is insufficient
- Why audit logging is incomplete
- Why session recording is the missing control layer
- How to implement modern privileged access management (PAM)
- How to measure and report on privileged access risk

Section I – Understanding the Problem

The Reality of Modern Engineering

Fast-growing startups prioritize:

- Rapid iteration
- Developer autonomy
- Shift-left ownership
- Self-service infrastructure

Production access becomes normal

Not exceptional

Over time:

- Direct SSH becomes standard practice
- Broad cluster access spreads
- Database access remains persistent
- Privileges accumulate
- Offboarding lags behind growth

The risk increases silently

The Structural Risk

The danger is not malicious engineers

The real risks are:

- Human error
- Over-permissioned access
- Credential compromise
- Supply Chain Malware
- Insider misuse
- External hijacking of legitimate credentials (Human vs. Non-Human identities, shared technical accounts, service accounts etc.)

Production contains:

- Personal data
- Financial information
- Authentication tokens
- Trade secrets

If access is frequent and under-monitored, exposure becomes inevitable.

Compliance & Legal Collision

Customer contracts increasingly require:

- Strict access control
- Encryption protections
- No unauthorized viewing or moving of customer data (deidentification of sensitive data, data masking or anonymization)
- Prior approval before accessing production data
- Full auditability and audit rights

SOC 2 requires:

- Logical access controls
- Monitoring of privileged users
- Ability to reconstruct incidents

ISO 27001 requires:

- Control of privileged access rights
- Logging and monitoring
- Segregation of duties
- Accountability

Modern engineering culture optimizes for speed

Contracts optimize for restriction

Without structured control, these worlds conflict

Framework	**Control Area**	**PAM Contribution**
SOC 2 CC6	Logical Access	JIT, RBAC
SOC 2 CC7	Monitoring	Session recording
ISO 27001 A.9	Access Control	Centralized enforcement
HIPAA 164.312(b)	Audit Controls	Session replay

Privileged Access Control Mapping Overview per Security Framework

The Maturity Gap

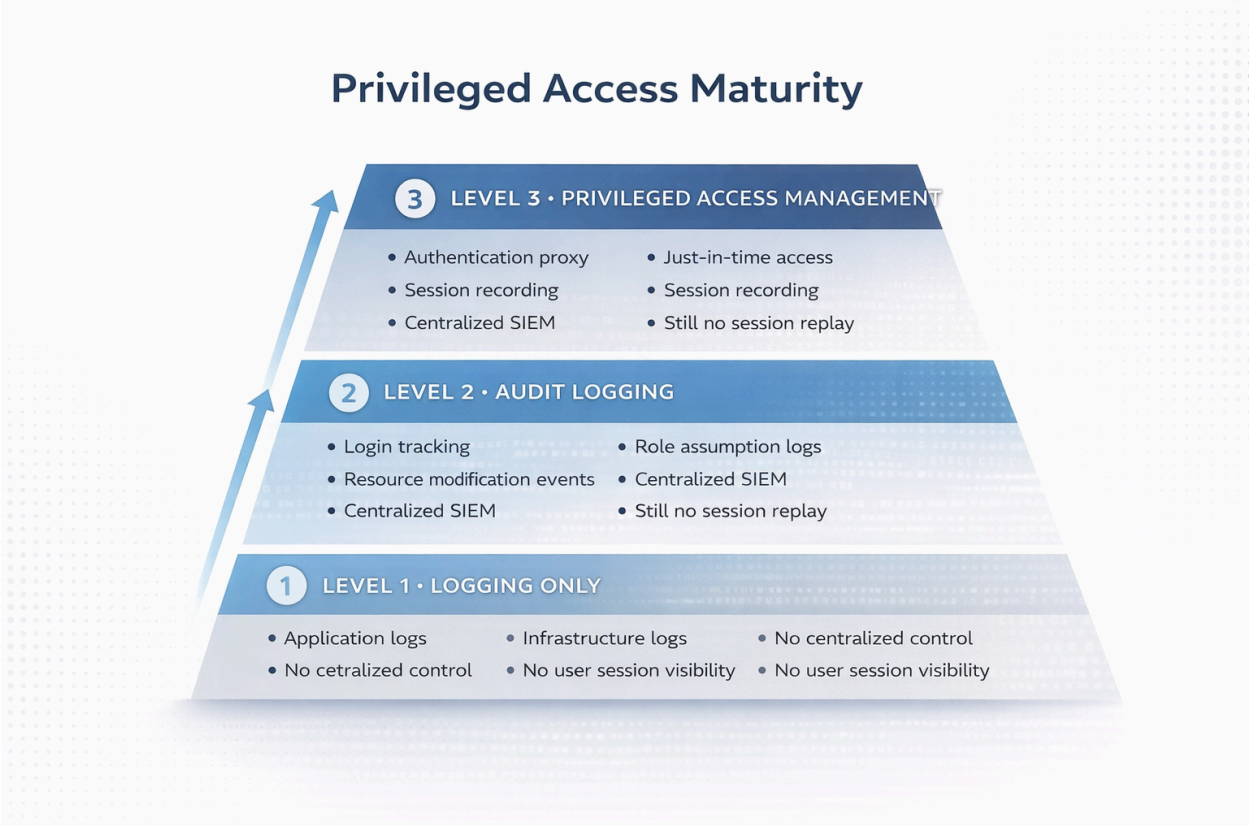
Many organizations believe: “We have logs. We’re covered.”

But logs ≠ oversight

To understand why, we must distinguish:

- **Logging**
- **Audit logging**
- **Session recording**

These represent different maturity levels of **risk oversight**

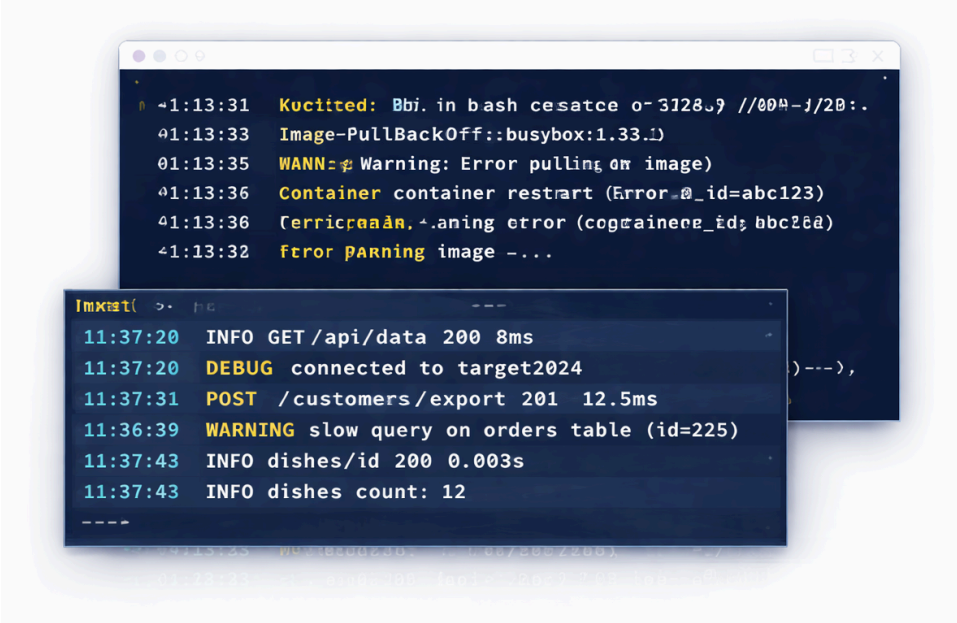


Section II – Understanding the Layers

Layer 1 – Logging (System-Level Telemetry)

Logging captures:

- Application errors
- Service interactions
- System metrics
- Container output



Purpose:

- Debugging
- Reliability
- Observability
- **Focus on Availability**

Perspective:

- Backend/system view.
- It tells you how the system behaved.
- It does not tell you what a human intentionally did.

Risks:

- Direct SSH bypass
- Hard-to-correlate logs
- No central enforcement
- Blind to user behavior
- Insufficient oversight

At scale, this becomes unmanageable.

Layer 2 – Audit Logging (Security-Level Events)

Audit logs record:

- Login events
- Role assumptions
- Resource creation/modification
- Database connection attempts
- Permission changes
- Domain-specific actions

Purpose:

- Accountability
- Traceability
- Compliance reporting
- Incident investigation
- **Focus on Security and Compliance**

Perspective:

- Structured security events
- Audit logs are “footprints.”
- You know someone entered
- But you don’t know what happened inside

Risks:

Audit logs improve traceability, but they still:

- Fragment evidence across systems
- Require manual correlation
- Leave gaps in context
- Allow ambiguous reconstruction

In incident scenarios, ambiguity is costly and operationally heavy.