PLATFORM · TEMPLATE

Evidence Map: Which SOC 2 Controls Your Terraform *Already* Proves

KAANSYSTEMS.COM/LIBRARY/SOC2-EVIDENCE-MAP-TERRAFORM · OCTOBER 16, 2026

ABOUT THIS TEMPLATE

Trace SOC 2 controls from pull requests, pipelines and Terraform state to audit evidence, and see which controls infrastructure-as-code covers and which it can't.

THE TEMPLATE

The Evidence Map

One-page SOC 2 Evidence Map for a platform that already runs through Terraform and a gated pipeline. Trace each control to the artifact the repo produces. "Strong for configuration" means the code shows intent; a clean drift report is what shows that intent is running. An auditor decides whether that is sufficient.

SOC 2 AREA	WHAT THE AUDITOR WANTS TO SEE	WHERE IT LIVES IN TERRAFORM AND CI/CD	COVERAGE
CC8.1 Change management	Changes are authorized, tested, approved and implemented in a controlled way	PR history with required review, posted plan, pipeline apply logs, branch protection settings	Strong
CC6.1 Logical access	Access to infrastructure is restricted; encryption protects data	IAM roles and policies in code; KMS or Key Vault keys and encryption settings on storage and databases	Strong for configuration
CC6.2 / CC6.3 Granting and removing access	Access is approved, role-based and removed when no longer needed	Group and role membership in code (if managed there); PR history of grants and removals	Partial: identity provider and HR offboarding records still needed

SOC 2 AREA	WHAT THE AUDITOR WANTS TO SEE	WHERE IT LIVES IN TERRAFORM AND CI/CD	COVERAGE
CC6.6 Boundary protection	Network access from outside is restricted	VPC/VNet design, security groups, firewall rules, private endpoints, WAF in code	Strong
CC6.7 Data in transmission	Data is protected in transit	TLS policies on load balancers, cert-manager or managed certificates, HTTPS-only settings	Strong for configuration
CC6.8 Unauthorized software	Changes to software are controlled	Pinned images and versions, pipeline-only deploys, image scanning in CI	Partial
CC7.1 Detecting configuration changes	Configuration changes and vulnerabilities are detected	Scheduled drift reports, policy-as-code scans, config monitoring enabled in code	Strong
CC7.2 Monitoring	System components are monitored for anomalies	Audit logging (CloudTrail, Activity Log, Cloud Audit Logs), alerting and log retention in code	Partial: someone still has to respond
A1.2 Backup and recovery infrastructure	Backups and recovery infrastructure exist	Backup policies, retention, replication and snapshot schedules in code	Strong for configuration
A1.3 Recovery testing	Recovery plans are tested	Ability to rebuild an environment from code; DR test runbooks and results	Partial: the test itself must happen and be recorded
CC1–CC5, CC9 Governance, risk, people, vendors	Oversight, risk assessment, training, vendor management	Not in Terraform	None: policy and records

"Strong for configuration" is an important qualifier. Code shows what you *intended*. A clean drift report shows that intent is what's *running*. Auditors will want both, and how much they accept is their decision.

WHAT TO DO WITH THE MAP

Use it to answer three questions about your own environment:

01 **Which rows are "Strong" in theory but not in practice?** For example, IAM is in code, but half the roles were created in the console.

02 **Where is the exception path?** Who can still change production outside the pipeline, and how would you know?

03 **Which rows need a person, not a pipeline?** Assign an owner to each.

The answers also help with enterprise security questionnaires, which ask many of the same questions in different words. Reading an enterprise security questionnaire shows how to map those questions back to evidence you already have.