CLOUDNIX TECHNICAL REPORT · CNX-TR-2026-09

# Defence in Depth for Business Platforms: Security Practice in Production

Ajmal Abubacker, Elango, Avinash KS, Upendranath Metikala and Avinash Nishant

*Cloudnix Software Labs, Bengaluru, India*

CNX-TR-2026-09 · Version 1.0 · November 2026

**Abstract**

Most security failures in business software are ordinary: databases reachable from the internet, servers that accept passwords, requests that cross tenants, backups never restored and fixes never applied. We describe the layered practice we apply to platforms we design, build and operate: edge protection with rate limits and escalating bans, key-only server access, scoped cloud accounts, granular roles with monthly reviews, tenant scope on every request, gateway-hosted card entry, encryption, backups on three schedules with measured restore times, a four-hour critical patch window, daily log review, incident response by the engineers who built the system, and a mapping of these controls to SOC 2, ISO 27001 and GDPR.

**Keywords:** *application security; defence in depth; rate limiting; access control; multi-tenancy; PCI DSS; encryption; backup and recovery; patch management; incident response; SOC 2; ISO 27001; GDPR*

---

**Contents**

1. Executive summary
2. Threat model
3. Defence in depth
4. The edge
5. Server access
6. Cloud accounts and network
7. Identity and roles
8. Request-level checks
9. Data protection and payments
10. Backups and recovery
11. Patching and monitoring
12. Incident response
13. Readiness for SOC 2, ISO 27001 and GDPR
14. A checklist for your own platform
15. References

## 1 Executive summary

Most security failures in business software are not exotic. A database answers on the public internet; a server still accepts passwords; a customer changes an ID in the address bar and sees another customer's order; a backup that was never restored turns out to be empty; a fix that existed for months was never applied. Each of these is a gap in one layer that the other layers were not built to catch.

This report describes the layered security practice we apply to the platforms we design, build and operate: commerce platforms serving many stores, workflow and SaaS platforms serving many organisations, and AI systems that read private data. It covers the edge, server access, cloud accounts, identity, request-level checks, data and payments, backups, patching and monitoring, incident response and readiness for SOC 2, ISO 27001 and GDPR. Every control described is in production on servers we run; numbers are taken from those servers and platforms.

**Principles.**

1. **Assume each layer can fail.** No single control is trusted to stop everything.
2. **Close by default.** Ports, permissions and data are shut until a need opens them.
3. **Never hold what you cannot lose.** Card numbers stay with the payment gateway.
4. **Scope every request.** No tenant, no data.
5. **Prove recovery.** A backup is real only after it has been restored.
6. **Patch in hours, not months.** Most attacks use flaws that already have fixes.
7. **Keep evidence as you go.** Audits read records, not intentions.

## 2 Threat model

We design against the attacks business platforms actually see, ranked by how often they appear in our own logs and in published industry data [1, 2].

| Threat | Typical form | Layer that stops it |
|---|---|---|
| Automated probing | Scripts requesting admin paths, backup files and known backdoors | Edge: probe traps, bans |
| Floods and scraping | Bursts of requests from one or many addresses | Edge: CDN, DDoS protection, rate limits |
| Credential attacks | Password spraying on logins, OTP and reset pages; server login guessing | Edge limits; key-only server access; per-account limits |
| Broken access control | Changing an ID or tenant in a request | Application: tenant scope, permission checks |
| Privilege creep | Former staff and unused roles that keep access | Identity: monthly reviews, expiry |
| Data exposure | Databases reachable from the internet; card data in tables or logs | Server: closed ports; data: gateway-hosted card entry |
| Unpatched software | Known vulnerabilities in the operating system or libraries | Patching: daily updates, 4-hour critical window |
| Data loss | Deletion, corruption, ransomware, provider outage | Backups on three schedules, two clouds, tested monthly |

## 3 Defence in depth

A request from a user crosses six layers before it reaches data. Each layer has its own controls and its own failure modes, and each assumes the previous one can be bypassed.

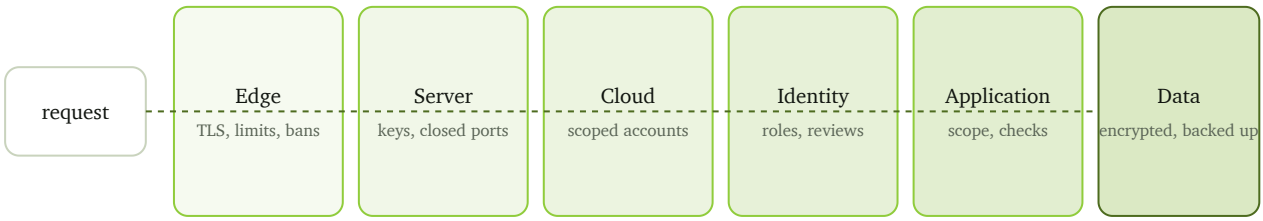


**Figure 1.** Six layers between a request and the data.

# 4 The edge

## 4.1 *Transport*

Every page is served over HTTPS with certificates renewed automatically. Protocol versions older than TLS 1.2 are disabled; plain HTTP exists only to redirect to HTTPS [3].

## 4.2 *Rate limits and escalating bans*

Each visitor is limited to 300 requests in five minutes on our commerce servers. A visitor who crosses the limit is banned for an hour; bans grow for repeat offenders, and three bans within a day bring a one-week ban. Logins, one-time passwords and password resets have their own tighter limits, per visitor and per account, and public forms carry a CAPTCHA.

## 4.3 *Probe traps*

Requests for paths that only an attacker would ask for (known backdoor scripts, backup files, administration tools we do not run) are banned on first sight. The rules are tuned per platform from its own logs, so that legitimate traffic, for example buyers on a sale day, is never caught.

## 4.4 *CDN and DDoS protection*

The busiest platforms sit behind a content delivery network and the cloud provider's DDoS protection, which absorbs volumetric attacks before they reach our servers. Response headers forbid framing by other sites and content-type guessing by browsers [4].

| Control | Setting |
|---|---|
| TLS | 1.2 and above; automatic renewal |
| Visitor rate limit | 300 requests / 5 minutes |
| First ban | 1 hour, increasing on repeat |
| Repeat offenders | 3 bans in 1 day → 1 week |
| Probe paths | banned on first request |
| Headers | frame-ancestors self; nosniff |

# 5 Server access

Servers accept key-based logins only; password and keyboard-interactive authentication are disabled, direct root login is refused, and a single named account is allowed per server. The login service listens on a non-default port watched by the same banning system as the web edge [5].

Databases listen only on the server's local interface. From the internet, the database port does not answer at all; the application reaches it from inside the machine. The only ports that answer from outside are the web ports and the login port.

**Why it matters.** Exposed databases and password-based server logins remain among the most common causes of large data breaches. Both are closed by configuration, at no cost, and both are checked on every server we run.

## 6 Cloud accounts and network

The cloud console can stop, copy or delete every server and backup. Full console access is held by one account; engineers, operations and billing each have access scoped to the services their work needs. A network firewall at the cloud level allows only web traffic and the login port to reach servers, so a misconfigured server cannot expose a new port by accident.

## 7 Identity and roles

Every person has their own login; there are no shared accounts on platforms we run. Roles are composed from fine-grained permissions rather than a single administrator switch: one commerce platform defines about 190 permissions, and a SaaS platform uses owner, administrator, member and guest roles per team, with one owner per team.

- **Step-up checks.** Sensitive changes, such as payment details or roles, ask for a fresh check.
- **Expiry.** Temporary access ends on its own; session time-outs end idle sessions.
- **Integration keys.** Each user or integration has its own key, so one can be revoked without affecting others.
- **Monthly review.** Access is reviewed each month; unused access is removed, and access ends the day someone leaves.

These follow the access-control requirements of the OWASP Application Security Verification Standard [6] and the authenticator guidance of NIST SP 800-63B [7].

## 8 Request-level checks

Broken access control is the most common serious flaw in web applications [1]. In multi-tenant platforms it usually takes one form: a request for a record that belongs to another tenant. Our platforms resolve the tenant once, early in every request (from the host name for a store, from the signed-in session for an organisation) and every query carries that scope.

```
-- every read is bound to the tenant resolved for this request
SELECT * FROM orders WHERE store_id = :tenant AND id = :id;
-- a request from tenant B for order 1042 of tenant A returns no rows
```

**Listing 1.** Example exchange or schema.

- **Inputs and uploads** are validated before they are stored; uploaded files are checked by type and kept outside executable paths.
- **Order and payment steps** are enforced on the server; a browser cannot skip from cart to paid.
- **One-time passwords** expire, and login, OTP and reset attempts are limited per account as well as per visitor.
- **Reviews** look for the same mistakes every time: missing scope, missing permission checks and trust in values sent by the browser.

## 9 Data protection and payments

### 9.1 *Payments*

Card numbers are entered on the payment gateway’s own page. The platform receives a payment reference and a status, verified with the gateway, and raw card numbers are never stored on our servers. This keeps

card data out of our databases, logs and backups and reduces the scope of PCI DSS obligations [8].

### 9.2 *Encryption and access*

All connections are encrypted in transit. On audited platforms, data is encrypted at rest with AES-256. Sensitive records are visible only to signed-in users whose role allows it.

### 9.3 *Classification and retention*

Data is classified by sensitivity, and retention rules decide what is kept and for how long, so that old personal data does not accumulate in tables and backups. These are the same principles GDPR sets out as data minimisation and storage limitation [9].

## 10 Backups and recovery

Every platform is copied on more than one schedule, to more than one place, and each copy has a known restore time. The schedule below is from one platform we run.

| Copy | Frequency | Location | Kept | Restore time |
|---|---|---|---|---|
| Disk snapshot | Daily | Same cloud, Singapore | 7 days | 10 minutes |
| Database and code | Every 6 hours | Different cloud, Bangalore | 4 days | 2 hours |
| Machine image | Weekly | Same cloud, Singapore | 1 month | 20 minutes |
| Regional copy | On demand | United States, European Union or United Kingdom | As agreed | Per region |

Where a customer’s data must stay in, or also be held in, a particular jurisdiction, we add backup copies on demand in the United States, the European Union or the United Kingdom, with the schedule and retention agreed for that platform.

Commerce platforms keep live orders backed up daily and the full store weekly, in three copies: production, off-site and a cold archive. Every month a backup is restored for real, so recovery is a measured procedure rather than an assumption. This follows the contingency planning practice of NIST SP 800-34 [10].

## 11 Patching and monitoring

| Item | Target |
|---|---|
| Critical vulnerability | patched within 4 hours |
| Operating system security updates | installed automatically, daily |
| Other updates | monthly |
| Certificates | renewed automatically |
| Logs | compressed, summarised and reviewed daily |

Alerts fire on repeated failed logins, role changes and traffic spikes; uptime checks run against every platform. Logins, role changes and other key actions are recorded, so any change can be traced to a person and a time.

## 12 Incident response

The engineers who built a system are the ones on call for it. An incident moves through detection, triage, containment, fix, root-cause analysis and a lasting change, following the structure of NIST SP 800-61 [11]. Customers are told directly by the engineers handling the incident, not through a ticket queue. Every incident ends with a change to code, configuration or process, so the same failure cannot recur the same way.

## 13 Readiness for SOC 2, ISO 27001 and GDPR

Audits are faster and cheaper when controls already exist and leave evidence. Our practice maps onto the main frameworks as follows; a platform we engineer and run is audited against SOC 2, with change and vendor management in scope [12, 13, 9].

| Practice | SOC 2 (TSC) | ISO 27001:2022 | GDPR |
|---|---|---|---|
| Access control, reviews | CC6.1–CC6.3 | A.5.15–A.5.18 | Art. 32 |
| Change management | CC8.1 | A.8.32 | – |
| Logging and monitoring | CC7.2 | A.8.15–A.8.16 | Art. 32 |
| Vulnerability management | CC7.1 | A.8.8 | Art. 32 |
| Backups and recovery | A1.2–A1.3 | A.8.13 | Art. 32(1)(c) |
| Incident response | CC7.3–CC7.5 | A.5.24–A.5.28 | Art. 33–34 |
| Encryption | CC6.7 | A.8.24 | Art. 32(1)(a) |
| Data minimisation, retention | C1.2 | A.5.33–A.5.34 | Art. 5(1)(c), (e) |
| Vendor management | CC9.2 | A.5.19–A.5.22 | Art. 28 |

## 14 A checklist for your own platform

1. Can anyone on the internet reach your database port?
2. Do your servers still accept password logins, or root logins?
3. If a signed-in customer changes an ID in a request, do they see another customer's data?
4. Can a login, OTP or reset page be tried a thousand times?
5. Who still has access who has left, and when were roles last reviewed?
6. How long does a critical patch take to reach production?
7. Are card numbers anywhere in your database, logs or backups?
8. When did you last restore a backup, and how long did it take?
9. Would you know, today, if someone changed an administrator's role?
10. If an auditor asked tomorrow, could you show evidence for each control?

## 15 References

1. OWASP Foundation. *OWASP Top 10: 2021*.
2. Verizon. *Data Breach Investigations Report*, 2024.
3. Mozilla. *Server Side TLS guidelines*.
4. OWASP Foundation. *HTTP Security Response Headers Cheat Sheet*.
5. Center for Internet Security. *CIS Benchmark: Ubuntu Linux*.

6. OWASP Foundation. *Application Security Verification Standard*, v4.0.3.
7. P. Grassi et al. *Digital Identity Guidelines: Authentication and Lifecycle Management*. NIST SP 800-63B.
8. PCI Security Standards Council. *PCI DSS v4.0*, 2022.
9. European Union. *Regulation (EU) 2016/679 (General Data Protection Regulation)*.
10. M. Swanson et al. *Contingency Planning Guide for Federal Information Systems*. NIST SP 800-34 Rev. 1.
11. P. Cichonski et al. *Computer Security Incident Handling Guide*. NIST SP 800-61 Rev. 2.
12. AICPA. *2017 Trust Services Criteria for Security, Availability, Processing Integrity, Confidentiality, and Privacy*.
13. ISO/IEC. *ISO/IEC 27001:2022 Information security management systems: Requirements*.

**About this report.** Prepared by the Cloudnix engineering team from systems we have designed, built and operated. Client and product names are withheld; illustrative values are marked as examples. Cloudnix Software Labs, Bengaluru, India · cloudnix.com. This report may be shared freely in its entirety.

**Document control**

| | | | |
|---|---|---|---|
| Report | CNX-TR-2026-09 | Version | 1.0 |
| Issued | November 2026 | Status | Published |
| File | cloudnix-security-practices-v1.0.pdf | | |