The Guide to Audit-Ready Infrastructure

Thank you for downloading this Spacelift's Guide. Carahsoft is the distributor for Spacelift's DevSeOps solutions available via NASA SEWP V, ITES-SW2 and other contract vehicles.

To learn how to take the next step toward acquiring Tabnine's solutions, please check out the following resources and information:

For additional resources:
carah.io/SpaceliftResources

For additional Spacelift solutions:
carah.io/SpaceliftSolutions

To set up a meeting:
spacelift@carahsoft.com
571-590-6500

For upcoming events:
carah.io/SpaceliftEvents

For additional DevSecOps solutions:
carah.io/DevSecOpSolutions

To purchase, check out the contract vehicles available for procurement:
carah.io/SpaceliftContracts

spacelift

EBOOK

The Guide to Audit-Ready Infrastructure

Contents

The need for audit-ready infrastructure

Nobody looks forward to an audit. For DevOps and platform teams under pressure to ship, audits can seem like unjustified distractions. But infrastructure that can't prove compliance on demand is infrastructure that can't be trusted.

And if your infrastructure can't be trusted, it jeopardizes your entire organization. Robust infrastructure security protects sensitive data against cyberattacks and prevents the downtime, reputational damage, and financial losses they cause. It also ensures compliance with industry regulations and standards.

We wouldn't have audits, compliance checks, or security guardrails if infrastructure never broke down or was broken into. But infrastructure teams often treat these checks as irritants rather than safety nets woven from hard-won experience. These checks ensure that you have a documented audit trail that allows you to create a system that resists failure and adapts to it.

Audit-ready IaC is far more than a compliance checkbox; it's a litmus test for your infrastructure's trustworthiness. Security teams don't care how fast you ship if you can't show what changed, who changed it, and whether your infrastructure still complies.

This guide will explain why you're sabotaging your audit posture and walk you through how to build infrastructure your security team can trust, without breaking your sprint velocity.

Why teams fail audits

Audits demand clear visibility, but most homegrown CI/CD and IaC workflows are security black boxes. The result? Audit paralysis. Because traditional security tools focus on production monitoring, the infrastructure development lifecycle has blind spots that must be investigated manually. Security is considered only after or close to production, delaying remediation, increasing costs, and creating forensic gaps that DevOps leaders dread having to explain at audit time.

One of the big forensic gaps that DevOps leaders struggle to address is configuration drift. Without a way to automatically detect inconsistencies between actual infrastructure and the IaC definition, invisible discrepancies become untraceable liabilities. Even a minor manual tweak to your infrastructure can introduce configuration inconsistencies that can destroy your compliance posture. Drift may not always be bad — or even need to be fixed. But if it goes unrecorded, it creates one of those forensic blind spots that weaken your audit posture. Remember, your auditors don't trust your Terraform, so if you can't show why it has drifted, you're facing audit failure.

Truecar's Experience with Drift:

Whether they're approved changes that just need to be documented or changes outside the process, managing drift isn't easy. The SRE team at auto digital marketplace TrueCar knows this only too well. They soon ran into trouble with their first approach, splitting infrastructure concerns across separate repositories under one Terraform organization and using a minimalist pull request process with basic deployments. It led to persistent drift, state inconsistencies between branches, and ongoing configuration headaches.

Switching to a monorepo approach didn't solve the problem either: The team reorganized by folder within a single large repository, dividing infrastructure by AWS account (QA, staging, production, etc.). But instead of using pull requests, configuration changes had to be committed directly to master and deployed immediately, introducing new risks and failing to fix the drift problem at its root.

> "This might occasionally prevent stray configurations from sticking on some month-old branch, but it still prompted similar issues when configuration and actual resources did not match. A module reference might not have been updated in source control, but deployed out locally. Or code might be committed and never rolled out."
>
> **Yongjie Lim**
> Software Developer at TrueCar

And there's far more to DevOps teams' audit headaches than drift. Here are the other key weaknesses that undermine infrastructure security:

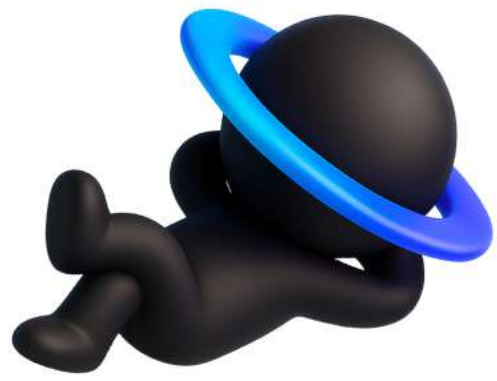
REASONS FOR AUDIT FAILURE	WHY THEY MAKE YOU FAIL
Misconfigurations	IaC misconfigurations can expose resources to the internet and increase the attack surface available to exploit vulnerabilities.
Hard-coded secrets	Adding secrets such as passwords, API keys, and tokens in the configuration code enables unauthorized access.

REASONS FOR AUDIT FAILURE	WHY THEY MAKE YOU FAIL
Lack of security tests	Not using an appropriate tool to assess your IaC configurations' vulnerabilities makes them exploitable.
Unencrypted data	Without encryption, information sent or received is stored in a readable format, making it vulnerable to unauthorized access.
Lack of audit trails	Without a clear record of who accessed what information and when, data integrity cannot be guaranteed, raising the risk of a breach.
Outdated and untrustworthy dependencies	Attackers often exploit known vulnerabilities in older versions of libraries, so failing to update can expose systems to breaches. You may also need to whitelist/blacklist dependencies.
Excessive privileges	Allowing unnecessary access to users expands the blast radius of any security incident, making it easier for attackers to move laterally, exfiltrate data, or cause widespread disruption.
Knowledge gaps	Infrastructure teams lack deep security expertise, and security teams lack infrastructure context. This knowledge gap leads to overly restrictive or permissive security policies. Deficiencies in infrastructure automation skills were the #1 people challenge cited in the **2025 Infrastructure Automation Leadership Index**
Lack of automated policy enforcement	Using manual code reviews means there is no automated way to prevent deployments that violate security standards. According to the **2025 Infrastructure Automation Leadership Index**, this is a top security/compliance challenge cited by 30% of respondents and 44% of infrastructure automation leaders.
Lack of approvals in workflow runs	Deploying your workflows without multiple reviews makes them prone to errors. Knowing who approved the runs helps to explain the reasoning behind the approval.

Faced with this array of security issues, DevOps leaders can quickly become so overwhelmed that they push audit preparation to the back burner. They scramble to find logs or recreate change histories, but their documentation is inadequate: Google Docs is no place for an audit trail, and tracking changes in a spreadsheet means compliance is already compromised. So the audit process drags on because these leaders must spend time they don't have explaining what happened, identifying gaps, and assembling proof.

Sound familiar? Diverting resources to meet audit requirements is an expense DevOps leaders can't afford. But audit failure sucks up even more of their team's valuable time. Their dilemma centers on how to keep shipping on schedule while also addressing the security concerns that undermine their infrastructure's trustworthiness.

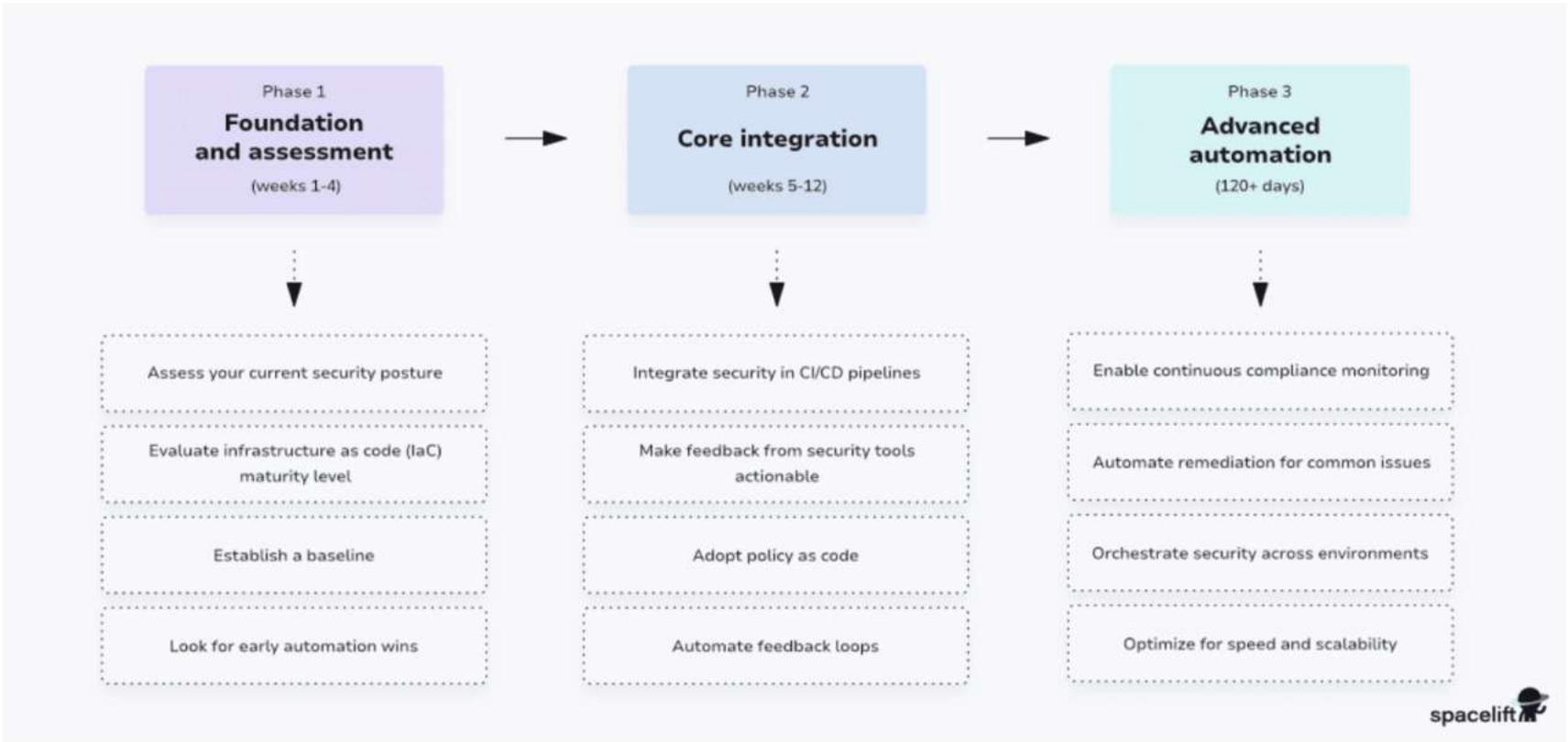
Audits without anxiety: A three-phase security implementation



There is a way to ensure audit readiness *and* maintain sprint velocity.

It involves adopting a shift-left approach to your infrastructure security, baking security into the start of the process rather than tacking it on only after or close to production. It means *continuous* compliance. No more waiting for audit season — everything's tracked, versioned, and exportable, so no nasty surprises. And it doesn't impede productivity because you don't need to invest more security effort; you simply move it to the area where it has the most impact.

Adopting a shift-left approach marks a sea change in your organization's security posture, so its success depends on a gradual introduction. The first step is a comprehensive assessment of your security tools, processes, and capabilities. You can then start integrating security into your development workflows. Finally, you scale your shift-left strategy with advanced automation.



Phase 1: Foundation (Week 1 to 4)

The move to shift-left security starts with an audit of your existing tools, processes, and capabilities:

- Which infrastructure security tools do you use now?
- How do you currently manage security validation, approvals, and compliance?
- Which elements of your security process are slowing deployments?

Assessment complete, move on to gauging your team's IaC maturity levels:

- Do you use IaC consistently across every environment?
- Does your team have the necessary experience in IaC and security tools?
- Is extra training needed?

Now, establish your starting position:

- How long does it take to identify and address security problems?
- What's your deployment frequency and lead time?

Countering cultural resistance to this new way of managing security is key, so build confidence and credibility for your strategy with quick wins, such as:

- Identifying repetitive security validation tasks
- Finding easy opportunities for automation
- Adding basic security scanners to CI/CD pipelines to identify vulnerabilities and misconfigurations before deployment.

Phase 2: Core integration (Week 5 to 12)

This stage involves moving from evaluation to actively embedding security into development workflows without breaking sprint velocity:

- Introduce automated security scanners to your CI/CD pipelines.
- Avoid breaking builds by starting with warning-only thresholds.
- As teams gain confidence, gradually increase enforcement.

Focus on actionable security tool feedback:

- Adjust scanning rules to prioritize high-impact, environment-specific risks.
- Avoid alert fatigue by tuning tools to eliminate false positives.
- Ensure developers get clear, actionable, and prioritized feedback.

Start using policy as code, a methodology that involves defining and enforcing policies through code stored and managed in a centralized policy engine. Use tools like OPA (Open Policy Agent), and expand policy coverage as your teams adapt.

POLICY AS CODE: ENFORCE CONTROLS WITHOUT SLOWING DELIVERY

Policies are vital for regulating environments with large-scale deployments, and policy as code makes it easier to enforce them. However, implementing and managing policy as code requires specific skills in areas like policy definition and automation. Integrating it with existing pipelines and other IT systems can also be complex.

Spacelift uses the open-source project Open Policy Agent and its rule language, Rego, to define policies. But you don't need to write Rego to create your policies because you can import modifiable templates from the Spacelift Policy Library. Read more about policy as code here.

"You could build Spacelift policies that are very restrictive of what a developer can do, so even in a PCI-regulated environment much like Checkout.com, you can still have teams that are empowered in their IaC to use these kinds of tools and innovate."

Joe Hutchinson
Director of Engineering – Developer Platform at Checkout.com

Start with core policies like:

- Data encryption
- Identity and access management (IAM)
- Network segmentation and firewall rules

Automate feedback loops so the right people receive security alerts at the right time:

- Link scanning tools to code review tools, chat platforms, and project management systems.
- Ideally, provide feedback at pull request time, so security is embedded in the development lifecycle.

Phase 3: Advanced automation (Week 13 to 24)

The final stage of your shift-left strategy is advanced automation.

Enable real-time compliance monitoring with:

- Automatic drift detection
- Automated alerts for critical issues
- Escalation paths that are defined according to severity

Resolve predictable issues automatically:

- Start with low-risk, high-impact actions.
- Move on to more complex resolutions (e.g., IAM adjustments, resource quarantining).

MONITOR CONTINUOUSLY FOR DRIFT AND COMPLIANCE VIOLATIONS

Monitoring systems need to go beyond initial deployment validation to ensure your security posture remains consistently compliant. Tools like AWS Config evaluate resource configurations against defined rules, trigger alerts when configurations drift from compliance standards, and can be set up to automatically fix simple security issues like unencrypted resources or overly permissive security groups. By building dashboards, you can enable real-time visibility into your security posture across all environments and then centralize security monitoring by implementing a tool like AWS Security Hub, which aggregates security findings from multiple sources. Find out more about drift detection here.

"Spacelift's drift detection means that even if anything falls between the cracks, we detect it within a reasonable time. And that on its own makes our day-to-day work a lot more predictable and fun."

Luca Hennart
Infrastructure Platform Engineer
at Picnic Technologies

Ensure a consistent security posture from development through production:

- Adopt security policies that adapt as environments require, while maintaining core security standards.
- Synchronize security standards across environments with orchestration tools.

Optimize automation for performance:

- Tweak scanning configurations to minimize unnecessary checks.
- Use parallel processing for security tasks.
- Refine feedback loops to provide faster insights without adding noise.

After six months of dedicated effort, you will have a solid, automated security strategy. Alternatively, you could adopt an infrastructure orchestration platform with all these security measures baked in. For example, the Spacelift platform incorporates cutting-edge security solutions that banish audit nightmares, such as a library of OPA policies, encryption, SSO, multifactor authentication (MFA), and private worker pools, so that you can reinforce every element of your IaC security strategy.

Formation Bio

"We took a well-established, policy-driven process and evolved it into an automated solution that maintains the same rigor but with greater speed and consistency."

Chris Niemira

Engineering Manager at Formation Bio

Audit readiness: A checklist for compliance

Visibility, control, and provability are the essential components of audit readiness. It's not enough to have all your security measures in place: You must also be able to show what changed, who changed it, and why — without spinning up a war room.

We've assembled a checklist of the practices required for trustworthy infrastructure and how you can automate them with Spacelift:

Implement SSO

Logging in using single sign-on (SSO) gives you centralized access to resources. More importantly, from a security perspective, it gives you greater oversight and access control. Spacelift supports SSO using either SAML 2.0 or OIDC. You'll find details on how to implement SSO with either method here.

Avoid hard-coded secrets

Secure injection and least-privilege defaults eliminate hardcoded secrets and insecure IAM configurations. This saves time chasing down access issues or postmortems after misconfigurations.

With Spacelift, you can add environment variables directly on the stack level or in a context that's attachable to multiple stacks. Adding them as plain text or secrets ensures only write operations are possible. You also have the flexibility to use Spacelift's lifecycle hooks to leverage a third-party secrets manager. If the tool supports it, you can even integrate the secrets manager via OIDC.

Enforce version control and peer-reviewed workflows

Store all infrastructure code in a version-controlled repository. Pull requests and code reviews can detect misconfigurations early, and a branching strategy will help you manage changes safely across environments.

Spacelift's push policies let you control what happens when a pull request is open or merged, and you can use approval policies to enforce your peer-reviewed workflows, so you can control how many approvals you need for runs and deny runs based on their configuration, if someone rejects that particular run.

Scan IaC templates for CI/CD misconfigurations

Static analysis tools like Checkov or Trivy automatically identify security risks during development. You should integrate these scans into your CI/CD pipelines to support secure-by-default practices.

Spacelift enables you to use lifecycle hooks to install and run tools, and you can add these hooks in contexts for reusability. You can also define all the commands necessary to install and run a tool like Checkov inside before auto-attaching the context to all your workflows. You can even customize the run behavior with custom policies based on Checkov's input.

Implement an audit trail

Audit trails ensure accountability and simplify compliance reporting. Spacelift offers an out-of-the-box audit trail so you can clearly understand the activity in the infrastructure your Spacelift instance manages. Using webhooks, you can also send all the data to your own system for long-term retention and integrations.

Apply the principle of least privilege to IAM roles and policies

Users and processes should only have the minimum necessary access rights to perform their tasks. Define specific identity and access management (IAM) policies using least privilege principles.

In Spacelift, you can organize resources within Spaces, logical containers that enable role-based access control (RBAC) and allow you to manage those resources in a hierarchical structure, similar to how Kubernetes resources are namespaced. Creating Spaces and isolating your resources enables you to implement least-privilege access to them and offer partial admin rights for a specific space.

Meeting the Requirements of Compliance Programs and Standards

Embedding a resilient IaC security strategy is a requirement for standards such as GDPR, HIPAA, or PCI DSS. Each standard has specific requirements, but there are parallels:

- You must identify where sensitive data is stored, processed, and transmitted within the organization and how it flows across networks, systems, and applications.
- You then need to implement appropriate security controls. These controls may include:
  - Network segmentation and firewalls to segregate sensitive data from less secure networks
  - Encryption of data in transit and at rest
  - Restricting access to sensitive data and implementing strong authentication mechanisms
  - Regularly monitoring and patching vulnerabilities to prevent breaches
  - Logging and monitoring to identify and respond to security incidents
- You must have clear policies and procedures for managing sensitive data.
- You must demonstrate employee awareness of the organization's compliance requirements and their responsibilities in maintaining data security and privacy.
- You need to regularly monitor your organization's compliance posture to surface opportunities for improvement and ensure ongoing compliance.
- Detailed documentation is crucial for all security controls, policies, procedures, and assessments.

CONTINUED ON NEXT PAGE →

Enforce policy as code

Use tools like OPA with rules to define and enforce security, compliance, and tagging through policy as code. This will ensure that non-compliant changes are automatically blocked or reverted.

Here are some factors Spacelift's policy types help you control:

- The resources engineers can create and their parameters, plus tagging enforcement
- The number of approvals required for runs and tasks
- What happens when a pull request is open or merged
- Where to send notifications — allowing you to integrate with tools like DataDog, Slack, MS Teams, Prometheus, and Grafana

Spacelift offers a policy library for easy integration of pre-defined policies in your workflows — no need to create any code!

Standardize and reuse secure modules

Pre-approved, reusable modules that enforce secure defaults for common resources (e.g., S3 buckets with encryption and logging, VPCs, etc.) reduce the potential for error, accelerate secure deployments, and ensure consistency.

Meeting the Requirements of Compliance Programs and Standards (continued)

Here are some specific requirements for each standard:

GDPR:

The General Data Protection Regulation (GDPR) centers on protecting the personal data of individuals within the EU. You must obtain consent for data processing, provide data subject rights (access, rectification, erasure), and implement appropriate security measures.

HIPAA:

The Health Insurance Portability and Accountability Act (HIPAA) is a US federal law that safeguards the privacy and security of protected health information (PHI). This involves administrative, physical, and technical measures to protect PHI.

PCI DSS:

The Payment Card Industry Data Security Standard (PCI DSS) ensures that all companies that accept, process, store, or transmit credit card information maintain a secure environment. To prove this, you will need to implement security controls for payment processing systems, networks, and applications.

With Spacelift's module registry, you can access features such as contexts, policies, and worker pools. You can also unlock tests to help ensure your module is working properly. You can even leverage lifecycle hooks to check the code for vulnerabilities. It is also recommended to use a plan policy to whitelist allowed modules or registries, ensuring only approved modules are used.

Encrypt data in transit and at rest by default

Encryption should be enabled for all data stores, message queues, and communication paths, and infrastructure modules should default to encryption. Automated scans validate that encryption settings are maintained.

SaaS Spacelift is hosted on AWS, with all data sources (S3, databases, SNS Topics, SQS Queues) encrypted at rest using AWS KMS keys with restricted and audited access. All data is also encrypted in transit, and all external traffic is handled using secure transport protocols.

Customer secrets are extra encrypted at rest, making even an internal attack likely to fail.

Keep dependencies and modules up to date

Rely on trusted sources for external modules, pin version numbers, and regularly update them to obtain the latest security patches. SBOM tools can also be useful for tracking and auditing the elements of your infrastructure code, to prevent introducing vulnerabilities through stale or compromised dependencies.

FedRAMP compliance

With its emphasis on standardized security practices in cloud environments, the Federal Risk and Authorization Management Program (FedRAMP) has huge implications for how IaC is used and secured within the U.S. federal government and its contractors. Cloud service providers must implement and manage specific security controls through IaC to ensure a consistent and repeatable security posture. Your infrastructure must comply with policies around:

- Access control
- System and information integrity
- Audit and accountability
- Risk assessment
- Incident response
- Configuration management

Ultimately, the shift to automated, repeatable, and auditable infrastructure management that FedRAMP requires aligns neatly with the hallmarks of a strong IaC security strategy.

Spacelift's registry shows all your modules and providers, their versions, and how to use them. This makes it easy to pin their version numbers and easily upgrade them to the latest version.

Integrating third-party tools for SBOM is easy with lifecycle hooks and contexts, as is defining custom policies for them.

Isolate environments and manage state separately

Dev, staging, and production should have separate configurations and backends. Segregating environments reduces the blast radius and enforces cleaner permission boundaries and resource scopes.

With Spacelift's Spaces, you can use the same code configuration and create clear and isolated workflows for different environments. You write the code just once, respecting the DRY principle and reducing the blast radius. Adding stacks to different spaces enables RBAC with least privilege.

Monitor for drift and continuously validate infrastructure

Drift detection options include terraform plan and CloudFormation's native drift detection.

Spacelift offers built-in drift detection and remediation that works on a schedule you define. You have full control over:

- Which stack should provide drift alerts
- Which stacks should automatically remediate
- Which stacks should run a remediation plan and wait for manual approval

Spacelift's dashboard provides an overview of the number and identity of drifted stacks, and Spacelift's IaC management view shows you everything that happened, including the exact resources that drifted.

> "The main takeaway for us [with Spacelift] is speed and auditability. On the infosec side, that is super important."
>
> **Robert Wlodarczyk**
> VP of Software Operations at Xealth

Final thoughts

If you're still using CI/CD pipelines to manage your infrastructure, security audits probably leave you feeling like a kid on Sunday night who hasn't done their homework. CI/CD was designed for app development, so cobbling together a series of workarounds to make your pipelines work for IaC will create forensic gaps that lead to audit paralysis. By adopting a shift-left approach to security and baking security into your sprints, you'll leave your auditors smiling and reinforce your infrastructure's resistance to failure — all while your team continues to ship uninterrupted, with no last-minute stoppages to fix security issues before a new feature or product is launched. Remember, your infrastructure's audit data isn't a checkbox; it's a competitive advantage.

Weaving security into your day-to-day operations requires less planning and investment if you adopt a specialized infrastructure orchestration platform with all the necessary features baked in. Letting a platform like Spacelift do the work ensures your infrastructure both resists failure and adapts to it, safeguarding the customers who rely on your organization and boosting productivity. Remember, you can do all the right things, but if you can't prove your infrastructure's compliance on demand, your security team won't believe you. Audit-ready infrastructure is *trustworthy* infrastructure.

1Password

"By using Spacelift's guardrails and security, we were able to confidently delegate much of IaC management to the individuals that owned it. This also allowed our teams to transform into service teams, not just remote hands for other teams and move from gatekeeping AWS to providing expertise."

Maxx Daymon

Staff Cloud Platform Engineer at 1Password

Sources:

Infrastructure as Code (IaC) Security: 10 Best Practices

The Infrastructure Automation Report, 2025

How Spacelift Can Improve Your Infrastructure Orchestration

What Makes Spacelift Secure

Drift Management in Cloud Infrastructure: Best Practices

What is Policy as Code (PaC) & How Do You Implement It?

Policy – Spacelift Documentation

Shifting Left in Infrastructure Security for Cloud-Native Teams

Navigating Kubernetes within the FedRAMP Guidelines

FedRAMP Compliance Guidelines for Container Security


spacelift