APEX IT SOLUTIONS

INFRASTRUCTURE · SECURITY · DEVOPS

50

QUESTIONS

POCKET GUIDE

The Pre-Hire Infrastructure Audit

50 questions to answer before you hire your next DevOps engineer

FORMAT	DOMAINS	EDITION	AUDIENCE
50-Question Checklist	Seven (A–G)	2026 Edition	Business Owners & CTOs

FOREWORD · A NOTE FROM THE AUTHOR

A Note to You, the Business Owner

Hiring is expensive. A bad DevOps hire is catastrophically expensive.

Not because they are incompetent — though that happens — but because the most common hiring mistake in technology is this: bringing in someone to fix problems that were never properly diagnosed. You hire a mid-level engineer when the situation actually needs a senior one. Or you hire a builder when what you need is a fixer. Or you hire anyone at all when the real issue is that your infrastructure has silently accumulated three years of technical debt that no single new hire will survive.

This guide exists to help you avoid that mistake.

We put it together because, frankly, we have seen too many businesses spend months and significant money on hiring — only to have the new engineer walk in, look at the environment, and start making plans to leave within 90 days. The infrastructure was not ready for them. Or they were not right for the infrastructure. The audit was never done.

This checklist covers 50 specific, auditable questions across seven domains of your infrastructure. It is not a surface-level health check. It is the kind of forensic review that tells you, before you post a single job ad, exactly what kind of hire you actually need and what they are walking into.

Work through it honestly. Score your answers. The results will be more useful than any job description you have written.

> **HOW TO USE THIS GUIDE**
>
> Read each item. Ask the question internally. Mark it as Passed (your team can clearly demonstrate this is in place), Failed (it is not in place), or Unknown (you are not sure — which is itself important information). At the end of each section, tally your score. The scoring guide on the final page tells you what the numbers mean for your hiring decision.

One more thing: if you get to the end of this and the honest answer is that you do not know who in your organization could even tell you whether these controls exist — that is the most important finding of all. And it is exactly the kind of situation we help with.

Sohaib Aftab
Senior DevOps Engineer & Co-Founder, Apex IT Solutions

ORIENTATION · BEFORE YOU BEGIN

Before You Begin: How to Read This Checklist

A short orientation so the checklist scores honestly — what the three risk levels mean, and exactly how to mark each item as you work through it.

The Three Risk Levels

Every question is tagged with one of three risk levels. Use them to prioritise: a single failure in **High Risk** is more important than several failures in **Lower Risk**.

HIGH RISK

A failure here is a breach, a major outage, or a compliance violation waiting to happen. Fix before hiring.

RECOMMENDED ACTION

Fix first

MEDIUM RISK

A failure here will slow you down, create inefficiency, or surface as a problem within 6-12 months.

RECOMMENDED ACTION

Plan to fix

LOWER RISK

A gap here represents optimization opportunities. Good to address but not urgent.

RECOMMENDED ACTION

Improve over time

Scoring Each Item

Mark each item using the system below. Treat **Unknown** as a fail — if no one can confirm a control exists, it does not.

STATUS	MARK IT	WHAT IT MEANS
Passed	[P]	You can demonstrate this control is in place and working.
Failed	[F]	You know this control is not in place.
Unknown	[?]	You are not certain. This counts as a fail for scoring purposes.

AUDIT DOMAIN A

A

Security & Compliance

SOC 2 / ISO 27001 Readiness — The controls auditors look for first

Security failures do not announce themselves. They accumulate quietly — an overly permissive IAM policy here, a hardcoded key there — until the day they become a headline. This section covers the specific controls that distinguish organizations that take security seriously from organizations that think they do.

QUESTIONS IN THIS DOMAIN

15 items

APEX IT SOLUTIONS

INFRASTRUCTURE ACCESS & IDENTITY

01 Principle of Least Privilege (PoLP) Enforcement

HIGH RISK

IAM roles must be scoped strictly to required actions only. Early-stage companies often create roles with wildcard (*) permissions because it's faster. It is. It's also the single most common reason a compromised credential becomes a catastrophic breach. Any policy allowing * on critical resources — S3, RDS, EC2 — is an automatic failure in a SOC 2 audit. Ask your team: "Can someone pull the IAM policies for our three most critical services and show me exactly what each one is permitted to do?" Red flag: Nobody can answer this without a 30-minute investigation. That means nobody is actively managing it.

02 MFA on All Administrative Access

HIGH RISK

Multi-factor authentication for console login is table stakes in 2026. The real question is whether MFA is enforced on CLI and API access. Hardcoded long-term access keys stored in ~/.aws/credentials on developer laptops are one of the primary supply chain attack vectors. Engineers should authenticate via SSO to generate short-lived session tokens. Ask your team: "Does any engineer on the team have static AWS access keys stored locally on their machine?" Red flag: The answer is yes, and nobody knows how many or where they are stored.

03 Service Account Isolation in Kubernetes

HIGH RISK

Sharing a ServiceAccount across multiple applications is extremely common and extremely dangerous. If App A and App B share an identity, a vulnerability in App A grants an attacker access to everything App B is permitted to do. One ServiceAccount per application workload is not a best practice — it is a requirement. Ask your team: "How many unique ServiceAccounts exist in our cluster, and how many applications share each one?" Red flag: There are fewer ServiceAccounts than applications, or nobody knows the answer.

04 Just-in-Time (JIT) Admin Access

HIGH RISK

Permanent administrative access is a permanently open door. The 2026 standard is that admin privileges are requested, approved, granted for a limited window (under 4 hours), and automatically revoked. Tools like AWS IAM Identity Center and Azure PIM implement this. Every elevation should generate a distinct audit log entry. Ask your team: "What steps does an engineer take when they need admin access to production right now?" Red flag: The answer is 'they already have it' or 'they ask someone for the password.'

05 Automated Offboarding

HIGH RISK

Manual offboarding is where ghost accounts come from. When an employee leaves, their access should be revoked across all systems — cloud, GitHub, VPN, SaaS tools — within 24 hours, automatically. An Identity Provider (Okta, Azure AD) configured with SCIM provisioning handles this. Without it, you almost certainly have former employee accounts still active somewhere. Ask your team: "Can you show me the last three offboarding events and confirm which systems had access revoked and when?" Red flag: There is no log, or the process is a checklist that someone manually works through when they remember.

KUBERNETES SECURITY

06 RBAC Authorization Audit

HIGH RISK

cluster-admin is the Kubernetes equivalent of giving every developer a root key to every server. Running kubectl describe clusterrolebindings should show that this role is bound only to a specific, named platform administration group. If developers have cluster-admin, a misconfigured deployment or a compromised account can destroy the entire cluster. Ask your team: "Who in our organization has cluster-admin access, and can we see that list right now?" Red flag: Nobody knows, or the list includes general developer groups.

07 Pod Security Standards Enforcement

MEDIUM RISK

PodSecurityPolicies were deprecated in Kubernetes 1.25. Their replacement — Pod Security Standards — must be actively enforced via admission controllers. The key things to block: privileged containers and HostPath mounts. A HostPath mount allows a container to access the node's filesystem, which effectively breaks container isolation entirely. Ask your team: "Are privileged containers blocked in our cluster? Can we deploy one right now to test?" Red flag: Nobody is sure, or a test deployment of a privileged container succeeds without challenge.

08 Secrets Never in Git or Plain-Text YAML

HIGH RISK

This is the most common security failure in early-stage engineering teams, and the most consistently embarrassing. Secrets — API keys, database passwords, JWT secrets — that have ever been committed to a Git repository remain in the history even after deletion. Scan your repository history for high-entropy strings before your next penetration test does it for you. Ask your team: "Can we run a secrets scan on our Git history right now?" Red flag: The team hesitates, or the scan returns results within the first 30 seconds.

09 Default-Deny Network Policies

HIGH RISK

Kubernetes allows all pods to talk to all other pods by default. In a production environment with multiple services, this means a compromised pod can freely communicate with your database, your payment service, and your internal APIs. NetworkPolicy resources must exist in every namespace with a default-deny rule, with specific allow rules added only for required paths. Ask your team: "What is the default network behavior between pods in our production namespace?" Red flag: "They can all talk to each other" or "We haven't configured network policies yet."

10 Policy-as-Code (OPA Gatekeeper or Kyverno)

MEDIUM RISK

Human reviewers miss things. Policies enforced by admission controllers do not. Tools like Kyverno or OPA Gatekeeper intercept every deployment request and validate it against defined rules — blocking deployments that run as root, pull from unauthorized registries, or lack required cost-allocation labels. This is compliance that cannot be bypassed by a rushed Friday afternoon deployment. Ask your team: "What stops a developer from deploying a container that runs as root to production?" Red flag: "The pipeline checks for that" — because pipelines can be bypassed. The answer must involve an admission controller.

SOFTWARE SUPPLY CHAIN SECURITY

11 SBOM Generation on Every Build

MEDIUM RISK

A Software Bill of Materials is a complete inventory of every component inside your application — including third-party libraries and their versions. When a major vulnerability emerges (like Log4Shell did), organizations with SBOMs identified their exposure in hours. Organizations without them spent days manually checking every service. SBOM generation should be automated in the CI pipeline using tools like Syft. Ask your team: "If a critical vulnerability was published today in a common library, how long would it take us to know which of our services are affected?" Red flag: The answer is 'a few days' or 'we'd have to check manually.'

12 Container Image Scanning and Signing

HIGH RISK

Every container image deployed to production should be scanned for known CVEs using tools like Trivy or Snyk, and the pass/fail gate should be enforced in the pipeline — not advisory. Images should also be cryptographically signed (Cosign/Notary) to prove that what was built is what was deployed. Without signing, there is no guarantee. Ask your team: "Can a developer push an image with known critical vulnerabilities to production right now?" Red flag: Yes, or the scanning happens after deployment.

13 Dependency Pinning with Hash Verification

MEDIUM RISK

Version numbers like 'react@18.2.0' do not guarantee you are getting the same code every build. A compromised package registry could serve malicious code under a known version. Lock files (package-lock.json, go.sum, requirements.txt with hashes) pin dependencies to specific cryptographic hashes. These files must be committed to version control and respected in every build. Ask your team: "Are our lock files committed to the repository and enforced in the build?" Red flag: Lock files are gitignored, or dependencies are installed fresh without hash verification.

14 SLSA Build Integrity (Level 2 or Above)

MEDIUM RISK

SLSA (Supply-chain Levels for Software Artifacts) is a framework for build integrity. Level 2 requires that builds are scripted (not manual), the build service is authenticated, and provenance metadata is generated and stored. Most mature CI/CD setups already partially meet this — the audit determines whether it is intentional or coincidental. Ask your team: "Can we trace any given production binary back to the exact commit, build environment, and pipeline run that produced it?" Red flag: No audit trail exists between what is running in production and where it came from.

15 Branch Protection and Signed Commits

HIGH RISK

Direct commits to the main branch should be impossible — not just discouraged. Branch protection rules must require all changes via Pull Requests, with mandatory code owner approval. Requiring signed commits verifies the cryptographic identity of the author, preventing a compromised developer account from injecting code undetected. Ask your team: "Can anyone on the team push directly to main right now?" Red flag: Yes. Even if it rarely happens, the fact that it is possible is the vulnerability.

AUDIT DOMAIN B

B

Operational Resilience & Reliability

How well does your system survive failure — and how fast does it recover?

Every system fails eventually. The question your infrastructure must answer is not 'will it fail?' but 'when it fails, how long does it take to recover, and how much data do you lose?' The controls in this section determine the difference between a 20-minute incident and a 3-day crisis.

QUESTIONS IN THIS DOMAIN

10 items

DISASTER RECOVERY & BACKUP

16 Defined RPO and RTO Per Service Tier

HIGH RISK

Recovery Point Objective (maximum data loss) and Recovery Time Objective (maximum downtime) must be defined for each service tier before any DR architecture makes sense. Without these numbers, you cannot know whether your backup strategy is appropriate. The definition itself forces a business conversation: how much does one hour of downtime actually cost? Ask your team: "What are our RPO and RTO for the payment processing service?" Red flag: Nobody knows what RPO and RTO mean, or the answer is 'we try to minimize downtime.'

17 Immutable, Isolated Backups

HIGH RISK

Ransomware attacks in 2026 routinely target backup systems before encrypting production data. Backups must be immutable (WORM — Write Once, Read Many) and stored in an account or location that administrative access to the primary account cannot reach. If a compromised admin credential can delete your backups, your backups are not a recovery strategy. Ask your team: "If an attacker gained admin access to our primary cloud account, could they delete our backups?" Red flag: Yes, or nobody has thought about this.

18 Weekly Automated Restore Tests

HIGH RISK

A backup that has never been restored is a hypothesis, not a guarantee. Restore tests must be automated, run regularly (weekly is the 2026 benchmark), and validate data integrity — not just that a file was copied. The test must confirm the restored system is actually functional. Ask your team: "When was the last time we successfully restored from backup to a non-production environment and verified the data?" Red flag: The last restore test was more than 3 months ago, or was never done.

19 Zero Infrastructure Drift

MEDIUM RISK

If your infrastructure code (Terraform, Pulumi) no longer matches your actual infrastructure state, your disaster recovery plan will fail at the worst possible moment. Running terraform plan should return 'No Changes.' Drift is caused by manual changes made in the cloud console — the audit must confirm drift detection is automated and alerts are active. Ask your team: "What does 'terraform plan' return right now against production?" Red flag: Nobody knows, the plan hasn't been run recently, or there is a substantial diff.

20 Multi-AZ Redundancy for Critical Workloads

HIGH RISK

A single Availability Zone failure — which happens — should not cause an outage. Critical workloads must be distributed across at least three AZs. The audit specifically checks for hardcoded zonal dependencies: EBS volumes attached to a specific AZ, database read replicas in only one zone, load balancer configurations that do not span zones. Ask your team: "Which AZs are our production databases and primary application instances running in?" Red flag: Everything is in one AZ, or the team is not sure.

OBSERVABILITY & MONITORING

21 Golden Signals Coverage for Every Service

MEDIUM RISK

The four Golden Signals — Latency, Traffic, Errors, and Saturation — must be monitored at the service level, not just at the load balancer. Monitoring only the edge tells you that users are having a bad experience. Monitoring per-service tells you which service is causing it and why. Ask your team: "Do we have dashboards showing error rate and latency for each individual microservice?" Red flag: Monitoring exists only at the infrastructure level (CPU, memory) with no application-level visibility.

22 Centralized Log Retention (365+ days)

MEDIUM RISK

Logs scattered across individual servers are inaccessible when those servers are down — which is precisely when you need them most. Centralized log aggregation (ELK, Loki, Splunk, Datadog) with a minimum 365-day retention period is required for SOC 2 compliance and for meaningful incident investigation. Ask your team: "If we had an incident today, how far back could we query logs, and could we do it from a single interface?" Red flag: Logs are on individual servers, retention is under 90 days, or querying requires SSH access to multiple machines.

23 Alert Fatigue Management

MEDIUM RISK

Alert fatigue is not a minor inconvenience — it is a safety issue. When engineers are conditioned by false alarms to treat alerts as noise, the real alert that precedes a major outage gets the same response: ignored. Review your on-call rotation logs. High alert volume with low acknowledgement rates or high 'no-action' rates is the symptom. Ask your team: "How many alerts fired last week, and what percentage of them required human action?" Red flag: Alert volume is high, engineers mute channels regularly, or nobody tracks this metric.

24 Synthetic Monitoring on Critical User Paths

MEDIUM RISK

Passive monitoring waits for users to experience problems and generate errors. Synthetic monitoring actively tests your most critical user paths — login, checkout, key API endpoints — from external locations on a schedule. It catches outages before users do, and provides a clean baseline independent of actual traffic patterns. Ask your team: "If our checkout flow breaks silently at 3 AM, how long before we know?" Red flag: The honest answer is 'when users start complaining.'

25 Runbooks for Every Critical Alert

MEDIUM RISK

An alert without a linked runbook is just a notification that something is wrong. A runbook turns that notification into a step-by-step guide for triage and remediation. The goal is that a junior engineer receiving a 3 AM alert can resolve a known class of incident without waking up the one person who knows the system best. Ask your team: "For our top 5 most common alerts, is there a documented runbook that any engineer could follow?" Red flag: Runbooks exist in someone's head, are outdated, or do not exist at all.

AUDIT DOMAIN C

C

Knowledge Distribution & Bus Factor

How many people does your infrastructure's survival depend on?

The Bus Factor is the number of team members who, if suddenly unavailable, would cause your infrastructure to become unmanageable. A Bus Factor of 1 means your entire operation depends on one person's continued presence. This section quantifies that dependency before it becomes a crisis.

QUESTIONS IN THIS DOMAIN

5 items

26

Quantified Bus Factor Analysis

HIGH RISK

Analyze your Git commit history. If more than 50% of commits to core infrastructure modules in the last 6 months come from a single person, your Bus Factor is 1. This is not a technology problem — it is a business continuity risk. Quantify it before your board or insurer asks. Ask your team: "Who are the top three contributors to our infrastructure codebase in the last 6 months, and what percentage of changes did each make?" Red flag: One person is responsible for over 60% of infrastructure changes.

27

Hero Dependency in Incident Resolution

HIGH RISK

Review your incident logs for the last 6 months. If one person's name appears as the resolver for more than 50% of production incidents, you have a single point of failure wearing a human face. This 'Hero' dynamic feels like a strength until that person takes annual leave, resigns, or burns out. Ask your team: "Who resolved the last 10 production incidents?" Red flag: It is the same person in 7 out of 10 cases.

28

Documentation Updated Within 90 Days

MEDIUM RISK

Documentation older than 90 days in a fast-moving team is unreliable. It describes a system that may no longer exist in the form documented. Architecture documentation must be treated as a deliverable — updated alongside code changes, with ownership assigned. Undated wikis are a warning sign. Ask your team: "When was our core architecture documentation last updated, and who owns it?" Red flag: The last update timestamp is over 3 months ago, or ownership is unclear.

29

New Engineer Productive in Under 48 Hours

MEDIUM RISK

The time it takes for a new engineer to make their first production-safe deployment is one of the best proxies for overall infrastructure health. If onboarding takes weeks, the environment is too fragile and undocumented. If it takes under 48 hours, the fundamentals are solid. Ask your team: "How long did it take your last hire to make their first deployment?" Red flag: More than one week, or 'we are still setting them up' after a month.

30

CODEOWNERS Files Enforced in Git

LOWER RISK

CODEOWNERS files in Git repositories ensure that changes to critical paths require review from the designated domain experts. Beyond enforcing quality, they actively distribute knowledge — reviewers develop familiarity with systems they may not have built, reducing single points of expertise. Ask your team: "Can a new developer merge changes to our core Terraform modules without review from the infrastructure team?" Red flag: Yes, or there are no designated reviewers for critical infrastructure code.

AUDIT DOMAIN D

D

Pipeline & Release Velocity

CI/CD — the engine room of your engineering team

Your CI/CD pipeline is not a tool. It is the manufacturing process for your product. A poorly designed pipeline introduces risk, slows engineers, and creates the batching problem: slow pipelines encourage large, infrequent deployments, which are harder to debug when they go wrong. This section audits the pipeline as infrastructure.

QUESTIONS IN THIS DOMAIN

5 items

31 Build-to-Deploy Cycle Under 30 Minutes

MEDIUM RISK

A pipeline that takes longer than 30 minutes fundamentally changes developer behavior. Engineers stop waiting for pipeline results and context-switch to other work. When they come back, they have lost focus. Long pipelines encourage batching changes, which compounds risk. Target under 15 minutes for a fully tested deployment. Ask your team: "How long does a complete pipeline run take from code push to production deployment?" Red flag: Over 45 minutes, or 'it depends' in a way that suggests it regularly exceeds 60 minutes.

32 Automated Test Coverage Above 80%

MEDIUM RISK

Test coverage is not the goal — confidence in deployments is. High coverage (above 80%) gives engineers the assurance to deploy frequently without exhaustive manual testing. Gates in the pipeline must reject builds that fall below the threshold. A coverage number that nobody tracks is not coverage. Ask your team: "What is our current test coverage, and does the pipeline fail if it drops?" Red flag: Coverage is below 50%, unknown, or not enforced as a pipeline gate.

33 Same Artifact Deployed to Staging and Production

HIGH RISK

Rebuilding code for different environments — even if it seems like a minor variation — introduces the possibility that what you tested is not what you shipped. The same binary, the same container image, the same artifact must be promoted through environments. Environment-specific configuration is injected at runtime, not at build time. Ask your team: "Is the container image running in production the exact same image that was tested in staging?" Red flag: Code is rebuilt for each environment, or the team is not sure.

34 One-Click Rollback in Under 5 Minutes

HIGH RISK

The ability to undo a deployment quickly is more valuable than the ability to prevent all bugs. A deployment that breaks production and takes 45 minutes to roll back costs more than a deployment that breaks production and is reversed in 3 minutes. Rollback must be automated and health-metric-triggered, not a manual process. Ask your team: "Walk me through exactly what happens if a production deployment needs to be rolled back right now." Red flag: Rollback requires manual steps, takes more than 10 minutes, or requires the one person who owns the deployment process.

35 Staging Environment Mirrors Production Architecture

MEDIUM RISK

Environment drift between staging and production is the leading cause of 'it worked in staging' incidents. The same IaC definitions, the same managed service versions, and realistic data volumes (with appropriate anonymization) should exist in staging. If staging is a simplified version of production, it tests a simplified version of your problems. Ask your team: "What are the architectural differences between our staging and production environments?" Red flag: Staging uses different instance types, different managed service tiers, or significantly different configuration.

AUDIT DOMAIN E

E

Cloud Hygiene & Cost (FinOps)

Most companies overspend on cloud by 30%. This section finds out if you are one of them.

Cloud costs are uniquely deceptive. Unlike a data center, where hardware costs are visible and finite, cloud spending grows silently — unused resources accumulate, data transfer charges appear unexpectedly, and over-provisioned instances run unchallenged for months. FinOps is the discipline of making cloud spending visible and intentional.

QUESTIONS IN THIS DOMAIN

5 items

36 Resource Tagging Above 95% Coverage

MEDIUM RISK

Untagged cloud resources are ghost spend — costs that cannot be attributed to a team, project, or cost center. A minimum tagging strategy must enforce Environment (prod/dev/staging), Owner (team or system), and CostCenter tags on every resource. Without this, cost optimization is guesswork. Ask your team: "What percentage of our cloud resources currently have all required tags applied?" Red flag: Below 70%, or no mandatory tagging policy exists.

37 Regular Idle Resource Cleanup

MEDIUM RISK

Every cloud account accumulates waste over time: unattached EBS volumes, unused Elastic IP addresses, old snapshots, stopped instances still incurring charges. This waste is entirely invisible without active scanning. A monthly audit using cloud-native advisor tools or third-party cost management tools typically identifies 10-20% in recoverable costs. Ask your team: "When was the last time we ran a scan for unattached volumes, unused IPs, and old snapshots?" Red flag: Never, or more than 6 months ago.

38 Instance Rightsizing Reviewed Quarterly

MEDIUM RISK

Average CPU utilization below 5% across compute instances means resources are significantly over-provisioned. Cloud provider advisor tools (AWS Trusted Advisor, Azure Advisor) identify specific instances where rightsizing would reduce cost without impacting performance. The audit checks whether these recommendations are reviewed and acted on. Ask your team: "What is the average CPU utilization across our EC2/compute instances?" Red flag: Below 10% with no rightsizing review in the last 6 months.

39 Spot Instances for Non-Critical Workloads

LOWER RISK

Stateless workloads — CI/CD runners, batch processing, Kubernetes non-critical nodes — are ideal candidates for Spot Instances (AWS) or Preemptible VMs (GCP), which cost 60-90% less than on-demand. Organizations running more than 50% of non-critical workloads on Spot typically see 30-40% overall compute cost reduction. Ask your team: "What percentage of our CI runners and non-critical Kubernetes nodes run on Spot/Preemptible instances?" Red flag: Zero Spot usage for any workload, regardless of criticality.

40 VPC Endpoints to Avoid NAT Gateway Charges

LOWER RISK

NAT Gateway data transfer is one of the most consistent sources of unexpected cloud spend. Traffic from resources in private subnets to AWS services like S3 or DynamoDB routes through NAT Gateways by default — at a per-GB cost. VPC Endpoints create a private path to these services, eliminating that data transfer charge entirely. Ask your team: "Does traffic from our application pods to S3 route through a NAT Gateway or a VPC Endpoint?" Red flag: Through the NAT Gateway, or the team is not sure how data transfer is routed.

AUDIT DOMAIN F

F

Architecture & Scalability

Can your system handle 10x the load without 10x the engineering effort?

Scalability problems rarely appear at current traffic levels. They appear at 3x growth, when the team is celebrating success and simultaneously firefighting infrastructure that was never designed to scale. This section identifies architectural decisions that will constrain growth before they become the reason growth slows down.

QUESTIONS IN THIS DOMAIN

5 items

41

Stateless Application Servers

HIGH RISK

Application servers that store session data, uploaded files, or in-memory state locally cannot be horizontally scaled or placed on Spot Instances reliably. Statelessness is not just an architectural preference — it is a prerequisite for auto-scaling, high availability, and cost-effective infrastructure. Ask your team: "Can we add or remove application server instances at any time without affecting active user sessions?" Red flag: No — sessions are stored in memory on the server, or file uploads go to local disk.

42

Read/Write Splitting for Databases

MEDIUM RISK

A single database endpoint handling all reads and writes becomes a bottleneck predictably as traffic grows. Read replicas exist precisely to offload read-heavy traffic. The audit checks whether read replicas are provisioned and whether application code is configured to use them — not just whether they exist in the infrastructure. Ask your team: "Is our application configured to send read queries to a read replica?" Red flag: One database endpoint handles all traffic, or read replicas exist but are not used.

43

CDN for Static Content Delivery

LOWER RISK

Serving static assets — JavaScript, CSS, images — through application servers wastes compute and degrades user experience for geographically distributed users. A CDN (CloudFront, Cloudflare, Azure CDN) caches content at edge locations globally. The audit checks not just whether a CDN exists, but whether it is actually offloading the majority of static traffic. Ask your team: "What percentage of our total HTTP requests are served by the CDN versus the origin server?" Red flag: No CDN is configured, or the CDN exists but the cache-hit ratio is below 50%.

44

Asynchronous Processing for Long-Running Tasks

MEDIUM RISK

HTTP requests that take more than 2 seconds are a symptom of synchronous tight coupling — the web server is waiting for a background task to complete before responding. Report generation, email sending, image processing, and similar tasks must be decoupled via message queues (SQS, Kafka, RabbitMQ) and processed asynchronously. Ask your team: "Are there any API endpoints in our application that take more than 2 seconds to respond?" Red flag: Yes, and they are doing work that could be offloaded to a queue.

45

Auto-Scaling on Application Metrics, Not Just CPU

MEDIUM RISK

CPU is a lagging indicator. By the time CPU peaks, users are already experiencing degraded performance. Auto-scaling policies should be triggered by application-level metrics — request queue depth, requests per second, error rate — that indicate load before it saturates the compute layer. CPU-only scaling policies respond too slowly. Ask your team: "What metric triggers our auto-scaling policies?" Red flag: CPU only, or auto-scaling is not configured.

AUDIT DOMAIN G

G

Third-Party & Vendor Risk

The risk you did not build — but still own

Your infrastructure's security posture is only as strong as the vendors and dependencies within it. A zero-day in a widely-used library, a SaaS provider with inadequate security controls, or a vendor whose contract ended six months ago but whose access was never revoked — these are the risks that appear in incident reports with the phrase 'we didn't know.'

QUESTIONS IN THIS DOMAIN

5 items

46

Complete Vendor and Dependency Inventory

HIGH RISK

When a major vulnerability is published, the first question is: which of our services uses this? If the answer requires manually checking dozens of codebases and asking individual engineers, the answer comes too slowly. A maintained inventory of all SaaS dependencies and a generated SBOM for software dependencies enables a response measured in minutes, not days. Ask your team: "If a critical vulnerability was disclosed in a major database driver today, how would we know which services use it and how long would that take?" Red flag: The answer requires manually checking multiple repositories and asking individual engineers.

47

Regular Third-Party Access Reviews

HIGH RISK

Vendors that were granted access during an integration project — and then the project ended — frequently retain that access indefinitely. A quarterly access review should verify that every third-party with access to your systems is still under contract, still needs the access they have, and holds only the minimum permissions required. Ask your team: "When was the last formal review of all third-party access to our systems?" Red flag: Never, more than 12 months ago, or 'we trust them, they've been with us for years.'

48

Vendor SLA Alignment with Internal Commitments

MEDIUM RISK

If you promise your customers 99.9% uptime and your critical payment provider guarantees only 99.0%, you have committed to a standard that your vendor can legally prevent you from meeting. SLA alignment is not a procurement formality — it is a service design requirement. Ask your team: "What uptime do our critical third-party vendors contractually guarantee, and does that match what we promise our customers?" Red flag: Nobody has compared vendor SLAs against customer-facing commitments.

49

Data Residency and Sovereignty Compliance

HIGH RISK

GDPR, CCPA, and regional data protection regulations specify where customer data can be stored and processed. A misconfigured cloud region or a SaaS tool that defaults to US servers can put you in violation. The audit must verify the region configuration of every service that touches customer data. Ask your team: "In which geographic regions is our customer data stored, processed, and backed up?" Red flag: Nobody knows, or 'wherever the cloud provider defaults to.'

50

Shadow IT Detection

MEDIUM RISK

Engineers solve problems with whatever tools are available. Sometimes those tools are unapproved SaaS services being accessed via company accounts or networks. Shadow IT creates security blind spots, compliance gaps, and data sovereignty risks. A periodic review of network logs or expense reports typically reveals subscriptions and services that IT was unaware of. Ask your team: "Are there any cloud services or SaaS tools being used by the engineering team that are not in our official approved list?" Red flag: 'Probably, but we're not sure what' — which means there are.

SECTION 1.9 · INTERPRETING YOUR RESULTS

Scoring Guide — What Your Results Mean

Count the number of items you marked as Passed (P) in each section and overall. Mark Unknowns as Fails.

SCORE	INFRASTRUCTURE STATE	HIRE PROFILE	WHAT TO DO FIRST
40–50	**Production-grade**	Builder / Senior	Ready to hire. Focus on scaling and advanced automation.
30–39	**Functional with gaps**	Mid-Senior with IaC focus	Fix all HIGH items first. Hire someone who can independently remediate.
20–29	**Significant technical debt**	Senior / Principal fixer	Do not hire a builder. You need a fixer who can diagnose and remediate before growing.
Below 20	**Systemic risk**	External audit first	Hiring now accelerates burn, not output. Get an independent infrastructure assessment.

THE MOST IMPORTANT INSIGHT

If you have more than 3 HIGH RISK items marked as Failed or Unknown, the hiring decision is secondary. Those items represent real, present risk — not future concerns. Address them before onboarding someone new into an environment where those risks are their daily reality.

NEXT STEPS · WHAT HAPPENS AFTER

What Happens After the Audit?

You have the diagnosis. Here is what to do with it.

If you have gone through this checklist honestly, you now have a clearer picture of your infrastructure than most organizations this size have ever had. That clarity is valuable regardless of what it reveals.
The three most common situations people find themselves in after completing this audit:

1

Situation 1: Score above 40

Your infrastructure is in good shape. Hire a builder — someone who can take solid foundations and extend them. Focus on automation, developer experience, and scaling. We can help you define the right role and vet candidates against your specific stack.

2

Situation 2: Score 20–39

You have gaps that will compound as you grow. The right hire is a senior engineer who can both remediate existing issues and build forward. Before posting the role, let's talk about which HIGH risk items should be addressed first — some of them you can fix internally with the right guidance.

3

Situation 3: Score below 20

This is not the time to hire. This is the time to assess. A new engineer walking into this environment will spend their first months in reactive mode, which is demoralizing for them and expensive for you. A targeted infrastructure remediation engagement first will make the subsequent hire dramatically more effective.

Whichever situation you are in — we work with all three regularly.
Apex IT Solutions provides DevOps advisory, infrastructure remediation, and managed DevOps partnerships for software companies that want to fix what needs fixing, build what needs building, and hire with confidence. We are not a staffing agency. We are engineers who have built and operated production infrastructure, and we approach engagements the same way we approach this checklist: honestly.
If something in this guide raised a question you want to talk through, reach out. No sales process. Just a direct conversation with someone who has seen this before.
The Pre-Hire Infrastructure Audit Checklist | Apex IT Solutions | 2026

TALK TO US

Apex IT Solutions

Senior DevOps Engineering & Cloud Infrastructure Partnerships — Phone: 03350900718 — Email: sohaib@apexitsolutions.co — Website: www.apexitsolutions.co — DevOps: devops.apexitsolutions.co