Vantalect

AWS SECURITY, COMPLIANCE & GOVERNANCE · SAMPLE DELIVERABLE

Audit Report

Week 2 of 6: Verify and plan · prepared for StrataForge Robotics, Inc.

Prepared by Charles Sieg · charles.sieg@vantalect.com · vantalect.com

July 24, 2026 · AWS Security, Compliance & Governance Audit: The Hardening (six weeks)

SAMPLE DELIVERABLE

This is a sample deliverable. StrataForge Robotics, Inc. is a fictional company: every account, resource, finding, figure, and person in this report is simulated to show the form and depth of a Vantalect engagement. AWS prices are list prices from the AWS Price List API on October 13, 2026.

Contents

Executive summary 2

Method 6

Deliverable 2: the findings register 7

Critical findings 12

High findings 25

Medium and Low findings 52

False positives 54

Deliverable 3: identity and access review 56

Deliverable 4: SOC 2 compliance map 66

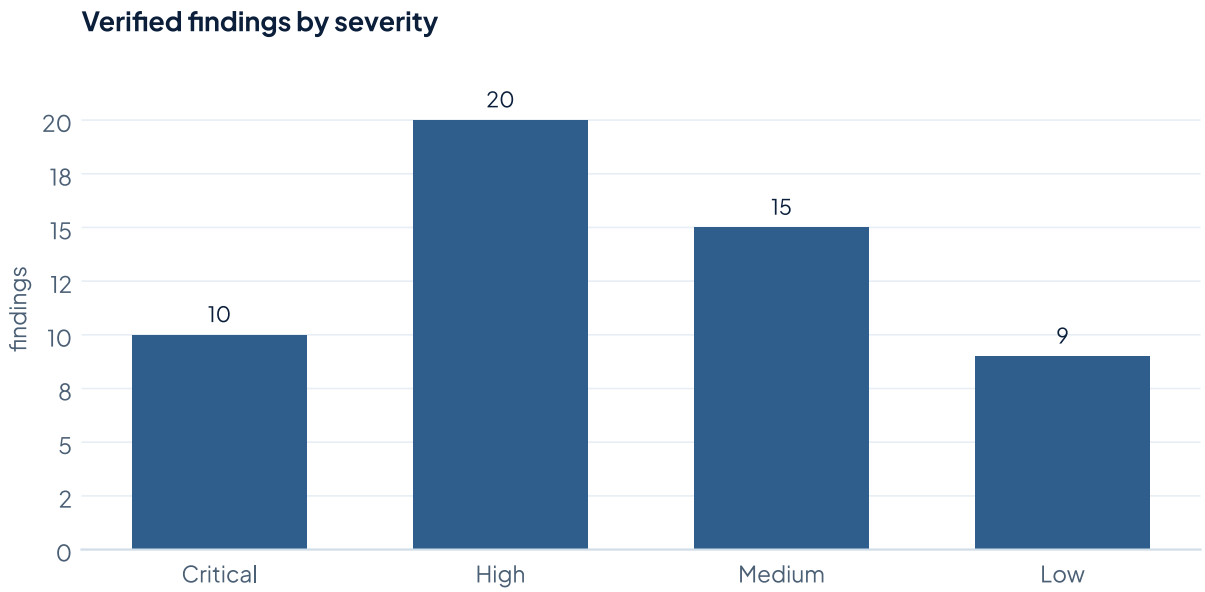
Deliverable 5: governance review 71

Deliverable 6: the remediation roadmap 77

Executive summary

We verified 54 findings in StrataForge's AWS organization: 10 Critical, 20 High, 15 Medium, and 9 Low. Marcus Oyelaran: this report delivers deliverables 2 to 6 of the Audit, and the sponsor readout (deliverable 7) puts the decisions it needs from you on July 24, 2026. Every Critical and High finding was checked by hand and every real finding was rated for how exploitable it is and how far it would reach. Of the scanner's results, 6 were wrong or intentional and are kept as false positives, with the reason, in False positives.

The 10 Critical findings are the ones that matter first. **6 of them need no credentials at all:** an attacker on the internet can read Kestrel's labeled customer-site images, restore a Kestrel database snapshot, reach Jupyter and Jenkins servers that run code on request, copy production volumes, and assume StrataForge deploy roles from a GitHub workflow of their own. **3 more need one stolen credential or one application bug,** and 1 needs a foothold inside the fleet. Of the 10, 2 reach robots in the field, which is why they carry a safety rating the benchmark does not.



Verified findings by severity.

Three things to know before the detail:

1. **The week-one score of 32.3% understates some risks and overstates others.** After the six false positives are removed the organization scores 32.4%, a small change, because most failures are real. But 5 findings are rated above their benchmark severity (`SF-SEC-001`, the OTA signing key, is Medium by benchmark and Critical here) and 15 are rated below it (`SF-SEC-041`, SSH and RDP open on sandbox instances with no production data, is Critical by benchmark and Medium here).
2. **The criticals are cheap to close.** All 10 are planned for week 3 and total 137 hours of engineering effort; the 20 High findings total 438 hours. The whole register is 1,165 hours, and 566 of them belong to the backlog after week 6 (Deliverable 6: the remediation roadmap).
3. **Identity is where the next breach would spread.** 1,009 of 1,139 active access keys are older than 90 days, 108 people sign in to the console without MFA, and we found 12 privilege-escalation paths to administrator (Deliverable 3: identity and access review).

The Critical findings at a glance

The Critical findings, in the order we recommend closing them. Every one is planned for week 3.

ID	Finding	What it takes	What it reaches
SF-SEC-001	A 1,127-day-old IAM user key can sign and publish firmware to every robot	Remote, with one precondition (stolen credentials or an application flaw)	A signed image in the distribution bucket reaches all 212,000 connected robots.
SF-SEC-002	One IoT policy lets any of 61,400 robots' certificates publish to any robot's command topic	Needs a foothold inside, an insider, or physical access	Command topics sf/v1/<thing>/cmd reach motion planning on every robot on the policy.
SF-SEC-003	4 GitHub Actions deploy roles can be assumed from any GitHub repository	Remote and unauthenticated, with a known technique	gha-artifact-publisher pushes the images every workload account deploys.
SF-SEC-004	Jupyter is open to the internet on 51 research and sandbox instances, some with no token	Remote and unauthenticated, with a known technique	Instance roles read the Crucible sensor-log lake and Kestrel model artifacts over IMDSv1.
SF-SEC-005	5 EBS snapshots in kestrel-prod are public, and snapshot block public access is off in all 816 account-Regions	Remote and unauthenticated, with a known technique	Volumes of the Kestrel perception API hosts, including application configuration.
SF-SEC-006	4 Kestrel dataset buckets are public: 468.8 million objects of labeled images from customer sites	Remote and unauthenticated, with a known technique	361 TB of images from customer sites; about 4% of a 200-image sample show people or badge IDs.
SF-SEC-007	A Kestrel MySQL snapshot with annotation records is restorable by any AWS account	Remote and unauthenticated, with a known technique	Annotator identities and customer-site labels for 1.9 million images.
SF-SEC-008	Kestrel's Jenkins controller is open to the internet on 8080 and 22	Remote and unauthenticated, with a known technique	Jenkins holds the Kestrel CI keys and a root access key for kestrel-legacy-mgmt.
SF-SEC-009	RDP is open to the internet on 32 PLM CAD workstations that hold product designs	Remote, with one precondition (stolen credentials or an application flaw)	Product designs for every robot line.

ID	Finding	What it takes	What it reaches
SF-SEC-010	Root users of 3 Kestrel accounts have access keys and no MFA	Remote, with one precondition (stolen credentials or an application flaw)	Root of kestrel-prod and kestrel-data, which hold the perception APIs and the labeled datasets.

What an attacker could reach

Read the register as paths, not as a list. Five paths matter most.

- **Through Kestrel.** The Kestrel Jenkins controller is open to the internet (SF-SEC-008). It holds the Kestrel CI keys and a root access key, and any Kestrel CI key can grant itself administrator and, through the pre-acquisition trust that still exists, administer four Kestrel accounts including `kestrel-prod` and `kestrel-data` (SF-SEC-018, SF-SEC-019). The data is already exposed without any of that: 4 dataset buckets and a database snapshot are public (SF-SEC-006, SF-SEC-007).
- **Through GitHub.** Any GitHub user can assume 4 StrataForge deploy roles from a workflow in their own repository (SF-SEC-003). One of them pushes the container images that every workload account deploys.
- **Through one key to the robots.** A single access key signs firmware and publishes it, and a signed image reaches every connected robot (SF-SEC-001). Separately, one certificate from any robot can publish to any robot's command topic (SF-SEC-002).
- **Through a notebook.** Jupyter is open to the internet on 51 research and sandbox instances, 1,631 instances still allow IMDSv1, and instance roles read the Crucible sensor-log lake and Kestrel model artifacts (SF-SEC-004, SF-SEC-014).
- **Through remote desktop.** RDP is open to the internet on the PLM CAD workstations that hold the product designs for every robot line (SF-SEC-009).

What we recommend, in order

1. **Week 3, starting July 27, 2026: close the 10 Criticals and put the guardrails in at the top of the organization,** so what closed stays closed: service control policies, a resource control policy that requires TLS, and an EC2 declarative policy (SF-SEC-026).
2. **Week 4, starting August 3, 2026: rebuild identity.** Retire old keys, require MFA, cut the 14 administrator policies to what each workload uses, and lock down root and break-glass access.
3. **Week 5, starting August 10, 2026: turn detection on everywhere.** GuardDuty, Security Hub CSPM, IAM Access Analyzer, Config, and Inspector in every account and Region, and an organization trail that no application account can alter.

4. **Week 6, starting August 17, 2026: re-scan** the same way as week 1, and measure.

Method

A finding here is a verified problem with an owner, an effort, and a fix, not a scanner line. This section says how a scanner line becomes one.

Where a severity starts

Every finding starts at the severity its source gives it: the Security Hub CSPM control's own severity, the AWS IoT Device Defender check's severity, or, for a problem no benchmark covers (a dangling DNS record, a trust policy with no `sub` condition), Vantalect's baseline for that class of problem. The source is printed with every finding.

How it is adjusted

A benchmark rates a control in the abstract. We rate the finding in StrataForge's estate. Each finding gets two adjustments, and then five written rules apply in order.

The two adjustments added to the benchmark severity (Low 1, Medium 2, High 3, Critical 4).

Adjustment	Values
Exploitability	Trivial +1 (remote and unauthenticated, with a known technique); moderate 0 (remote, with one precondition such as stolen credentials or an application flaw); limited −1 (needs a foothold inside, an insider, or physical access); theoretical −1 (a defense-in-depth gap with no direct path on its own)
Blast radius	Fleet +2 (robots in the field at customer sites); organization +1 (every account); production 0 (production systems or customer data); account −1 (one non-production account); contained −1 (a sandbox with no production data)

1. **Start at the benchmark severity** and add the two adjustments, then keep the result between Low and Critical.
2. **Critical needs a remote path and a wide reach.** It needs a trivial or moderate exploit and a production, organization, or fleet blast radius; otherwise the finding is capped at High.
3. **A theoretical exploit or a contained blast radius caps the finding at Medium.**
4. **A credible path to unsafe robot behavior in the field is always Critical.** A fleet-wide blast radius with a safety impact outranks every other rule.
5. **Order within a severity** is exploitability plus blast radius, plus two for safety impact, then the number of resources, so SF-SEC-001 is the highest risk and the last Low is the lowest.

The effect: 5 findings are rated above their benchmark, 15 below it, and 34 at it. Every printed finding shows its trail. SF-SEC-002 is the clearest example, because two rules act on it:

- benchmark Critical (4) -1 exploitability (limited) +2 blast radius (fleet) = 5
- rule 2: Critical needs a remote path and a production-or-wider blast radius; capped at High
- rule 4: safety impact on robots in the field; raised to Critical

What "verified by hand" means

Every Critical and High finding was verified by a Vantalect engineer, with the owner of the account or system where it was needed, before it was rated. Verification is one of four things, and each finding says which:

- **Read the configuration** that makes the weakness true and the configuration that would make it false (a bucket's policy and its public access block together, not either alone).
- **Reproduce it, where that is safe and approved in writing.** We assumed a deploy role from a Vantalect-owned test repository to prove the GitHub trust is open, and published from one robot's certificate to another on a bench robot with the fleet team.
- **Confirm intent with the owner.** A port open to the internet is a finding unless it must be open, and only the owner knows.
- **Trace the blast radius.** What the credential, role, or instance can reach, from policies and trust relationships.

We did not scan ports, exploit a server, or read any data. Where a finding says "no port scan was run", it means that. A result that verification shows to be wrong or intentional is not deleted. It stays in the register with the reason, who confirmed it, and the setting that makes the scanner stop reporting it, so the next scan does not raise it again and an auditor can see the decision (False positives).

Deliverable 2: the findings register

The register is the list of what to fix. It has 54 findings. Every Critical and High finding has its own section with its rating and how it got there, the evidence, how exploitable it is and how far it reaches, the controls and SOC 2 criteria it touches, an owner, a repository, an effort, how we verified it, and the fix: the steps, the key code, and how it is verified. Medium and Low findings are in a table. The complete code for every finding, with the commands and how to roll each change back, is in the companion document, Audit Report Appendix: Remediation Code, under the same finding ID. The same register is delivered as a spreadsheet, `findings-register.csv`, with one row per finding and every field in the tables below.

How to read a finding

- **Owner** is the person who approves the change and whose team merges it. **Repository** is where the change is made: `platform-landing-zone` (CloudFormation StackSets, service control policies, and account baselines for every account), `fleet-infra`, `crucible-infra`, or `enterprise-infra` (AWS CDK in TypeScript for those groups of accounts). Nothing is changed by hand in the console.
- **Effort** is the engineering time a team would spend doing the change by hand, from first commit to a verified close, including review and the soak in non-production. It is an estimate, and it is reset at each change review. In the Hardening (weeks 3 to 5), Vantalect writes the pull requests with agents, under review, inside its 110 hours; the owners review, merge, and roll out. Effort sizes the work. It is not the time StrataForge's engineers must find.
- **Remediation** gives the change: a summary, the steps in order, the key code, and how it is verified. The appendix has every code block, written to run once you replace the placeholders in angle brackets, such as `<account-id>`, which are the only values you fill in. Where a resource is not in code, the steps are a runbook of AWS CLI commands.
- **Rolling back.** Every change goes in as a pull request, so a rollback starts as a revert of the commit and a run of your pipeline. Each remediation in the appendix then says what a revert alone does not cover: the order to undo things in, and what cannot be undone by design, such as a deleted credential or a compliance-mode retention lock.

The register at a glance

Every verified finding, most severe first.

ID	Severity	Finding	Owner	Planned	Hours
SF-SEC-001	Critical	A 1,127-day-old IAM user key can sign and publish firmware to every robot	Sam Okafor	Week 3	24
SF-SEC-002	Critical	One IoT policy lets any of 61,400 robots' certificates publish to any robot's command topic	Sam Okafor	Week 3	40
SF-SEC-003	Critical	4 GitHub Actions deploy roles can be assumed from any GitHub repository	Elena Vasquez	Week 3	6
SF-SEC-004	Critical	Jupyter is open to the internet on 51 research and sandbox instances, some with no token	Hiro Tanaka	Week 3	10
SF-SEC-005	Critical	5 EBS snapshots in kestrel-prod are public, and snapshot block public access is off in all 816 account-Regions	Jonah Reyes	Week 3	6

ID	Severity	Finding	Owner	Planned	Hours
SF-SEC-006	Critical	4 Kestrel dataset buckets are public: 468.8 million objects of labeled images from customer sites	Jonah Reyes	Week 3	14
SF-SEC-007	Critical	A Kestrel MySQL snapshot with annotation records is restorable by any AWS account	Jonah Reyes	Week 3	3
SF-SEC-008	Critical	Kestrel's Jenkins controller is open to the internet on 8080 and 22	Jonah Reyes	Week 3	16
SF-SEC-009	Critical	RDP is open to the internet on 32 PLM CAD workstations that hold product designs	Mei Chen	Week 3	10
SF-SEC-010	Critical	Root users of 3 Kestrel accounts have access keys and no MFA	Grace Liu	Week 3	8
SF-SEC-011	High	docs-legacy.strataforge.example points at a deleted S3 website bucket that anyone can claim	Elena Vasquez	Week 3	3
SF-SEC-012	High	1,009 active access keys are older than 90 days; 400 are over three years old	Grace Liu	Week 4	63
SF-SEC-013	High	108 IAM users sign in to the console without MFA, 2 of them in the management account	Grace Liu	Week 4	22
SF-SEC-014	High	1,631 instances still allow IMDSv1	Elena Vasquez	Week 3	56
SF-SEC-015	High	144 Lambda functions have secret-named environment variables	Sam Okafor	Week 4	32
SF-SEC-016	High	All 16 EKS cluster API endpoints accept connections from 0.0.0.0/0	Sam Okafor	Week 3	38
SF-SEC-017	High	14 customer managed policies grant Action * on Resource * to 17 workload roles and 46 IAM users	Elena Vasquez	Week 4	40
SF-SEC-018	High	12 privilege-escalation paths lead to administrator in 11 accounts	Grace Liu	Week 4	28
SF-SEC-019	High	Kestrel's former management account still administers four Kestrel member accounts	Jonah Reyes	Week 4	6
SF-SEC-020	High	A former contractor's AWS account holds AdministratorAccess in mfg-mes-prod with no ExternalId	Mei Chen	Week 4	4
SF-SEC-021	High	The staging OTA test key lets the production pipeline sign, and pilot robots still trust it	Sam Okafor	Week 3	8

ID	Severity	Finding	Owner	Planned	Hours
SF-SEC-022	High	AWS Config records nothing in 570 account-Regions	Elena Vasquez	Week 5	10
SF-SEC-023	High	GuardDuty is off in 558 account-Regions, including every Kestrel account; most protection plans are off	Nadia Petrova	Week 5	12
SF-SEC-024	High	S3 Block Public Access is off at the account level in 31 accounts and on 47 buckets	Elena Vasquez	Week 3	10
SF-SEC-025	High	The 2019 organization trail can be stopped from the management account, and its logs can be deleted	Nadia Petrova	Week 5	20
SF-SEC-026	High	Service control policies stop nothing: only FullAWSAccess and DenyLeaveOrganization are attached	Elena Vasquez	Week 3	24
SF-SEC-027	High	SSH bastions accept connections from the whole internet	Elena Vasquez	Week 3	6
SF-SEC-028	High	SSH is open to the internet on 76 development instances	Sam Okafor	Week 3	18
SF-SEC-029	High	4 more member-account root users have no MFA; 40 use virtual MFA	Grace Liu	Week 4	6
SF-SEC-030	High	23 robot certificates are shared by 1,840 robots at 3 customer sites	Sam Okafor	Week 3	32
SF-SEC-031	Medium	1,247 Lambda functions run on deprecated runtimes	Sam Okafor	Backlog	143
SF-SEC-032	Medium	EBS encryption by default is off in 762 account-Regions; 1,245 volumes are unencrypted	Elena Vasquez	Week 3	78
SF-SEC-033	Medium	556 running instances are not managed by Systems Manager; 2,380 run AMIs over a year old	Elena Vasquez	Backlog	20
SF-SEC-034	Medium	39 internet-facing ALBs have no AWS WAF; 84 ALBs do not redirect HTTP to HTTPS	Sam Okafor	Backlog	16
SF-SEC-035	Medium	22 database instances store data unencrypted	Sam Okafor	Backlog	74
SF-SEC-036	Medium	9 ElastiCache replication groups lack encryption at rest or in transit	Sam Okafor	Backlog	33

ID	Severity	Finding	Owner	Planned	Hours
SF-SEC-037	Medium	5 EFS file systems are not encrypted at rest	Hiro Tanaka	Backlog	12
SF-SEC-038	Medium	Security Hub CSPM runs in only 3 Regions, with FSBP only and no CIS standard	Nadia Petrova	Week 5	8
SF-SEC-039	Medium	7 accounts are outside Control Tower, and the landing zone has drifted	Elena Vasquez	Week 5	20
SF-SEC-040	Medium	Break-glass access is not alarmed and was last tested in April 2025	Nadia Petrova	Week 4	8
SF-SEC-041	Medium	SSH and RDP are open to the internet on 27 sandbox instances	Elena Vasquez	Week 3	10
SF-SEC-042	Medium	Amazon Inspector is off almost everywhere; 132 ECR repositories have no image scanning	Nadia Petrova	Week 5	8
SF-SEC-043	Medium	750 default VPCs, most in Regions nobody uses, have open default security groups and network ACLs	Elena Vasquez	Week 5	16
SF-SEC-044	Medium	IAM Access Analyzer covers only us-east-1; 768 account-Regions have no external access analyzer	Nadia Petrova	Week 5	4
SF-SEC-045	Medium	31 databases keep one day of backups; 112 resilience checks fail	Sam Okafor	Backlog	12
SF-SEC-046	Low	130 customer managed policies use service-wide wildcards such as s3:*	Elena Vasquez	Backlog	40
SF-SEC-047	Low	2,850 of 14,464 resources have no Team tag, so no owner can be found for them	Elena Vasquez	Backlog	12
SF-SEC-048	Low	1,922 workload roles have not been used in 90 days or more	Grace Liu	Backlog	16
SF-SEC-049	Low	MFA delete is off on every bucket	Nadia Petrova	Week 5	4
SF-SEC-050	Low	Account baseline gaps: no security contact in 42 accounts, no support role in 20, CloudShell full access on 195 identities	Elena Vasquez	Week 3	10
SF-SEC-051	Low	Server access logging is off on 233 buckets	Elena Vasquez	Week 5	12

ID	Severity	Finding	Owner	Planned	Hours
SF-SEC-052	Low	220 buckets accept requests without TLS	Elena Vasquez	Week 3	6
SF-SEC-053	Low	78 customer managed keys do not rotate	Elena Vasquez	Week 3	4
SF-SEC-054	Low	EKS control-plane logging is off on 15 of 16 clusters, and every cluster still uses the aws-auth ConfigMap	Sam Okafor	Week 5	24

Critical findings

This section has 10 findings. Each is verified, and each is planned for week 3.

SF-SEC-001: A 1,127-day-old IAM user key can sign and publish firmware to every robot

Rating: Critical. Source: Security Hub CSPM IAM.3 on the key; raised for blast radius and safety. How it was rated: benchmark Medium (2) +0 exploitability (moderate) +2 blast radius (fleet) = 4 It has a safety impact on robots in the field.

Where: `fleet-ota-prod` (…5117) in us-east-1; 1 IAM user. **Plan:** Sam Okafor; `fleet-infra`; 24 hours; Week 3, starting Jul 27. **Controls:** IAM.2, IAM.3.

Fix. An access key more than three years old for `ota-signing-ci` can both sign firmware and publish it, so one stolen key reaches every robot in the fleet. Split signing and publishing into two short-lived roles behind a two-person approval, and let only the signer role use the key.

Steps

1. Create the GitHub environment `production` on `strataforge/ota-release` with two required reviewers, `prevent_self_review`, and a deployment branch policy that allows only `main`. GitHub needs one approval, and self-review prevention makes the approver a second person, which is the two-person rule.
2. Deploy the CDK stack that creates `ota-release-signer` (kms:Sign only, ECDSA_SHA_384) and `ota-release-publisher` (s3:PutObject on `releases/*` only), both trusting only `repo:strataforge/ota-release:environment:production`.
3. Change the release workflow to assume the signer role for the signing job and the publisher role for the upload job, with `id-token: write` and no stored AWS secrets.

4. Look up the key ID with `describe-key` (`put-key-policy` takes a key ID or key ARN, never an alias) and save the current key policy to the change record. Then apply the new key policy so kms:Sign is denied to everyone except the signer role.
5. Create the EventBridge rule in the key's Region with state `ENABLED_WITH_ALL_CLOUDTRAIL_MANAGEMENT_EVENTS`, because Sign is a read-only management event and an ordinary `ENABLED` rule never sees it, and add the target that pages the security team. Confirm one full release signs and publishes.
6. Deactivate the user's access key, wait one release cycle, then delete the key and the user.

Key code: ota-unexpected-sign-pattern.json: page on any Sign call that did not come from the signer role (EventBridge event pattern). Appendix SF-SEC-001: the other 3 blocks, evidence, SOC 2, rollback.

```JSON
{
  "source": [
    "aws.kms"
  ],
  "detail-type": [
    "AWS API Call via CloudTrail"
  ],
  "detail": {
    "eventSource": [
      "kms.amazonaws.com"
    ],
    "eventName": [
      "Sign"
    ],
    "userIdentity": {
      "arn": [
        {
          "anything-but": {
            "prefix": "arn:aws:sts::<fleet-ota-prod-account-id>:assumed-role/ota-release-
signer/"
          }
        }
      ]
    }
  }
}
```

How it is verified

- `aws iam list-access-keys --user-name ota-signing-ci` shows the key Inactive in fleet-ota-prod; after the next clean release the key and the user are deleted (the last step), and `aws iam get-user` then returns NoSuchEntity.
- A release run assumes both roles with no stored AWS secret, and CloudTrail shows every Sign call under `assumed-role/ota-release-signer`.

- A test kms:Sign call from any other role, an administrator included, is denied and raises the EventBridge page (check the rule's `State` is `ENABLED_WITH_ALL_CLOUDTRAIL_MANAGEMENT_EVENTS`).

SF-SEC-002: One IoT policy lets any of 61,400 robots' certificates publish to any robot's command topic

Rating: Critical. Source: AWS IoT Device Defender IOT_POLICY_OVERLY_PERMISSIVE_CHECK. How it was rated: benchmark Critical (4) -1 exploitability (limited) +2 blast radius (fleet) = 5; rule 2: Critical needs a remote path and a production-or-wider blast radius; capped at High; rule 4: safety impact on robots in the field; raised to Critical It has a safety impact on robots in the field.

Where: `fleet-ingest-prod` (…5012) in 4 Regions; 61,400 robot certificates. **Plan:** Sam Okafor; `fleet-infra`; 40 hours; Week 3, starting Jul 27. **Controls:** none; found by IOT_POLICY_OVERLY_PERMISSIVE_CHECK.

Fix. The IoT policy `sf-robot-legacy-v1` grants `iot:*` on `*` to every robot certificate, so one stolen certificate can read, publish to, or control the whole fleet (Device Defender IOT_POLICY_OVERLY_PERMISSIVE_CHECK). `sf-robot-scoped-v2` limits each robot to its own client ID, topics, shadow, and jobs, and `sf-robot-legacy-v1-restricted` does the same on the old topic tree for robots whose firmware has not moved yet.

Steps

1. Create `sf-robot-scoped-v2` and `sf-robot-legacy-v1-restricted` in each of the four IoT Regions: us-east-1, eu-central-1, ap-northeast-1, and ap-southeast-1.
2. The `${iot:Connection.Thing.ThingName}` variable resolves only when a robot connects with its thing name as the client ID, so check the connect logs of a bench robot and of each wave first. Put the robots that pass in `wave-1-things.txt` (one thing name per line), and leave out any that connect with another client ID until their firmware is fixed.
3. Create the wave's thing group, add every robot in `wave-1-things.txt` to it, and attach v2 to the group; IoT policies add up, so v2 changes nothing until v1 is detached.
4. Detach `sf-robot-legacy-v1` from the wave's certificates, skipping any certificate that more than one robot uses (SF-SEC-030 retires those), and watch connection failures for 24 hours before the next wave.
5. Robots whose firmware still publishes under `sf/v1/<thing>/` go into the thing group `sf-legacy-firmware` under `sf-robot-legacy-v1-restricted`: each robot keeps its own v1 topics, its shadow, and its jobs (the jobs topics deliver the firmware fix), every cloud command topic is closed to a robot certificate (these robots take commands only from the site's local controller), and no robot can reach another robot's topics. Record the residual risk as a signed acceptance with an end date.

6. Enable Device Defender audit in all four Regions on a daily schedule, so a permissive policy is found again if it returns.

Key code: Turn on Device Defender audit checks in all four Regions (AWS CLI). Appendix SF-SEC-002: the other 3 blocks, evidence, SOC 2, rollback.

```bash
CHECKS="IOT_POLICY_OVERLY_PERMISSIVE_CHECK DEVICE_CERTIFICATE_SHARED_CHECK
CONFLICTING_CLIENT_IDS_CHECK IOT_POLICY_POTENTIAL_MISCONFIGURATION_CHECK"
for region in us-east-1 eu-central-1 ap-northeast-1 ap-southeast-1; do
  aws iot update-account-audit-configuration --region "$region" \
    --role-arn arn:aws:iam::<fleet-ingest-prod-account-id>:role/DeviceDefenderAuditRole \
    --audit-check-configurations '{"IOT_POLICY_OVERLY_PERMISSIVE_CHECK": {"enabled": true},
"DEVICE_CERTIFICATE_SHARED_CHECK": {"enabled": true}, "CONFLICTING_CLIENT_IDS_CHECK":
{"enabled": true}, "IOT_POLICY_POTENTIAL_MISCONFIGURATION_CHECK": {"enabled": true}}'
  aws iot create-scheduled-audit --region "$region" --scheduled-audit-name sf-daily-iot-audit
--frequency DAILY --target-check-names $CHECKS
done
```

How it is verified

- `aws iot list-targets-for-policy --policy-name sf-robot-legacy-v1` returns only the certificates the cutover skipped as shared in each of the four Regions (SF-SEC-030 retires those), and `sf-robot-legacy-v1-restricted` is attached only to the `sf-legacy-firmware` thing group.
- A Device Defender audit shows no IOT_POLICY_OVERLY_PERMISSIVE_CHECK finding.
- A test certificate attached to `robot-a` cannot connect as client ID `robot-b` or publish to `sf/v2/robot-b/telemetry`.
- A certificate in `sf-legacy-firmware` cannot publish to or subscribe to any `sf/v1/<thing>/cmd` topic or reach another robot's topics, and its robot still receives its own jobs.

SF-SEC-003: 4 GitHub Actions deploy roles can be assumed from any GitHub repository

Rating: Critical. Source: Vantalect baseline; IAM Access Analyzer reports the federated principal as external access. How it was rated: benchmark High (3) +1 exploitability (trivial) +1 blast radius (organization) = 5

Where: `sf-cicd` (…1356), `fleet-api-prod` (…5044), `customer-portal-prod` (…5215), `fleet-staging` (…6014) in us-east-1; 4 IAM roles. **Plan:** Elena Vasquez; `fleet-infra`; 6 hours; Week 3, starting Jul 27. **Controls:** none; found by IAM Access Analyzer external access.

Fix. Four deploy roles trust `token.actions.githubusercontent.com` with only an `aud` condition, so a workflow in any GitHub repository can assume them. Add a `sub` condition that names the repository and environment.

Steps

1. List the repository and environment each role really serves, from 90 days of AssumeRoleWithWebIdentity events.
2. Write one trust policy per role with `aud` equal to `sts.amazonaws.com` and `sub` pinned with StringEquals (no wildcards).
3. Protect each GitHub environment (required reviewers for production) so the `environment` claim means something.
4. The four roles already exist, and CloudFormation cannot create a role whose name is taken, so first bring them under the stack with `cdk import` (the stack sets `RemovalPolicy.RETAIN` on each role, which an import requires); then `cdk deploy` writes the pinned trust policies. Save each role's current trust policy (`aws iam get-role --query Role.AssumeRolePolicyDocument`) before the import. Use the CLI command only for an emergency change, and reconcile the stack afterward.
5. Run each deploy workflow once and confirm it assumes the role; run a workflow from a scratch repository and confirm AccessDenied.

Key code: Trust policy for gha-deploy-fleet-api (IAM trust policy). Appendix SF-SEC-003: the other 2 blocks, evidence, SOC 2, rollback.

```JSON
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::<fleet-api-prod-account-id>:oidc-provider/
token.actions.githubusercontent.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
          "token.actions.githubusercontent.com:sub": "repo:strataforge/fleet-
api:environment:production"
        }
      }
    }
  ]
}
```

How it is verified

- Every trust policy shows both `:aud` and `:sub` conditions with StringEquals and no `*`.

- A workflow in a repository outside `strataforge` fails with `Not authorized to perform sts:AssumeRoleWithWebIdentity`.
- The production deploy workflow still succeeds, and CloudTrail shows its subject as `repo:strataforge/fleet-api:environment:production`.

SF-SEC-004: Jupyter is open to the internet on 51 research and sandbox instances, some with no token

Rating: Critical. Source: Security Hub CSPM EC2.19. How it was rated: benchmark Critical (4) +1 exploitability (trivial) +0 blast radius (production) = 5

Where: `crucible-research` (…7088), `kestrel-dev` (…9076), `sandbox-robotics` (…4033), `sandbox-ml` (…4056) in us-east-1; 51 instances. **Plan:** Hiro Tanaka; `crucible-infra`; 10 hours; Week 3, starting Jul 27. **Controls:** EC2.18, EC2.19, EC2.53, EC2.54.

Fix. Jupyter on port 8888 is open to 0.0.0.0/0 in crucible-research, kestrel-dev, and the sandbox accounts, and some servers were launched with an empty token, so anyone can run code on them. Close the port, rotate whatever the notebooks could read, and move researchers to port forwarding or SageMaker AI Studio.

Steps

1. Revoke the 8888 rules open to the internet in every account and Region.
2. Stop notebook servers that run with an empty token, and check their instance profiles and home directories for credentials; rotate anything found.
3. Review CloudTrail for the instance roles of the open notebooks for activity the researchers did not run.
4. Give researchers a Session Manager port-forwarding command for the servers they must keep.
5. Create a SageMaker AI Studio domain in private subnets for the rest, with an execution role scoped to the research data.

Key code: Revoke port 8888 from the internet in a Region (AWS CLI). Appendix SF-SEC-004: the other 2 blocks, evidence, SOC 2, rollback.

```bash
REGION=us-east-1
for sg in $(aws ec2 describe-security-groups --region $REGION \
    --filters Name=ip-permission.from-port,Values=8888 Name=ip-permission.cidr,Values=0.0.0.0/0
\
    --query 'SecurityGroups[].GroupId' --output text); do
  aws ec2 revoke-security-group-ingress --region $REGION --group-id "$sg" --protocol tcp --port
8888 --cidr 0.0.0.0/0
done
for sg in $(aws ec2 describe-security-groups --region $REGION \
    --filters Name=ip-permission.from-port,Values=8888 Name=ip-permission.ipv6-cidr,Values=::/0
\
    --query 'SecurityGroups[].GroupId' --output text); do
  aws ec2 revoke-security-group-ingress --region $REGION --group-id "$sg" \
    --ip-permissions 'IpProtocol=tcp,FromPort=8888,ToPort=8888,Ipv6Ranges=[{CidrIpv6=::/0}]'
done
```

How it is verified

- `aws ec2 describe-security-groups --filters Name=ip-permission.from-port,Values=8888 Name=ip-permission.cidr,Values=0.0.0.0/0` returns nothing in every account and Region.
- Security Hub CSPM EC2.19 passes after the next evaluation.
- A scan from outside the VPC finds nothing listening on port 8888 at any of the old addresses.

SF-SEC-005: 5 EBS snapshots in kestrel-prod are public, and snapshot block public access is off in all 816 account-Regions

Rating: Critical. Source: Security Hub CSPM EC2.1 and EC2.182. How it was rated: benchmark Critical (4) +1 exploitability (trivial) +0 blast radius (production) = 5

Where: All 48 accounts in 17 Regions; 5 EBS snapshots. **Plan:** Jonah Reyes; `platform-landing-zone`; 6 hours; Week 3, starting Jul 27; needs SF-SEC-026 first. **Controls:** EC2.1, EC2.182.

Fix. EBS snapshots in kestrel-prod are shared with everyone. Remove the public permission, then turn on snapshot block public access for every account with an EC2 declarative policy so it cannot happen again.

Steps

1. Find public snapshots in each Region of kestrel-prod and record what each contains.
2. Remove the `all` group from each snapshot's createVolumePermission.
3. Enable snapshot block public access with `block-all-sharing` in each Region; note that switching it back to `block-new-sharing` would make existing public snapshots public again.
4. Enable the declarative policy type, create the EC2 policy, and attach it to the root.
5. Treat the contents as disclosed until the owning team confirms otherwise.

Key code: Find and unshare public snapshots, then block sharing per Region (AWS CLI). Appendix SF-SEC-005: the other 2 blocks, evidence, SOC 2, rollback.

```bash
for region in us-east-1 eu-west-1; do
  for snap in $(aws ec2 describe-snapshots --region $region --owner-ids self --restorable-by-user-ids all \
                  --query 'Snapshots[].SnapshotId' --output text); do
    aws ec2 modify-snapshot-attribute --region $region --snapshot-id "$snap" \
      --attribute createVolumePermission --operation-type remove --group-names all
  done
  aws ec2 enable-snapshot-block-public-access --region $region --state block-all-sharing
  aws ec2 get-snapshot-block-public-access-state --region $region
done
```

How it is verified

- `aws ec2 describe-snapshots --owner-ids self --restorable-by-user-ids all` returns nothing in every Region of kestrel-prod.
- `aws ec2 get-snapshot-block-public-access-state` returns `block-all-sharing`.
- Security Hub CSPM EC2.182 passes in every account after the next evaluation.

SF-SEC-006: 4 Kestrel dataset buckets are public: 468.8 million objects of labeled images from customer sites

Rating: Critical. Source: Security Hub CSPM S3.2. How it was rated: benchmark Critical (4) +1 exploitability (trivial) +0 blast radius (production) = 5

Where: `kestrel-data` (…9098) in us-east-1; 4 S3 buckets. **Plan:** Jonah Reyes; `crucible-infra`; 14 hours; Week 3, starting Jul 27. **Controls:** S3.1, S3.2, S3.8.

Fix. The four dataset buckets the finding names in kestrel-data are open to the internet. Block public access first, remove the public policy and ACLs, and treat the contents as disclosed until the access history says otherwise.

Steps

1. Save each bucket's policy, ACL, and public access block settings to the incident folder as evidence before changing anything.
2. Turn on all four S3 Block Public Access settings on each bucket; that stops public reads at once without touching the objects.
3. Delete the public bucket policy, set the bucket ACL to private, and switch object ownership to BucketOwnerEnforced so ACLs stop working.

4. Put the TLS-only policy on each bucket; access for the Crucible training roles comes from their IAM policies, so the bucket policy needs no Allow.
5. Work out the exposure window: the organization trail has recorded S3 data events for these buckets since the Kestrel accounts joined in November 2025; before that, only the bucket creation dates are known. Give Legal the list of datasets and the outside IP addresses that read them.
6. Turn on server access logging for these buckets (see s3-logging) and move them into the crucible-infra CDK app with `cdk import`.

Key code: Evidence, then close the public buckets (AWS CLI). Appendix SF-SEC-006: the other 2 blocks, evidence, SOC 2, rollback.

```bash
for b in <dataset-bucket-1> <dataset-bucket-2> <dataset-bucket-3> <dataset-bucket-4>; do
  mkdir -p evidence/$b
  aws s3api get-bucket-policy --bucket $b > evidence/$b/policy.json 2>&1
  aws s3api get-bucket-acl --bucket $b > evidence/$b/acl.json
  aws s3api put-public-access-block --bucket $b \
    --public-access-block-configuration
BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets=true
  aws s3api delete-bucket-policy --bucket $b
  aws s3api put-bucket-ownership-controls --bucket $b --ownership-controls
'Rules=[{ObjectOwnership=BucketOwnerEnforced}]'
  sed "s/<dataset-bucket-name>/$b/g" tls-only.json > tls-only-$b.json
  aws s3api put-bucket-policy --bucket $b --policy file://tls-only-$b.json
done
```

How it is verified

- `aws s3api get-public-access-block` shows all four settings true on each of these buckets (Security Hub CSPM S3.1, S3.8).
- An anonymous `curl` of a known object URL returns 403.
- `aws s3api get-bucket-policy-status` reports `IsPublic` false and the ACL grants nothing to AllUsers or AuthenticatedUsers.

SF-SEC-007: A Kestrel MySQL snapshot with annotation records is restorable by any AWS account

Rating: Critical. Source: Security Hub CSPM RDS.1. How it was rated: benchmark Critical (4) +1 exploitability (trivial) +0 blast radius (production) = 5

Where: `kestrel-prod` (…9031) in us-east-1; 1 DB snapshot. **Plan:** Jonah Reyes; `crucible-infra`; 3 hours; Week 3, starting Jul 27. **Controls:** RDS.1.

Fix. The manual MySQL 5.7 snapshot `kestrel-annotations-2025-10-28-premigration` in kestrel-prod is shared with all AWS accounts (RDS.1). Remove the public attribute, then limit who can change snapshot sharing.

Steps

1. Remove the `all` value from the snapshot's `restore` attribute in every Region where it exists.
2. Ask the Perception Platform team whether the snapshot is still needed; if so, copy it with a customer managed key and delete the original after a hold.
3. Treat the data as disclosed for the time it was shared, and give Legal the date range from the snapshot creation date.
4. Attach `sf-protect-account-defaults` (see scp-guardrails); its ProtectRdsSnapshotSharing statement lets only the platform roles change snapshot sharing.
5. Deploy the Config rule that reports any public snapshot.

Key code: Remove public sharing and read the attribute back (AWS CLI). Appendix SF-SEC-007: the other 2 blocks, evidence, SOC 2, rollback.

```bash
REGION=us-east-1
aws rds modify-db-snapshot-attribute --region $REGION \
  --db-snapshot-identifier kestrel-annotations-2025-10-28-premigration \
  --attribute-name restore --values-to-remove all
aws rds describe-db-snapshot-attributes --region $REGION \
  --db-snapshot-identifier kestrel-annotations-2025-10-28-premigration \
  --query 'DBSnapshotAttributesResult.DBSnapshotAttributes'
aws rds copy-db-snapshot --region $REGION --source-db-snapshot-identifier kestrel-
annotations-2025-10-28-premigration \
  --target-db-snapshot-identifier kestrel-annotations-2025-10-28-premigration-private --kms-
key-id alias/sf-rds
aws rds wait db-snapshot-available --region $REGION --db-snapshot-identifier kestrel-
annotations-2025-10-28-premigration-private
```

How it is verified

- `describe-db-snapshot-attributes` shows an empty `restore` attribute value list.
- `rds:ModifyDBSnapshotAttribute` from an ordinary administrator role is denied by `sf-protect-account-defaults`.
- Security Hub CSPM RDS.1 passes in kestrel-prod after the next evaluation.

SF-SEC-008: Kestrel's Jenkins controller is open to the internet on 8080 and 22

Rating: Critical. Source: Security Hub CSPM EC2.19. How it was rated: benchmark Critical (4) +1 exploitability (trivial) +0 blast radius (production) = 5

Where: `kestrel-legacy-mgmt` (…9120) in us-east-1; 1 instance. **Plan:** Jonah Reyes; `crucible-infra`; 16 hours; Week 3, starting Jul 27. **Controls:** EC2.18, EC2.19, EC2.53.

Fix. The Kestrel Jenkins controller `i-0b89690e9c5782dbe` in kestrel-legacy-mgmt accepts 8080 and 22 from 0.0.0.0/0 and still allows IMDSv1. Close the ports now, require IMDSv2, rotate every credential Jenkins holds, and move the jobs to GitHub Actions or put the controller behind an internal load balancer with Okta sign-in.

Steps

1. Revoke the 0.0.0.0/0 rules on ports 8080 and 22 today, and require IMDSv2 on the instance.
2. Treat every credential stored in Jenkins as exposed: list the credentials store, the AWS keys, GitHub tokens, and SSH keys, and rotate each one.
3. Check CloudTrail for the Jenkins instance profile and any IAM user keys it holds for unfamiliar activity since the ports were opened.
4. Move the jobs to GitHub Actions with OIDC roles (see oidc-trust-sub), and shut down the controller when the last job has moved.
5. If the controller must stay for a while, put it behind an internal ALB that authenticates with Okta through OIDC, and allow ports only from the ALB.

Key code: Close the ports, require IMDSv2, and list the credentials to rotate (AWS CLI). Appendix SF-SEC-008: the other 1 block, evidence, SOC 2, rollback.

```bash
REGION=us-east-1
SG=$(aws ec2 describe-instances --region $REGION --instance-ids i-0b89690e9c5782dbe \
  --query 'Reservations[0].Instances[0].SecurityGroups[0].GroupId' --output text)
for port in 8080 22; do
  aws ec2 revoke-security-group-ingress --region $REGION --group-id "$SG" --protocol tcp --port
$port --cidr 0.0.0.0/0
done
aws ec2 modify-instance-metadata-options --region $REGION --instance-id i-0b89690e9c5782dbe \
  --http-tokens required --http-endpoint enabled --http-put-response-hop-limit 1
aws iam list-access-keys --user-name <jenkins-iam-user-name>
aws iam list-role-policies --role-name <jenkins-instance-role-name>
```

How it is verified

- `aws ec2 describe-security-groups --group-ids <jenkins-security-group-id>` shows no rule open to 0.0.0.0/0 or ::/0.
- `aws ec2 describe-instances` shows `HttpTokens` required for `i-0b89690e9c5782dbe` (Security Hub CSPM EC2.8).
- Every credential in the Jenkins credentials store has a rotation record, and the old values are rejected by their services.

SF-SEC-009: RDP is open to the internet on 32 PLM CAD workstations that hold product designs

Rating: Critical. Source: Security Hub CSPM EC2.19, EC2.53, EC2.54, and EC2.18. How it was rated: benchmark Critical (4) +0 exploitability (moderate) +0 blast radius (production) = 4

Where: `plm-prod` (…8061) in us-east-1; 3 security groups. **Plan:** Mei Chen; `enterprise-infra`; 10 hours; Week 3, starting Jul 27. **Controls:** EC2.18, EC2.19, EC2.53.

Fix. Security groups on bastions, PLM CAD workstations, and dev and sandbox instances allow SSH (22) and RDP (3389) from 0.0.0.0/0 and ::/0 (EC2.53, EC2.54, EC2.19, EC2.18). Revoke the rules, give instances the Systems Manager role, and reach them through Session Manager and Fleet Manager remote desktop.

Steps

1. Find every security group rule that opens 22 or 3389 to `0.0.0.0/0` or `::/0` in each account and Region.
2. Attach the Systems Manager managed instance role to the affected instances and confirm each shows as a managed node.
3. Revoke the open rules, IPv4 and IPv6 both, and keep a copy of each rule for rollback.
4. Give engineers `ssm:StartSession` on tagged instances through their permission set, and RDP users a port-forwarding session or Fleet Manager remote desktop.
5. Change the CDK security group constructs so no construct can open an administration port to the internet, then remove the bastion hosts that Session Manager replaces.

Key code: Find and revoke the open rules in one Region (AWS CLI). Appendix SF-SEC-009: the other 2 blocks, evidence, SOC 2, rollback.

```bash
REGION=us-east-1
for port in 22 3389; do
  for sg in $(aws ec2 describe-security-groups --region $REGION \
      --filters Name=ip-permission.from-port,Values=$port Name=ip-
permission.cidr,Values=0.0.0.0/0 \
      --query 'SecurityGroups[].GroupId' --output text); do
    aws ec2 revoke-security-group-ingress --region $REGION --group-id "$sg" --protocol tcp --
port $port --cidr 0.0.0.0/0
  done
  for sg in $(aws ec2 describe-security-groups --region $REGION \
      --filters Name=ip-permission.from-port,Values=$port Name=ip-permission.ipv6-
cidr,Values=::/0 \
      --query 'SecurityGroups[].GroupId' --output text); do
    aws ec2 revoke-security-group-ingress --region $REGION --group-id "$sg" \
      --ip-permissions "IpProtocol=tcp,FromPort=$port,ToPort=$port,Ipv6Ranges=[{CidrIpv6=::/
0}]"
  done
done
```

How it is verified

- `aws ec2 describe-security-groups --filters Name=ip-permission.from-port,Values=22,3389 Name=ip-permission.cidr,Values=0.0.0.0/0` returns nothing in every Region, and the same for `ip-permission.ipv6-cidr` with `::/0`.
- Security Hub CSPM EC2.53, EC2.54, EC2.19, and EC2.18 pass after the next evaluation.
- Engineers reach each instance through Session Manager, and the session appears in the audit log.

SF-SEC-010: Root users of 3 Kestrel accounts have access keys and no MFA

Rating: Critical. Source: Security Hub CSPM IAM.4. How it was rated: benchmark Critical (4) +0 exploitability (moderate) +0 blast radius (production) = 4

Where: `kestrel-prod` (…9031), `kestrel-data` (…9098), `kestrel-legacy-mgmt` (…9120) in us-east-1; 3 root users. **Plan:** Grace Liu; `platform-landing-zone`; 8 hours; Week 3, starting Jul 27. **Controls:** IAM.4, IAM.6, IAM.9.

Fix. Some Kestrel accounts have root access keys, some member accounts have no root MFA, and elsewhere root uses a virtual MFA device. Centralized root access management removes root credentials from every member account, so only the management account root needs a hardware MFA device.

Steps

1. From the management account, enable trusted access for IAM, then root credentials management and root sessions.
2. Audit each member account's root credentials with a short root session using the IAMAuditRootUserCredentials task policy.
3. Delete the root login profile, access keys, signing certificates, and MFA devices in each member account with the IAMDeleteRootUserCredentials task policy.
4. Deploy the EventBridge rule for sts:AssumeRoot so every root session pages the security team.
5. Register a FIDO2 hardware key on the management account root user in the console, remove the virtual device, and keep a second key at the second site.
6. Apply the member-account root deny SCP (see scp-guardrails) only after this cleanup finishes.

Key code: Enable centralized root access management (management account) (AWS CLI). Appendix SF-SEC-010: the other 2 blocks, evidence, SOC 2, rollback.

```bash
aws organizations enable-aws-service-access --service-principal iam.amazonaws.com
aws iam enable-organizations-root-credentials-management
aws iam enable-organizations-root-sessions
aws iam list-organizations-features
```

How it is verified

- `aws iam list-organizations-features` lists RootCredentialsManagement and RootSessions.
- In each member account, `aws iam get-account-summary` shows AccountAccessKeysPresent 0 (Security Hub CSPM IAM.4) and the root login profile is gone.
- Security Hub CSPM controls IAM.4, IAM.9, and IAM.6 pass in the management account once the hardware key is registered.

High findings

This section has 20 findings, 438 hours between them.

SF-SEC-011: docs-legacy.strataforge.example points at a deleted S3 website bucket that anyone can claim

Rating: High. Source: Vantalect baseline: subdomain takeover. How it was rated: benchmark Medium (2) +1 exploitability (trivial) +0 blast radius (production) = 3

Where: `sf-shared-services` (…1188) in us-east-1; 1 DNS record. **Plan:** Elena Vasquez; `platform-landing-zone`; 3 hours; Week 3, starting Jul 27. **Controls:** no benchmark control covers this.

Fix. `docs-legacy.strataforge.example` still points at an S3 website endpoint whose bucket was deleted in 2024, so anyone who creates a bucket with that name can serve content on a StrataForge subdomain. Delete the record, then sweep every hosted zone for records that point at AWS resources that no longer exist.

Steps

1. Delete the CNAME from the hosted zone in sf-shared-services; do not recreate the bucket, because that only hides the problem.
2. Sweep every hosted zone for CNAME and alias records that point at S3 website endpoints, CloudFront distributions, Elastic Beanstalk environments, and Elastic IP names.
3. Check each target exists; delete the record when it does not, and ask the owner when it is not clear.
4. Add the sweep to the monthly platform review, and require that a record be deleted in the same change that deletes its target.

Key code: Delete the dangling record (AWS CLI). Appendix SF-SEC-011: the other 1 block, evidence, SOC 2, rollback.

```bash
ZONE=<hosted-zone-id>
cat > delete-docs-legacy.json <<'JSON'
{"Comment": "Remove dangling S3 website record",
 "Changes": [{"Action": "DELETE", "ResourceRecordSet": {
   "Name": "docs-legacy.strataforge.example.", "Type": "CNAME", "TTL": <record-ttl>,
   "ResourceRecords": [{"Value": "docs-legacy.strataforge.example.s3-website-us-
east-1.amazonaws.com"}]}}]}
JSON
aws route53 change-resource-record-sets --hosted-zone-id "$ZONE" --change-batch file://delete-
docs-legacy.json
```

How it is verified

- `dig docs-legacy.strataforge.example CNAME` returns no answer.
- The sweep prints no DANGLING line for any hosted zone.

- The monthly review checklist includes the sweep, with the date of the last run.

SF-SEC-012: 1,009 active access keys are older than 90 days; 400 are over three years old

Rating: High. Source: Security Hub CSPM IAM.3, IAM.22, and IAM.2. How it was rated: benchmark Medium (2) +0 exploitability (moderate) +1 blast radius (organization) = 3

Where: All 48 accounts in us-east-1; 984 IAM users. **Plan:** Grace Liu; `platform-landing-zone`; 63 hours; Week 4, starting Aug 3; the rest (Replace the factory gateway users' keys with IAM Roles Anywhere during factory maintenance windows) is due November 20, 2026; needs SF-SEC-003, SF-SEC-008 first. **Controls:** IAM.2, IAM.3, IAM.22.

Fix. Many IAM users carry access keys older than 90 days or unused for 45 days or more, and some hold attached policies. Triage from the credential report, deactivate before deleting, replace CI keys with OIDC roles, and block new users and keys.

Steps

1. Generate the credential report in every account and list keys older than 90 days (IAM.3) and credentials unused for 45 days or more (IAM.22).
2. Deactivate keys unused for 45 days or more, identifying each key by its creation time (the credential report names a slot, not a key ID), and record `user,key-id` for every key you deactivate. Wait 14 days for anyone to object, then delete only the keys that are still Inactive.
3. For keys still in use, find the workload and replace it: CI keys with OIDC roles (see oidc-trust-sub), workload keys with roles.
4. Save every attached and inline user policy to disk, then detach or delete exactly what was saved, and grant access through groups or roles instead (IAM.2).
5. Attach `sf-protect-account-defaults` (see scp-guardrails); its NoNewIamUsers statement denies iam:CreateUser and iam:CreateAccessKey outside the PlatformIdentityAdmin role. Deploy the AWS Config rule that reports keys over 90 days.

Key code: Triage keys by age and last use from the credential report (AWS CLI). Appendix SF-SEC-012: the other 3 blocks, evidence, SOC 2, rollback.

```bash
aws iam generate-credential-report >/dev/null; sleep 5
aws iam get-credential-report --query Content --output text | base64 --decode > credential-report.csv
# GNU date first, then macOS date.
OLD=$(date -u -d '90 days ago' +%Y-%m-%dT%H:%M:%S+00:00 2>/dev/null || date -u -v-90d +%Y-%m-%dT%H:%M:%S+00:00)
UNUSED=$(date -u -d '45 days ago' +%Y-%m-%dT%H:%M:%S+00:00 2>/dev/null || date -u -v-45d +%Y-%m-%dT%H:%M:%S+00:00)
# Key 1: 9 active, 10 last_rotated, 11 last_used_date. Key 2: 14, 15, 16. A key never used counts as unused only once it is 45 days old.
awk -F, -v old="$OLD" 'NR > 1 && $9 == "true" && $10 < old {print $1 ",key1," $10} NR > 1 && $14 == "true" && $15 < old {print $1 ",key2," $15}' \
  credential-report.csv > keys-older-than-90-days.csv
awk -F, -v cutoff="$UNUSED" 'NR > 1 && $9 == "true" && (($11 == "N/A" && $10 < cutoff) || ($11 != "N/A" && $11 < cutoff)) {print $1 ",key1," $10} NR > 1 && $14 == "true" && (($16 == "N/A" && $15 < cutoff) || ($16 != "N/A" && $16 < cutoff)) {print $1 ",key2," $15}' \
  credential-report.csv > keys-unused-45-days.csv
wc -l keys-older-than-90-days.csv keys-unused-45-days.csv
```

How it is verified

- A fresh credential report shows no active key older than 90 days or unused for 45 days or more (IAM.3, IAM.22).
- `aws iam list-attached-user-policies` returns nothing for every user (IAM.2).
- `aws iam create-user` from an ordinary administrator role is denied by `sf-protect-account-defaults`.

SF-SEC-013: 108 IAM users sign in to the console without MFA, 2 of them in the management account

Rating: High. Source: Security Hub CSPM IAM.5. How it was rated: benchmark Medium (2) +0 exploitability (moderate) +1 blast radius (organization) = 3

Where: 23 accounts in us-east-1; 108 IAM users. **Plan:** Grace Liu; `platform-landing-zone`; 22 hours; Week 4, starting Aug 3; needs SF-SEC-010, SF-SEC-029 first. **Controls:** IAM.5.

Fix. IAM users, including some in the management account, have console passwords and no MFA. People should sign in through IAM Identity Center with Okta, so remove their login profiles and block new ones.

Steps

1. Pull the credential report and list every user with a password and no MFA.
2. Assign each person to the right IAM Identity Center permission sets through Okta groups, and have them sign in through the access portal.

3. Delete the IAM login profile of each person who has signed in through Identity Center at least once.
4. Attach `sf-protect-account-defaults` (see scp-guardrails); its NoNewIamUsers statement lets only the PlatformIdentityAdmin role create or change login profiles.
5. Keep passwords only for `breakglass-01` and `breakglass-02`, each with a hardware MFA device (see break-glass).

Key code: Find users with a password and no MFA from the credential report (AWS CLI). Appendix SF-SEC-013: the other 2 blocks, evidence, SOC 2, rollback.

```bash
aws iam generate-credential-report >/dev/null; sleep 5
aws iam get-credential-report --query Content --output text | base64 --decode > credential-report.csv
# Columns: 1 user, 4 password_enabled, 8 mfa_active.
awk -F, 'NR > 1 && $4 == "true" && $8 == "false" {print $1}' credential-report.csv > console-users-without-mfa.txt
wc -l console-users-without-mfa.txt
```

How it is verified

- A new credential report shows no user with `password_enabled` true and `mfa_active` false (Security Hub CSPM IAM.5).
- `iam:CreateLoginProfile` from an administrator who is not the identity administrator role is denied by `sf-protect-account-defaults`.
- The Identity Center sign-in log shows each moved person signing in through Okta.

SF-SEC-014: 1,631 instances still allow IMDSv1

Rating: High. Source: Security Hub CSPM EC2.8. How it was rated: benchmark High (3) +0 exploitability (moderate) +0 blast radius (production) = 3

Where: 40 accounts in 5 Regions; 1,631 instances. **Plan:** Elena Vasquez; `platform-landing-zone`; 56 hours; Week 3, starting Jul 27; the rest (Switch the instances that still make IMDSv1 calls after their agents and SDKs are upgraded) is due October 30, 2026. **Controls:** EC2.8.

Fix. Many instances still allow IMDSv1 (EC2.8), which lets a server-side request forgery bug read instance credentials. An EC2 declarative policy makes IMDSv2 the default for new launches, and existing instances are changed one at a time once the `MetadataNoToken` metric shows nothing uses v1.

Steps

1. Create the declarative policy for instance metadata defaults and attach it to the root; it applies to new launches only, and existing instances keep their settings.
2. For each existing instance, read the `MetadataNoToken` CloudWatch metric for 14 days; a nonzero sum means something still calls IMDSv1.
3. Update SDKs and agents that use IMDSv1, then run `modify-instance-metadata-options --http-tokens required`; use a hop limit of 2 for containers that read metadata.
4. Set the account default per Region with `modify-instance-metadata-defaults` as a second layer.
5. Set `requireImdsv2` in every CDK instance and launch template construct.

Key code: sf-ec2-imdsv2-default: IMDSv2 required for new launches, hop limit 2 (EC2 declarative policy). Appendix SF-SEC-014: the other 2 blocks, evidence, SOC 2, rollback.

```JSON
{
  "ec2_attributes": {
    "instance_metadata_defaults": {
      "http_tokens": {
        "@@assign": "required"
      },
      "http_put_response_hop_limit": {
        "@@assign": "2"
      }
    }
  }
}
```

How it is verified

- `aws ec2 describe-instances --query 'Reservations[].Instances[?MetadataOptions.HttpTokens!=required].InstanceId'` returns an empty list in every account and Region (Security Hub CSPM EC2.8).
- A new instance launched without any metadata option comes up with `HttpTokens` required.
- The `MetadataNoToken` metric is zero for changed instances, and no application error appears in the week after.

SF-SEC-015: 144 Lambda functions have secret-named environment variables

Rating: High. Source: Vantalect baseline: secrets in plaintext configuration. How it was rated: benchmark High (3) +0 exploitability (moderate) +0 blast radius (production) = 3

Where: `fleet-api-prod` (…5044), `customer-portal-prod` (…5215), `fleet-dev` (…6037), `kestrel-prod` (…9031) in 2 Regions; 144 Lambda functions. **Plan:** Sam Okafor; `fleet-infra`; 32 hours; Week 4, starting Aug 3; the rest (Move the fleet-dev functions' secrets to Secrets Manager) is due September 30, 2026. **Controls:** no benchmark control covers this.

Fix. Lambda functions keep `DATABASE_URL`, `PARTNER_API_KEY`, `GITHUB_TOKEN`, `DB_PASSWORD`, or `SAP_RFC_PASSWORD` in plain environment variables, readable by anyone with lambda:GetFunctionConfiguration. Store them in Secrets Manager, grant read to the function role, and rotate every value that was exposed.

Steps

1. List the functions and the names of their secret-named variables (names only; never print values).
2. Create one Secrets Manager secret per credential in the owning account, encrypted with a customer managed key.
3. Change the function code to read the secret at start-up and cache it, and pass only the secret's name in the environment.
4. Grant the function role read access on that one secret with `grantRead` in CDK.
5. Rotate every credential that lived in an environment variable, because it has been readable since it was set, then remove the old variables.

Key code: List the secret-named variables, without their values (AWS CLI). Appendix SF-SEC-015: the other 2 blocks, evidence, SOC 2, rollback.

```bash
for region in us-east-1 us-west-2; do
  aws lambda list-functions --region "$region" \
    --query 'Functions[?Environment.Variables!=`null`].{name: FunctionName, vars:
keys(Environment.Variables)}' --output json |
    jq -r --arg region "$region" '.[] | select(.vars | map(test("PASSWORD|SECRET|TOKEN|API_KEY|
DATABASE_URL|PRIVATE_KEY"; "i")) | any) | [$region, .name, (.vars | join(";"))] | @csv'
done > lambda-secret-variables.csv
wc -l lambda-secret-variables.csv
```

How it is verified

- Re-running the inventory returns no function whose secret-named variable holds a value instead of a secret name.
- `aws lambda get-function-configuration` for a migrated function shows only a secret name in its environment.
- The old credential values are rejected by the partner, GitHub, and the database, which proves the rotation happened.

SF-SEC-016: All 16 EKS cluster API endpoints accept connections from 0.0.0.0/0

Rating: High. Source: Security Hub CSPM EKS.1. How it was rated: benchmark High (3) +0 exploitability (moderate) +0 blast radius (production) = 3

Where: 10 accounts in 5 Regions; 16 EKS clusters. **Plan:** Sam Okafor; `fleet-infra`; 38 hours; Week 3, starting Jul 27; the rest (Make the non-production endpoints private once CI runners reach them through the Transit Gateway) is due September 30, 2026. **Controls:** EKS.1.

Fix. EKS clusters expose a public API endpoint open to 0.0.0.0/0 (EKS.1). Turn on private endpoint access, limit the public endpoint to the corporate egress range, and move to private-only once CI runs inside the VPC.

Steps

1. List every cluster in every account and Region and its `resourcesVpcConfig` endpoint settings.
2. Turn on private access first, so nodes keep reaching the API server inside the VPC.
3. Set `publicAccessCidrs` to the corporate egress range (get the real CIDR from the network team; the examples use the `<corporate-egress-cidr>` placeholder); the update takes several minutes per cluster.
4. Move deploy jobs onto runners inside the VPC (the sf-cicd account), then set `endpointPublicAccess` to false.
5. Set the endpoint access in the CDK cluster constructs so a rebuild keeps it.

Key code: Private access on, public access limited to the corporate range (AWS CLI). Appendix SF-SEC-016: the other 1 block, evidence, SOC 2, rollback.

```bash
REGION=us-east-1
for cluster in $(aws eks list-clusters --region $REGION --query 'clusters[]' --output text); do
  aws eks update-cluster-config --region $REGION --name "$cluster" \
    --resources-vpc-config
endpointPrivateAccess=true,endpointPublicAccess=true,publicAccessCidrs=<corporate-egress-cidr>
  aws eks wait cluster-active --region $REGION --name "$cluster"
done
# When CI runs inside the VPC, close the public endpoint:
aws eks update-cluster-config --region $REGION --name <cluster-name> \
  --resources-vpc-config endpointPrivateAccess=true,endpointPublicAccess=false
```

How it is verified

- `aws eks describe-cluster --name <cluster-name> --query cluster.resourcesVpcConfig` shows `endpointPrivateAccess` true and `publicAccessCidrs` of only the corporate egress range, or `endpointPublicAccess` false.
- Security Hub CSPM EKS.1 passes after the next evaluation.
- Nodes stay Ready and the deploy pipeline still reaches the cluster.

SF-SEC-017: 14 customer managed policies grant Action * on Resource * to 17 workload roles and 46 IAM users

Rating: High. Source: Security Hub CSPM IAM.1. How it was rated: benchmark High (3) +0 exploitability (moderate) +0 blast radius (production) = 3

Where: 14 accounts in us-east-1; 14 IAM policies. **Plan:** Elena Vasquez; `fleet-infra`; 40 hours; Week 4, starting Aug 3. **Controls:** IAM.1.

Fix. Workload roles carry `"Action": "*"` (IAM.1) and customer managed policies with service wildcards such as s3:* (IAM.21), including the EKS node role `fleet-api-node-role`. Generate least-privilege policies from CloudTrail with IAM Access Analyzer and cap each role with the one `sf-workload-boundary` permissions boundary (defined in the privilege-escalation entry), which allows every service action and takes away only identity and organization actions.

Steps

1. List roles with an `Action: *` statement and customer managed policies with `service:*` actions, and rank them by production impact.
2. Start an IAM Access Analyzer policy generation for each role from the organization trail's last 90 days.
3. Review the generated policy with the owning team, add the actions CloudTrail does not record, and replace the wildcard policy with it.
4. Move pod permissions off the node role by giving each workload its own role through EKS Pod Identity.
5. `fleet-api-node-role` already exists, and CloudFormation cannot create a role whose name is taken, so bring it under the stack with `cdk import` (the stack sets `RemovalPolicy.RETAIN` on it, which an import requires) before the first `cdk deploy`.
6. Publish `sf-workload-boundary` (see privilege-escalation; it is one document for every role) and attach it. In a non-production account first, run `simulate-principal-policy` for the VPC CNI's network-interface and IP actions and for `eks-auth:AssumeRoleForPodIdentity`: the boundary must leave all of them allowed, or new pods get no IP address and Pod Identity stops working.

Key code: Generate a policy from 90 days of CloudTrail for fleet-api-node-role (AWS CLI). Appendix SF-SEC-017: the other 3 blocks, evidence, SOC 2, rollback.

```bash
cat > trail-details.json <<'JSON'
{"trails": [{"cloudTrailArn": "arn:aws:cloudtrail:us-east-1:<management-account-id>:trail/sf-org-trail", "allRegions": true}],
 "accessRole": "arn:aws:iam::<fleet-api-prod-account-id>:role/AccessAnalyzerMonitorServiceRole",
 "startTime": "<start-timestamp-iso-8601>", "endTime": "<end-timestamp-iso-8601>"}
JSON
JOB=$(aws accessanalyzer start-policy-generation \
  --policy-generation-details principalArn=arn:aws:iam::<fleet-api-prod-account-id>:role/fleet-api-node-role \
  --cloud-trail-details file://trail-details.json --query jobId --output text)
aws accessanalyzer get-generated-policy --job-id "$JOB" --include-resource-placeholders --include-service-level-template
```

How it is verified

- Security Hub CSPM IAM.1 and IAM.21 show no failed resource in fleet-api-prod.
- The generated policy replaced the wildcard policy, and the service ran a full release cycle with no new AccessDenied errors in CloudTrail.
- `aws iam get-role --role-name fleet-api-node-role` shows `PermissionsBoundary` set, `simulate-principal-policy` allows the CNI and Pod Identity actions, and a new pod in a non-production cluster gets an IP address.

SF-SEC-018: 12 privilege-escalation paths lead to administrator in 11 accounts

Rating: High. Source: Vantalect privilege-escalation path analysis. How it was rated: benchmark High (3) +0 exploitability (moderate) +0 blast radius (production) = 3

Where: 11 accounts in us-east-1; 12 paths. **Plan:** Grace Liu; `platform-landing-zone`; 28 hours; Week 4, starting Aug 3. **Controls:** no benchmark control covers this.

Fix. Several roles can raise their own privileges: iam:PassRole with lambda:CreateFunction or ec2:RunInstances, iam:CreatePolicyVersion, iam:PutUserPolicy on themselves, and iam:UpdateAssumeRolePolicy. One boundary, `sf-workload-boundary`, allows every service action except IAM, Organizations, and Account, adds back only a scoped iam:PassRole, and denies the escalation actions and any change to itself. An SCP forces the boundary onto new roles and protects it, and Identity Center permission sets take it through a permission-set boundary.

Steps

1. Run `simulate-principal-policy` for the escalation actions against every role the scan flagged, to confirm each path.
2. Publish `sf-workload-boundary` as the same document everywhere: a service-managed stack set for every member account, and a stack of its own in sf-management, because service-managed stack sets never deploy to the management account. Its first statement allows everything except `iam:*`, `organizations:*`, and `account:*`, so the VPC CNI, EKS Pod Identity, SSM, IoT, SageMaker, and Kinesis keep working.
3. Scope every remaining iam:PassRole to named `sf-workload-*` roles with `iam:PassedToService`, and never `Resource: *`.
4. Attach the boundary to the flagged IAM roles and users with `put-role-permissions-boundary`. The roles behind Identity Center (`AWSReservedSSO_`) reject that call, so attach it to the permission set instead with `sso-admin put-permissions-boundary-to-permission-set`, and re-provision the permission set.
5. Set the boundary on every CDK app (`PermissionsBoundary.of(app).apply(...)`) so roles the pipelines create carry it, then attach `sf-require-workload-boundary`, which makes new roles and users carry the boundary and denies anyone but the platform roles from deleting or editing it.
6. Before production, run `simulate-principal-policy` against a representative role of each kind (a pipeline role, an EKS node role, a Lambda execution role) in a non-production account and confirm the service actions stay allowed and the escalation actions are denied.

Key code: Publish the boundary, attach it, and prove the result (AWS CLI). Appendix SF-SEC-018: the other 3 blocks, evidence, SOC 2, rollback.

```bash
aws cloudformation create-stack-set --stack-set-name sf-workload-boundary --template-body
file://sf-workload-boundary.yaml \
  --permission-model SERVICE_MANAGED --auto-deployment
Enabled=true,RetainStacksOnAccountRemoval=true \
  --capabilities CAPABILITY_NAMED_IAM
aws cloudformation create-stack-instances --stack-set-name sf-workload-boundary \
  --deployment-targets OrganizationalUnitIds=<root-organizational-unit-id> --regions us-east-1
# Service-managed stack sets skip the management account: deploy the same template there as a
stack.
aws cloudformation create-stack --profile <management-profile> --stack-name sf-workload-
boundary \
  --template-body file://sf-workload-boundary.yaml --capabilities CAPABILITY_NAMED_IAM

# IAM roles and users:
aws iam put-role-permissions-boundary --role-name <flagged-role-name> \
  --permissions-boundary arn:aws:iam::<account-id>:policy/sf-workload-boundary

# Identity Center permission sets (FleetDeveloper, FleetOnCall, DataAnalyst); their
```

```
AWSReservedSSO_ roles reject the call above.
INSTANCE_ARN=$(aws sso-admin list-instances --query 'Instances[0].InstanceArn' --output text)
for ps in <fleet-developer-permission-set-arn> <fleet-on-call-permission-set-arn> <data-
analyst-permission-set-arn>; do
  aws sso-admin put-permissions-boundary-to-permission-set --instance-arn "$INSTANCE_ARN" --
permission-set-arn "$ps" \
    --permissions-boundary CustomerManagedPolicyReference={Name=sf-workload-boundary,Path=/}
  aws sso-admin provision-permission-set --instance-arn "$INSTANCE_ARN" --permission-set-arn
"$ps" --target-type ALL_PROVISIONED_ACCOUNTS
done

# Non-production proof, one role of each kind: escalation denied, service actions allowed.
aws iam simulate-principal-policy --policy-source-arn arn:aws:iam::<non-production-account-
id>:role/<flagged-role-name> \
  --action-names iam:CreatePolicyVersion iam:SetDefaultPolicyVersion iam:PutUserPolicy
iam:PutRolePolicy \
                 iam:AttachRolePolicy iam:UpdateAssumeRolePolicy iam:PassRole
lambda:CreateFunction ec2:RunInstances \
  --query 'EvaluationResults[].[EvalActionName,EvalDecision]' --output table
aws iam simulate-principal-policy --policy-source-arn arn:aws:iam::<non-production-account-
id>:role/<flagged-role-name> \
  --action-names ssm:SendCommand iot:Publish sagemaker:CreateTrainingJob kinesis:PutRecord
ec2:AssignPrivateIpAddresses \
                 eks-auth:AssumeRoleForPodIdentity \
  --query 'EvaluationResults[].[EvalActionName,EvalDecision]' --output table
```

How it is verified

- `simulate-principal-policy` returns `implicitDeny` or `explicitDeny` for every escalation action on each flagged role, and `allowed` for the service actions in the second simulation.
- `aws iam create-role` without the boundary is denied by `sf-require-workload-boundary`, and with the boundary it succeeds for the platform workflow.
- `aws iam delete-role-permissions-boundary` and `aws iam create-policy-version` on `sf-workload-boundary` are denied for an ordinary administrator and succeed for the stack set's execution role.
- The next scan reports no role with PassRole plus lambda:CreateFunction or ec2:RunInstances on a wildcard resource.

SF-SEC-019: Kestrel's former management account still administers four Kestrel member accounts

Rating: High. Source: Vantalect baseline: cross-account admin trust. How it was rated: benchmark High (3) +0 exploitability (moderate) +0 blast radius (production) = 3

Where: `kestrel-prod` (…9031), `kestrel-staging` (…9054), `kestrel-dev` (…9076), `kestrel-data` (…9098) in us-east-1; 4 IAM roles. **Plan:** Jonah Reyes; `platform-landing-zone`; 6 hours; Week 4, starting Aug 3; needs SF-SEC-008 first. **Controls:** no benchmark control covers this.

Fix. Four Kestrel member accounts still let Kestrel's former management account assume `OrganizationAccountAccessRole`. That account is now a StrataForge member running CI, so replace the trust with the real management account.

Steps

1. List who in `kestrel-legacy-mgmt` still assumes the role, from CloudTrail in each Kestrel account, and move that CI to OIDC roles.
2. Open each Kestrel account through an IAM Identity Center permission set, not through the role you are about to change.
3. Replace the role's trust policy so only the management account root can assume it, and require MFA on the session.
4. Deploy the EventBridge rule so any assumption from another account pages the security team.
5. Enroll the Kestrel accounts in the landing zone (see control-tower-enrollment) so the role is managed from then on.

Key code: OrganizationAccountAccessRole trusting the StrataForge management account (IAM trust policy). Appendix SF-SEC-019: the other 2 blocks, evidence, SOC 2, rollback.

```JSON
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::<management-account-id>:root"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "Bool": {
          "aws:MultiFactorAuthPresent": "true"
        }
      }
    }
  ]
}
```

How it is verified

- The four roles show `arn:aws:iam::<management-account-id>:root` and no `<kestrel-legacy-mgmt-account-id>` in their trust policies.
- An assume-role attempt from `kestrel-legacy-mgmt` returns AccessDenied.
- The Kestrel CI jobs that ran under the legacy trust now assume OIDC roles and still pass.

SF-SEC-020: A former contractor's AWS account holds AdministratorAccess in mfg-mes-prod with no ExternalId

Rating: High. Source: IAM Access Analyzer external access. How it was rated: benchmark High (3) +0 exploitability (moderate) +0 blast radius (production) = 3

Where: `mfg-mes-prod` (…8084) in us-east-1; 1 IAM role. **Plan:** Mei Chen; `enterprise-infra`; 4 hours; Week 4, starting Aug 3. **Controls:** none; found by IAM Access Analyzer external access.

Fix. `kettleworth-mes-integration` in mfg-mes-prod gives a contractor account AdministratorAccess with no ExternalId, and the contract ended in 2024. Delete the role, and make every future vendor role narrow, short-lived, and bound to an ExternalId.

Steps

1. Check the role's last-used date and its CloudTrail activity for the last 90 days; record any real use and who owns it.
2. Detach AdministratorAccess and any inline policies, remove any instance profiles, and delete the role.
3. Ask Legal to confirm the contract end and tell the contractor that access is closed.
4. For any future vendor, create the role in CDK from the pattern below: a named vendor principal, an ExternalId, a one-hour session, and a policy listing only the actions the vendor needs.
5. Add each vendor role to the quarterly access review, with an owner and an expiry date in its tags.

Key code: Prove the role is idle, then remove it (AWS CLI). Appendix SF-SEC-020: the other 2 blocks, evidence, SOC 2, rollback.

```bash
set -euo pipefail
ROLE=kettleworth-mes-integration
aws iam get-role --role-name "$ROLE" --query 'Role.
[RoleLastUsed.LastUsedDate,RoleLastUsed.Region]'
# Save everything first; with set -e nothing is detached or deleted if a save fails.
mkdir -p retired-kettleworth
aws iam get-role --role-name "$ROLE" --output json > retired-kettleworth/role.json
aws iam list-attached-role-policies --role-name "$ROLE" --output json > retired-kettleworth/
attached.json
for name in $(aws iam list-role-policies --role-name "$ROLE" --query 'PolicyNames[]' --output
text); do
  aws iam get-role-policy --role-name "$ROLE" --policy-name "$name" --output json > "retired-
kettleworth/inline-$name.json"
done
test -s retired-kettleworth/role.json
for arn in $(aws iam list-attached-role-policies --role-name "$ROLE" --query
'AttachedPolicies[].PolicyArn' --output text); do
  aws iam detach-role-policy --role-name "$ROLE" --policy-arn "$arn"
done
for name in $(aws iam list-role-policies --role-name "$ROLE" --query 'PolicyNames[]' --output
text); do
  aws iam delete-role-policy --role-name "$ROLE" --policy-name "$name"
done
aws iam delete-role --role-name "$ROLE"
```

How it is verified

- `aws iam get-role --role-name kettleworth-mes-integration` returns NoSuchEntity in mfg-mes-prod.
- IAM Access Analyzer shows no external-access finding for a principal outside the organization in mfg-mes-prod.
- Each vendor role has an ExternalId condition, a `MaxSessionDuration` of 3600, and no AdministratorAccess.

SF-SEC-021: The staging OTA test key lets the production pipeline sign, and pilot robots still trust it

Rating: High. Source: Vantalect baseline: a signing key shared across environments. How it was rated: benchmark High (3) +0 exploitability (moderate) +0 blast radius (production) = 3

Where: `fleet-ota-prod` (…5117), `fleet-ota-staging` (…6148) in us-east-1; 1 KMS key. **Plan:** Sam Okafor; `fleet-infra`; 8 hours; Week 3, starting Jul 27. **Controls:** none; found by Vantalect OTA chain-of-trust review.

Fix. The key policy on `alias/ota-test-signing` lets the production role `ota-release-pipeline` sign with a key that staging treats as disposable, and the 1,240 Sentinel pilot robots on firmware 4.1 still keep its public key in bootloader slot 2, so they accept test-signed images. Scope the key to the staging pipeline role, then remove the test key from the pilot robots with a job signed by the production key.

Steps

1. Read the CloudTrail history for `Sign` on the test key and list every principal that used it, so nothing legitimate breaks.
2. Look up the key ID with `describe-key` (the key-policy calls take a key ID or key ARN, never an alias) and save the current key policy to the change record.
3. Write the new key policy: account administration, Sign for the staging pipeline role only, DescribeKey and GetPublicKey in their own statement (those two calls carry no signing-algorithm condition), and a deny of Sign for every principal except the staging pipeline role.
4. Apply it with `put-key-policy` and run one staging pipeline build to prove signing still works.
5. Create the EventBridge rule in the key's Region with state `ENABLED_WITH_ALL_CLOUDTRAIL_MANAGEMENT_EVENTS` (Sign is a read-only management event) so any later Sign by another principal pages the platform team.
6. Close the pilot-robot gap: have the firmware team write a job document that clears the test public key from bootloader slot 2, sign the job with the production key, put the 1,240 Sentinel pilot units in the thing group `sf-sentinel-pilot`, and roll the job out in waves with an abort rule. Production robots must already trust only the production public key.
7. Confirm from the job's execution summary and the pilots' next reported key manifest that no unit still lists the test key, then schedule the test key's deletion only after that.

Key code: Review who has used the key, save and apply the policy, and read it back (AWS CLI). Appendix SF-SEC-021: the other 3 blocks, evidence, SOC 2, rollback.

```bash
set -euo pipefail
aws cloudtrail lookup-events --profile <fleet-ota-staging-profile> --region us-east-1 \
  --lookup-attributes AttributeKey=EventName,AttributeValue=Sign --max-results 50 \
  --query 'Events[].CloudTrailEvent' --output text | grep -o '"arn":"[^"]*"' | sort | uniq -c
KEY_ID=$(aws kms describe-key --profile <fleet-ota-staging-profile> --region us-east-1 --key-id
alias/ota-test-signing \
  --query KeyMetadata.KeyId --output text)
# Save the current policy first; the rollback restores exactly this file.
aws kms get-key-policy --profile <fleet-ota-staging-profile> --region us-east-1 --key-id
"$KEY_ID" --policy-name default \
  --query Policy --output text > ota-test-key-policy.before.json
test -s ota-test-key-policy.before.json
aws kms put-key-policy --profile <fleet-ota-staging-profile> --region us-east-1 \
  --key-id "$KEY_ID" --policy-name default --policy file://ota-test-key-policy.json
aws kms get-key-policy --profile <fleet-ota-staging-profile> --region us-east-1 \
  --key-id "$KEY_ID" --policy-name default --query Policy --output text
# Sign is a read-only management event: only this rule state delivers it. The SNS topic must
allow events.amazonaws.com to publish.
aws events put-rule --profile <fleet-ota-staging-profile> --region us-east-1 --name sf-ota-
test-unexpected-sign \
  --state ENABLED_WITH_ALL_CLOUDTRAIL_MANAGEMENT_EVENTS --event-pattern file://ota-test-
unexpected-sign-pattern.json
aws events put-targets --profile <fleet-ota-staging-profile> --region us-east-1 --rule sf-ota-
test-unexpected-sign \
  --targets Id=page-platform,Arn=<platform-alarm-topic-arn>
```

How it is verified

- `aws kms get-key-policy` shows no principal from fleet-ota-prod.
- A `kms:Sign` call with the production `ota-release-pipeline` role against the test key returns AccessDeniedException, and the EventBridge rule's `State` is `ENABLED_WITH_ALL_CLOUDTRAIL_MANAGEMENT_EVENTS`.
- The next staging pipeline run signs successfully as `ota-staging-pipeline`.
- The job `sentinel-clear-test-key-slot-2` shows every one of the 1,240 pilot units SUCCEEDED, and an image signed with the test key is refused by a pilot unit on the bench.

SF-SEC-022: AWS Config records nothing in 570 account-Regions

Rating: High. Source: Security Hub CSPM Config.1. How it was rated: benchmark Critical (4) -1 exploitability (limited) +1 blast radius (organization) = 4; rule 2: Critical needs a remote path and a production-or-wider blast radius; capped at High

Where: All 48 accounts in 17 Regions; 570 account-Regions. **Plan:** Elena Vasquez; `platform-landing-zone`; 10 hours; Week 5, starting Aug 10; needs SF-SEC-039 first. **Controls:** Config.1.

Fix. AWS Config records in only six Regions and not in the Kestrel accounts (Config.1). Extend the Control Tower landing zone's governed Regions to all 17, enroll the accounts that Control Tower does not manage yet, and aggregate everything in sf-security-tooling.

Steps

1. Register sf-security-tooling as the Config delegated administrator.
2. Update the landing zone manifest so `governedRegions` lists all 17 Regions; Control Tower then deploys its Config recorder and rules to them.
3. Enroll the unenrolled accounts (see control-tower-enrollment) so Control Tower records them too.
4. Create the organization configuration aggregator for all Regions in sf-security-tooling.
5. Remember that Control Tower's recorder uses its own role, which is why Config.1 is set to skip the service-linked role check in Security Hub CSPM (see securityhub-org).

Key code: Delegated administrator and the landing zone Region change (management account) (AWS CLI). Appendix SF-SEC-022: the other 1 block, evidence, SOC 2, rollback.

```bash
aws organizations enable-aws-service-access --service-principal config.amazonaws.com
aws organizations register-delegated-administrator --service-principal config.amazonaws.com --
account-id <security-tooling-account-id>

LZ=$(aws controltower list-landing-zones --query 'landingZones[0].arn' --output text)
aws controltower get-landing-zone --landing-zone-identifier "$LZ" --query
'landingZone.manifest' > lz-manifest.json
# Edit lz-manifest.json so governedRegions lists these 17 Regions:
#   us-east-1 us-east-2 us-west-1 us-west-2 ca-central-1 sa-east-1 eu-central-1 eu-west-1 eu-
west-2
#   eu-west-3 eu-north-1 ap-northeast-1 ap-northeast-2 ap-northeast-3 ap-southeast-1 ap-
southeast-2 ap-south-1
aws controltower update-landing-zone --landing-zone-identifier "$LZ" --version <landing-zone-
version> --manifest file://lz-manifest.json
aws controltower get-landing-zone-operation --operation-identifier <operation-identifier> --
query 'operationDetails.status'
```

How it is verified

- `aws configservice describe-configuration-recorder-status` shows a recording recorder in all 17 Regions of a sample account, including a Kestrel account (Security Hub CSPM Config.1).
- `aws configservice describe-configuration-aggregator-sources-status` shows every account and Region as succeeded.
- `aws controltower get-landing-zone` shows `governedRegions` with all 17 Regions and status ACTIVE.

SF-SEC-023: GuardDuty is off in 558 account-Regions, including every Kestrel account; most protection plans are off

Rating: High. Source: Security Hub CSPM GuardDuty.1 and GuardDuty.5–13. How it was rated: benchmark High (3) -1 exploitability (limited) +1 blast radius (organization) = 3

Where: All 48 accounts in 17 Regions; 558 account-Regions. **Plan:** Nadia Petrova; `platform-landing-zone`; 12 hours; Week 5, starting Aug 10. **Controls:** GuardDuty.1, GuardDuty.5, GuardDuty.6, GuardDuty.7, GuardDuty.8, GuardDuty.9, GuardDuty.11, GuardDuty.12, GuardDuty.13.

Fix. GuardDuty runs in only six Regions, not in the Kestrel accounts, and with partial protection plans (GuardDuty.1, GuardDuty.5 to GuardDuty.13). Make sf-security-tooling the delegated administrator in all 17 Regions and auto-enable it, with every protection plan, for all members.

Steps

1. From the management account, enable trusted access and make sf-security-tooling the GuardDuty delegated administrator in each of the 17 Regions.
2. In sf-security-tooling, set the organization configuration to auto-enable `ALL` members and every protection plan the API exposes: S3 Protection, EKS Protection, Runtime Monitoring, Malware Protection for EC2, RDS Protection, and Lambda Protection.
3. Turn on Malware Protection for S3 on the sensitive buckets, Malware Protection for AWS Backup on the backup vaults, and AI Protection where Bedrock or SageMaker AI is used; these are set per resource or plan.
4. Confirm the five Kestrel accounts show as members in every Region.
5. Route severity 7 and above findings to the security on-call from every Region (the routing script forwards each Region to us-east-1), and check that Extended Threat Detection findings arrive, since it is on by default.

Key code: Where GuardDuty stands today, per Region (read-only, sf-security-tooling) (AWS CLI). Appendix SF-SEC-023: the other 3 blocks, evidence, SOC 2, rollback.

```bash
for region in us-east-1 us-east-2 us-west-1 us-west-2 ca-central-1 sa-east-1 eu-central-1 eu-west-1 eu-west-2 eu-west-3 eu-north-1 ap-northeast-1 ap-northeast-2 ap-northeast-3 ap-southeast-1 ap-southeast-2 ap-south-1; do
  echo "$region detector: $(aws guardduty list-detectors --profile <security-tooling-profile> --region $region --query 'DetectorIds[0]' --output text)"
done
```

How it is verified

- `aws guardduty list-detectors` returns a detector in all 17 Regions, `list-organization-admin-accounts` names sf-security-tooling in each, and `list-members` shows every member account as Enabled, which is every active account except the administrator itself (Security Hub CSPM GuardDuty.1).
- `aws guardduty describe-organization-configuration` shows `AutoEnableOrganizationMembers` ALL and every listed feature on ALL.
- A sample finding generated with `create-sample-findings` in a Region other than us-east-1 reaches the on-call route.

SF-SEC-024: S3 Block Public Access is off at the account level in 31 accounts and on 47 buckets

Rating: High. Source: Security Hub CSPM S3.1 and S3.8. How it was rated: benchmark High (3) -1 exploitability (limited) +1 blast radius (organization) = 3

Where: 37 accounts in 4 Regions; 31 accounts. **Plan:** Elena Vasquez; `platform-landing-zone`; 10 hours; Week 3, starting Jul 27. **Controls:** S3.1, S3.8.

Fix. Account-level Block Public Access is off in many accounts, so any bucket can be made public by a policy or ACL (S3.1, S3.8). Set it with `s3control put-public-access-block` in every account, report drift with a Config rule, and deny anyone but the platform role from changing it.

Steps

1. List every active account and run the s3control command in each one; account-level settings apply to all buckets, current and future. Then set each bucket's own block, which is what S3.8 reads, except the buckets that serve public content on purpose.
2. Check for buckets that serve public content on purpose (static websites) before you run it; move them behind CloudFront with origin access control.
3. Deploy the Config rule as a stack set so a later change shows up as non-compliant.
4. Attach `sf-protect-account-defaults` (see scp-guardrails); its KeepS3AccountBlock statement lets only the platform roles change or delete the account setting.
5. Consider the Organizations Amazon S3 policy type as a later step: it sets all four settings across accounts and overrides the account-level values, and detaching it reverts accounts to their own settings.

Key code: Set the account-level block and every bucket's own block in every account (AWS CLI). Appendix SF-SEC-024: the other 2 blocks, evidence, SOC 2, rollback.

```bash
# Active accounts come from the State field; the older Status field is deprecated.
# intentionally-public-buckets.txt lists, one name per line, the buckets that serve public
content on purpose.
BLOCK=BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets=t
rue
for account in $(aws organizations list-accounts --query 'Accounts[?State==`ACTIVE`].Id' --
output text); do
  creds=$(aws sts assume-role --role-arn arn:aws:iam::$account:role/
OrganizationAccountAccessRole \
    --role-session-name s3-bpa --query Credentials --output json) || { echo "SKIP $account:
cannot assume the role"; continue; }
  (export AWS_ACCESS_KEY_ID=$(echo "$creds" | jq -r .AccessKeyId) AWS_SECRET_ACCESS_KEY=$(echo
"$creds" | jq -r .SecretAccessKey)
   export AWS_SESSION_TOKEN=$(echo "$creds" | jq -r .SessionToken)
   # Account level (S3.1).
   aws s3control put-public-access-block --account-id "$account" --public-access-block-
configuration $BLOCK
   aws s3control get-public-access-block --account-id "$account" --query
PublicAccessBlockConfiguration
   # Bucket level (S3.8 evaluates each bucket's own setting).
   for bucket in $(aws s3api list-buckets --query 'Buckets[].Name' --output text); do
     grep -qxF "$bucket" intentionally-public-buckets.txt 2>/dev/null && { echo "KEEP $bucket";
continue; }
     aws s3api put-public-access-block --bucket "$bucket" --public-access-block-configuration
$BLOCK \
       || echo "FAILED $account $bucket"
   done)
done
```

How it is verified

- `aws s3control get-public-access-block --account-id <account-id>` shows four true values in every account, and `aws s3api get-public-access-block --bucket <bucket-name>` does for every bucket except the intentionally public ones.
- Security Hub CSPM S3.1 passes in every account and S3.8 passes for every bucket after the next evaluation.
- `aws s3control delete-public-access-block` from an ordinary administrator role is denied by `sf-protect-account-defaults`.

SF-SEC-025: The 2019 organization trail can be stopped from the management account, and its logs can be deleted

Rating: High. Source: Vantalect baseline (log archive without Object Lock); Security Hub CSPM CloudTrail.2, .4, .7. How it was rated: benchmark High (3) -1 exploitability (limited) +1 blast radius (organization) = 3

Where: `sf-management` (…0007), `kestrel-prod` (…9031), `kestrel-data` (…9098), `kestrel-legacy-mgmt` (…9120) in us-east-1; 4 trails. **Plan:** Nadia Petrova; `platform-landing-zone`; 20 hours; Week 5, starting Aug 10. **Controls:** CloudTrail.2, CloudTrail.4, CloudTrail.7.

Fix. `sf-org-trail` has no log file validation (CloudTrail.4) or KMS key (CloudTrail.2), and its bucket in sf-log-archive has no Object Lock and can be written and deleted from the management account. Build a new bucket in sf-log-archive with Object Lock in COMPLIANCE mode for 400 days, then a new organization trail `sf-org-trail-v2` that logs management events and S3 data events for every bucket. Protect the key and the account, and alarm in the management account on any attempt to stop the trail, schedule the key for deletion, or close sf-log-archive.

Steps

1. Deploy the archive stack in sf-log-archive: a KMS key that is retained if the stack is deleted, an access-log bucket, and the Object Lock bucket with a bucket policy that allows only CloudTrail to write, plus the TLS-only Deny.
2. Deploy the trail stack in the management account: a multi-Region organization trail with log file validation and the KMS key, delivering to the new bucket.
3. Run both trails side by side for a week and compare event counts; do not stop the old trail until the new one is complete.
4. Deploy the EventBridge rule in the management account in us-east-1, where these API calls are recorded; it pages on StopLogging, DeleteTrail, UpdateTrail, PutEventSelectors, ScheduleKeyDeletion, DisableKey, and CloseAccount, and exempts no one, so a planned pipeline change pages too and the on-call confirms it.
5. Attach the `sf-protect-log-archive` SCP to the sf-log-archive account, and attach the IAM deny for `organizations:CloseAccount` to every administrator role in the management account. COMPLIANCE-mode objects are deleted if the account is closed, and an SCP cannot stop the management account from closing it, so the deny and the alarm are the controls on that path.
6. Stop the old trail, keep its bucket read-only for the retention period, and tighten its bucket policy to Allow statements for the security team only.
7. COMPLIANCE retention cannot be shortened by anyone, root included, so confirm the 400 days with Legal before the first object is written.

Key code: Management account: the new organization trail (CloudFormation). Appendix SF-SEC-025: the other 6 blocks, evidence, SOC 2, rollback.

```yaml
AWSTemplateFormatVersion: '2010-09-09'
Description: Multi-Region organization trail with log file validation, delivered to the
immutable archive.
Parameters:
  TrailKeyArn: {Type: String, Description: KMS key ARN from the sf-log-archive stack.}
Resources:
  OrgTrail:
    Type: AWS::CloudTrail::Trail
    Properties:
      TrailName: sf-org-trail-v2
      S3BucketName: sf-log-archive-org-trail-locked-us-east-1
      KMSKeyId: !Ref TrailKeyArn
      IsLogging: true
      IsMultiRegionTrail: true
      IsOrganizationTrail: true
      IncludeGlobalServiceEvents: true
      EnableLogFileValidation: true
      EventSelectors:
        - ReadWriteType: All
          IncludeManagementEvents: true
          DataResources:
            - {Type: 'AWS::S3::Object', Values: ['arn:aws:s3']}
            - {Type: 'AWS::Lambda::Function', Values: ['arn:aws:lambda']}
```

How it is verified

- `aws cloudtrail describe-trails` shows `LogFileValidationEnabled` true and a `KmsKeyId` for `sf-org-trail-v2` (Security Hub CSPM CloudTrail.4, CloudTrail.2).
- `aws s3api get-object-lock-configuration` on the new bucket shows COMPLIANCE for 400 days, and `aws s3api delete-object --bucket sf-log-archive-org-trail-locked-us-east-1 --key <log-object-key> --version-id <version-id>` run as an administrator of sf-log-archive fails with AccessDenied because of the retention.
- `aws cloudtrail get-event-selectors --trail-name sf-org-trail-v2` shows S3 data events for all buckets, and `aws kms describe-key` on the trail key shows KeyState Enabled with no deletion date.
- A test `StopLogging` on a scratch trail in the management account pages the security team, and `aws cloudtrail validate-logs` passes for the last 24 hours.

SF-SEC-026: Service control policies stop nothing: only FullAWSAccess and DenyLeaveOrganization are attached

Rating: High. Source: Vantalect baseline: organization guardrails. How it was rated: benchmark High (3) -1 exploitability (limited) +1 blast radius (organization) = 3

Where: All 48 accounts in 17 Regions; 2 policies. **Plan:** Elena Vasquez; `platform-landing-zone`; 24 hours; Week 3, starting Jul 27. **Controls:** no benchmark control covers this.

Fix. The organization has only FullAWSAccess and a deny-leave-organization SCP, and no resource control policies or declarative policies. Add four SCPs: one protects the security services and the trail, one denies member-account root, one denies the 11 unused Regions, and one keeps the account defaults on (EBS encryption; S3, snapshot, and AMI Block Public Access; RDS snapshot sharing; KMS key rotation) and stops new IAM users, access keys, and console passwords. Then enable and attach the RCP and declarative policy types.

Steps

1. Create the four SCPs below; each is well under the 10,240-character limit, and an entity can hold 10 SCPs.
2. Attach each to the Sandbox OU first, run a day of normal work and the platform pipelines, then attach to the workload OUs and finally the root.
3. Apply the member-root deny only after the root credential cleanup finishes (see root-credentials). Its `Null` condition on `aws:AssumedRoot` exempts the short root sessions that `sts:AssumeRoot` starts from the management account; without it the deny also blocks the audit, credential-deletion, and bucket-unlock tasks that centralized root access exists for. Before attaching it to the root, start an `IAMAuditRootUserCredentials` session in a sandbox account that already has the policy and confirm it works.
4. Enable the resource control policy type and attach the S3 TLS RCP (see s3-tls-rcp); enable the declarative policy type and attach the EC2 policy (see imdsv2 and ebs-snapshot-public). RCPs do not affect the management account.
5. Keep the exception roles (PlatformAdmin, AWSControlTowerExecution, stack set execution roles) in the policies so pipelines keep working, and review the exceptions every quarter.

Key code: sf-deny-member-root: deny every action by a member account's root user, except root sessions started with sts:AssumeRoot (Service control policy). Appendix SF-SEC-026: the other 4 blocks, evidence, SOC 2, rollback.

```JSON
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyMemberRoot",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:PrincipalArn": "arn:aws:iam::*:root"
        },
        "Null": {
          "aws:AssumedRoot": "true"
        }
      }
    }
  ]
}
```

How it is verified

- `aws organizations list-policies-for-target --target-id <org-root-id> --filter SERVICE_CONTROL_POLICY` lists the four new SCPs, and the RCP and the two EC2 declarative policies are attached to the root.
- `cloudtrail:StopLogging` from an administrator who is not the security role is denied, and an EC2 call in `us-east-2` is denied.
- With `sf-deny-member-root` attached, `aws sts assume-root --target-principal <sandbox-account-id> --task-policy-arn arn=arn:aws:iam::aws:policy/root-task/IAMAuditRootUserCredentials` succeeds and its session can list the root user's credentials, while a call made with the account's real root user credentials is denied.
- The platform pipelines and the Control Tower baseline run unchanged for a week after the root attachment.

SF-SEC-027: SSH bastions accept connections from the whole internet

Rating: High. Source: Security Hub CSPM EC2.19, EC2.53, EC2.54, and EC2.18. How it was rated: benchmark Critical (4) -1 exploitability (limited) +1 blast radius (organization) = 4; rule 2: Critical needs a remote path and a production-or-wider blast radius; capped at High

Where: `sf-shared-services` (…1188) in us-east-1; 1 security group. **Plan:** Elena Vasquez; `platform-landing-zone`; 6 hours; Week 3, starting Jul 27. **Controls:** EC2.18, EC2.19, EC2.53.

Fix. This is the same change as SF-SEC-009, applied to the accounts listed above. Its steps, code, verification, and rollback are the same, and the appendix lists them under SF-SEC-009.

SF-SEC-028: SSH is open to the internet on 76 development instances

Rating: High. Source: Security Hub CSPM EC2.19, EC2.53, EC2.54, and EC2.18. How it was rated: benchmark Critical (4) +0 exploitability (moderate) -1 blast radius (account) = 3

Where: `fleet-dev` (…6037) in us-east-1; 7 security groups. **Plan:** Sam Okafor; `fleet-infra`; 18 hours; Week 3, starting Jul 27. **Controls:** EC2.18, EC2.19, EC2.53, EC2.54.

Fix. This is the same change as SF-SEC-009, applied to the accounts listed above. Its steps, code, verification, and rollback are the same, and the appendix lists them under SF-SEC-009.

SF-SEC-029: 4 more member-account root users have no MFA; 40 use virtual MFA

Rating: High. Source: Security Hub CSPM IAM.9 and IAM.6. How it was rated: benchmark Critical (4) -1 exploitability (limited) +0 blast radius (production) = 3

Where: 44 accounts in us-east-1; 44 root users. **Plan:** Grace Liu; `platform-landing-zone`; 6 hours; Week 4, starting Aug 3. **Controls:** IAM.6, IAM.9.

Fix. This is the same change as SF-SEC-010, applied to the accounts listed above. Its steps, code, verification, and rollback are the same, and the appendix lists them under SF-SEC-010.

SF-SEC-030: 23 robot certificates are shared by 1,840 robots at 3 customer sites

Rating: High. Source: AWS IoT Device Defender DEVICE_CERTIFICATE_SHARED_CHECK. How it was rated: benchmark Critical (4) -1 exploitability (limited) +0 blast radius (production) = 3

Where: `fleet-ingest-prod` (…5012) in 2 Regions; 23 certificates. **Plan:** Sam Okafor; `fleet-infra`; 32 hours; Week 3, starting Jul 27; the rest (Re-provision the robots at the Bremen and Ota sites during their maintenance windows) is due September 30, 2026. **Controls:** none; found by DEVICE_CERTIFICATE_SHARED_CHECK; CONFLICTING_CLIENT_IDS_CHECK.

Fix. Some certificates are shared by many robots (DEVICE_CERTIFICATE_SHARED_CHECK and CONFLICTING_CLIENT_IDS_CHECK), so one robot's connection kicks another off and revoking a stolen certificate would take all of them offline. Give each robot its own certificate through fleet provisioning by claim, with a pre-provisioning hook that lets only listed serial numbers through, and revoke the shared ones.

Steps

1. List the shared certificates and the robots on each from the Device Defender findings, and save the robots' serial numbers.

2. Deploy the pre-provisioning hook below and load the saved serial numbers into its allow list, so a stolen claim certificate cannot register a robot that is not on the list.
3. Create the claim certificate, attach a policy that allows only the provisioning topics, and create the provisioning template with the hook; the template's override settings let it adopt a robot that already exists as a thing.
4. Ship a firmware job (it reaches these robots through the jobs topics that `sf-robot-legacy-v1-restricted` allows) that generates a key on the robot, requests its own certificate with a CSR, and stores it.
5. Roll the job out by thing group; confirm each robot reconnects with its own certificate, then detach the shared certificate from that robot's thing.
6. Revoke each shared certificate only when no thing is attached to it, then rotate the claim certificate.

Key code: List shared certificates and the robots on them (AWS CLI). Appendix SF-SEC-030: the other 3 blocks, evidence, SOC 2, rollback.

```bash
TASK=$(aws iot start-on-demand-audit-task --target-check-names DEVICE_CERTIFICATE_SHARED_CHECK
CONFLICTING_CLIENT_IDS_CHECK --query taskId --output text)
# Wait until `aws iot describe-audit-task --task-id "$TASK" --query taskStatus` returns
COMPLETED.
aws iot list-audit-findings --task-id "$TASK" --check-name DEVICE_CERTIFICATE_SHARED_CHECK \
  --query 'findings[].nonCompliantResource.resourceIdentifier.deviceCertificateId' --output
text | tr '\t' '\n' | sort -u > shared-certificate-ids.txt
: > shared-robots.txt
for id in $(cat shared-certificate-ids.txt); do
  arn=$(aws iot describe-certificate --certificate-id "$id" --query
certificateDescription.certificateArn --output text)
  echo "$id $(aws iot list-principal-things --principal "$arn" --query 'things | length(@)')"
  aws iot list-principal-things --principal "$arn" --query 'things[]' --output text | tr '\t'
'\n' >> shared-robots.txt
done
```

How it is verified

- A new on-demand audit shows no DEVICE_CERTIFICATE_SHARED_CHECK or CONFLICTING_CLIENT_IDS_CHECK finding.
- `aws iot list-principal-things` returns at most one thing for every active certificate.
- Every shared certificate shows status REVOKED, and no robot has connected with one since.
- A provisioning request from the claim certificate with a serial number that is not in `sf-robot-serial-allowlist` is refused by the hook.

Medium and Low findings

The register has 15 Medium and 9 Low findings. Their evidence, steps, code, verification, and rollback are in the appendix under the same finding ID.

Medium and Low findings: scope and controls. Severity, owner, effort, and plan are in the register above, and the repository and SOC 2 criteria are in the appendix.

ID	Finding	Where	Controls
SF-SEC-031	1,247 Lambda functions run on deprecated runtimes	5 accounts, 1,247 Lambda functions	Lambda.2
SF-SEC-032	EBS encryption by default is off in 762 account-Regions; 1,245 volumes are unencrypted	48 accounts, 1,245 EBS volumes	EC2.7
SF-SEC-033	556 running instances are not managed by Systems Manager; 2,380 run AMIs over a year old	37 accounts, 556 instances	SSM.1
SF-SEC-034	39 internet-facing ALBs have no AWS WAF; 84 ALBs do not redirect HTTP to HTTPS	8 accounts, 123 load balancers	ELB.1
SF-SEC-035	22 database instances store data unencrypted	11 accounts, 22 DB instances	RDS.3
SF-SEC-036	9 ElastiCache replication groups lack encryption at rest or in transit	3 accounts, 9 clusters and domains	
SF-SEC-037	5 EFS file systems are not encrypted at rest	5 accounts, 5 file systems	EFS.1, EFS.8
SF-SEC-038	Security Hub CSPM runs in only 3 Regions, with FSBP only and no CIS standard	48 accounts, 687 account-Regions	
SF-SEC-039	7 accounts are outside Control Tower, and the landing zone has drifted	7 accounts, 7 accounts	
SF-SEC-040	Break-glass access is not alarmed and was last tested in April 2025	1 account, 3 emergency identities	
SF-SEC-041	SSH and RDP are open to the internet on 27 sandbox instances	2 accounts, 3 security groups	EC2.18, EC2.19, EC2.53, EC2.54

ID	Finding	Where	Controls
SF-SEC-042	Amazon Inspector is off almost everywhere; 132 ECR repositories have no image scanning	48 accounts, 3,380 failed checks	ECR.1, Inspector.1, Inspector.2, Inspector.3, Inspector.4
SF-SEC-043	750 default VPCs, most in Regions nobody uses, have open default security groups and network ACLs	48 accounts, 2,350 failed checks	EC2.2, EC2.6, EC2.21
SF-SEC-044	IAM Access Analyzer covers only us-east-1; 768 account-Regions have no external access analyzer	48 accounts, 768 account-Regions	IAM.28
SF-SEC-045	31 databases keep one day of backups; 112 resilience checks fail	15 accounts, 143 database settings	RDS.5, RDS.13
SF-SEC-046	130 customer managed policies use service-wide wildcards such as s3:*	14 accounts, 130 IAM policies	IAM.21
SF-SEC-047	2,850 of 14,464 resources have no Team tag, so no owner can be found for them	48 accounts, 2,850 resources	
SF-SEC-048	1,922 workload roles have not been used in 90 days or more	47 accounts, 1,922 IAM roles	
SF-SEC-049	MFA delete is off on every bucket	23 accounts, 317 S3 buckets	S3.20
SF-SEC-050	Account baseline gaps: no security contact in 42 accounts, no support role in 20, CloudShell full access on 195 identities	48 accounts, 275 failed checks	Account.1, IAM.15, IAM.16, IAM.18, IAM.26, IAM.27
SF-SEC-051	Server access logging is off on 233 buckets	23 accounts, 233 S3 buckets	S3.9
SF-SEC-052	220 buckets accept requests without TLS	22 accounts, 220 S3 buckets	S3.5
SF-SEC-053	78 customer managed keys do not rotate	36 accounts, 78 KMS keys	KMS.4
SF-SEC-054	EKS control-plane logging is off on 15 of 16 clusters, and every cluster still uses the aws-auth ConfigMap	10 accounts, 16 EKS clusters	

False positives

Six scanner results were wrong or intentional, and none was deleted. Together they account for 247 of the 19,141 failed checks in the week-one scan. Removing them moves the organization score from 32.3% to 32.4%. The larger point is the practice: each stays in the register with its reason and its owner's confirmation, so an auditor sees the decision and the next scan does not raise it again.

False positives FP-01 to FP-03, with the reason each is not a finding and the setting that stops the scanner reporting it.

ID	Source	What the scanner reported	Count	Why it is not a finding	Confirmed by	What changes
FP-01	Config.1	AWS Config recorders that Control Tower manages fail the service-linked-role test	240 Regions	Recording is on and complete in every governed account-Region; Control Tower's recorder uses its own role (aws-controltower-ConfigRecorderRole) by design, so only the role half of Config.1 fails. Verified in the AWS Config console and with describe-configuration-recorders in 12 sampled account-Regions.	Elena Vasquez	Set the Config.1 parameter includeConfigServiceLinkedRoleCheck to false in the Security Hub CSPM configuration policy (week five).
FP-02	EC2.18	TURN port 3478 on the WebRTC relays is open to the internet on purpose	3 security groups	The remote-assist relays must accept TURN (3478) and TLS (443) from robots and operators anywhere; the relays run no shell access and their role can only write session metrics. Confirmed with Sam Okafor on 07/21/2026.	Sam Okafor	Add 3478 to EC2.18's authorizedTcpPorts parameter in the Security Hub CSPM configuration policy.
FP-03	IAM.22	Break-glass users are unused by design	2 IAM users	breakglass-01 and breakglass-02 exist for an IAM Identity Center or Okta outage and must stay unused; their passwords and hardware MFA devices are sealed in the CISO's safe. Their use is covered by the break-glass finding instead.	Nadia Petrova	Suppress IAM.22 for the two users with the workflow note "break-glass, alarmed".

False positives FP-04 to FP-06, with the reason each is not a finding and the setting that stops the scanner reporting it.

ID	Source	What the scanner reported	Count	Why it is not a finding	Confirmed by	What changes
FP-04	KMS.4	Keys with imported key material cannot rotate automatically	2 KMS keys	The two SAP HANA backup keys use key material imported from StrataForge's HSM (origin EXTERNAL); AWS KMS does not rotate imported material, and the HSM key ceremony rotates it every 12 months (last on 02/03/2026).	Ana Kowalski	Suppress KMS.4 for keys with origin EXTERNAL; the annual key ceremony is the control.
FP-05	Vantalect check: secret-named Lambda environment variables	Variables that hold Secrets Manager ARNs, not secrets	3 Lambda functions	The scanner collects variable names only. The customer-portal team showed the CDK code: DATABASE_URL and PARTNER_API_KEY hold Secrets Manager ARNs that the function resolves at start-up.	Sam Okafor	Add the three functions to the scanner's allow list with the CDK reference.
FP-06	DEVICE_CERTIFICATE_SHARED_CHECK	Simulator rigs at the Pittsburgh test center share an identity on purpose	2 certificates	Each rig runs two hardware-in-the-loop simulators that present one test identity; neither is a robot, and the certificates are scoped to the test thing group.	Sam Okafor	Move the rigs' certificates to a thing group excluded from the scheduled audit's shared-certificate check.

Deliverable 3: identity and access review

Of the 1,468 people with AWS access, 439 do not use single sign-on; 1,009 of 1,139 access keys are older than 90 days; and 12 paths lead from ordinary access to administrator. This is the answer to "who and what can reach production", as of July 15, 2026. It covers human access, IAM users and keys, root and break-glass accounts, cross-account and OIDC trust, privilege escalation, unused roles, and administrator policies. Every item that is a finding links to its register entry.

People and Identity Center

Human access to AWS.

Measure	Value
People with AWS access	1,468
People using IAM Identity Center	1,029
People with IAM users only	439
Identity Center coverage	70.1%
Identity Center permission sets	23
IAM users with a console password	553
Console users without MFA	108
Accounts still using a legacy Okta SAML application	6

Identity Center covers 70.1% of the people who have AWS access. The rest sign in through IAM users, and 6 accounts still use a legacy Okta SAML application that grants administrator or power-user access and whose trust checks only the SAML audience. The sharper gap is 108 console users with no MFA at all, including users in the management account (SF-SEC-013).

IAM users and access keys

There are 1,394 IAM users, and only 544 are people. The rest are machines that were given a user because it was the quickest thing: 247 for CI, 488 for services, and 113 for vendors.

IAM users by what they are used for.

Kind of IAM user	Users
Human	544
CI	247
Service	488
Vendor	113
Break-glass	2
All IAM users	**1,394**

The 1,139 active access keys by age and by the kind of user that holds them. Keys over 90 days old break the 90-day rotation rule (IAM.3).

Age of the access key	People	CI	Services	Vendors	All
Under 90 days	28	26	62	14	130
90 days to 1 year	38	16	30	3	87
1-2 years	68	53	102	20	243
2-3 years	67	67	121	24	279
Over 3 years	59	93	193	55	400

Two of those numbers need action. **161 keys have never been used,** which is a credential nobody needs; deactivate each one, confirm nothing breaks over a release cycle, and then delete it. And the oldest keys belong to CI and service users: keys over three years old are held by 93 CI users and 193 service users, which is where a leaked key is least likely to be noticed. The fix is in SF-SEC-012; it replaces most of them with roles.

The accounts with the most old keys

The 12 accounts with the most access keys older than 90 days.

Account	Account ID	IAM users	People	Console, no MFA	Active keys	Keys over 90 days	Administrator users
`sf-cicd`	…1356	93	3	1	96	84	7
`fleet-dev`	…6037	100	64	5	76	68	1
`mfg-mes-prod`	…8084	64	0	0	68	60	0
`kestrel-prod`	…9031	92	52	11	65	54	24
`kestrel-legacy-mgmt`	…9120	70	24	6	58	53	21
`fleet-api-prod`	…5044	55	0	0	57	51	1
`kestrel-dev`	…9076	74	61	21	43	41	24
`corp-apps`	…8129	49	6	1	46	40	0
`crucible-research`	…7088	64	58	7	41	38	0
`suspended-legacy-ci`	…4099	44	3	3	44	38	20
`kestrel-data`	…9098	59	36	8	41	37	13
`fleet-ingest-prod`	…5012	38	0	0	39	34	1

Root users and break-glass access

Root user protection across the 48 accounts.

Root user status	Accounts
Hardware MFA	1
Virtual MFA	40
No MFA	7
With an access key	3

Accounts whose root user has no MFA. The three with access keys are the Critical finding.

Account	Account ID	OU	Root MFA	Access keys	Keys last used	Last root sign-in
`kestrel-prod`	…9031	Workloads/Kestrel	None	1	December 9, 2025	Never
`kestrel-staging`	…9054	Workloads/Kestrel	None	0	Never	Never
`kestrel-dev`	…9076	Workloads/Kestrel	None	0	Never	Never
`kestrel-data`	…9098	Workloads/Kestrel	None	1	Never	Never
`kestrel-legacy-mgmt`	…9120	Workloads/Kestrel	None	1	June 30, 2026	December 8, 2025
`sandbox-hackathon`	…4078	Sandbox	None	0	Never	December 5, 2024
`suspended-legacy-ci`	…4099	Suspended	None	0	Never	January 29, 2026

Root is the account's owner, and nobody should be using it. The 3 root users with access keys, all in Kestrel accounts, are SF-SEC-010 (Critical), and a Jenkins job used one of them last month. Another 4 member-account root users have no MFA and 40 use virtual MFA (SF-SEC-029). The management account's root has hardware MFA, and its last sign-in was February 11, 2026.

Break-glass access is two IAM users, `breakglass-01` and `breakglass-02`, in the management account. Their passwords and hardware MFA devices are sealed in the CISO's safe, which is the right design, but nothing alarms when one is used, and the access has not been tested recently (SF-SEC-040). AWS now lets the management account delete a member account's root credentials centrally, so the fix is to remove the root passwords and keys from every member account and keep one hardware-MFA root, in the management account only.

Cross-account and OIDC trust

We reviewed 17 trust relationships: who can assume a role in StrataForge's accounts, from where, on what condition, with what permissions.

Cross-account, external, and OIDC trust relationships, findings first.

Role	Accounts	Trusted principal	Conditions	Permissions	Assessment
`backup-vendor-restore`	1	backup vendor (external)	sts:ExternalId	AWS Backup restore operator	Expected
`observability-vendor-integration`	24	observability SaaS vendor (external)	sts:ExternalId	ReadOnly metrics and logs	Expected
`vantalect-audit-readonly`	48	Vantalect, LLC (external)	sts:ExternalId	SecurityAudit + ViewOnlyAccess	Expected
`AWSControlTowerExecution`	40	sf-management (…0007)	none	AdministratorAccess	Expected
`OrganizationAccountAccessRole`	42	sf-management (…0007)	none	AdministratorAccess	Expected
`cdk-hnb659fds-deploy-role (CDK bootstrap, --trust)`	29	sf-cicd (…1356)	none	CDK deploy (passes the CloudFormation execution role)	Expected
`sf-security-automation`	48	sf-security-tooling (…3374)	none	SecurityAudit plus remediation actions	Expected
`EKS IAM roles for service accounts (96 roles)`	10	oidc.eks.`<region>`.amazonaws.com (16 cluster providers)	aud + sub pinned to namespace and service account	workload roles	Expected

Role	Accounts	Trusted principal	Conditions	Permissions	Assessment
`gha-* (33 roles)`	9	token.actions.githubusercontent.com	aud + sub pinned to repository and environment	deploy roles	Expected
`gha-cache-writer`	1	token.actions.githubusercontent.com	sub = repo:strataforge/* (any repository in the organization)	build-cache bucket write	Expected
`kettleworth-mes-integration`	1	Kettleworth Integration, LLC (`<contractor-account-id>`)	none	AdministratorAccess	**Finding**, SF-SEC-020
`OrganizationAccountAccessRole (Kestrel, pre-acquisition)`	4	kestrel-legacy-mgmt (…9120), Kestrel's former management account	none	AdministratorAccess	**Finding**, SF-SEC-019
`gha-artifact-publisher`	1	token.actions.githubusercontent.com	aud only (no sub)	ECR push and artifact bucket write for every workload	**Finding**, SF-SEC-003
`gha-deploy-fleet-api`	1	token.actions.githubusercontent.com	aud only (no sub)	EKS and CDK deploy	**Finding**, SF-SEC-003
`gha-deploy-portal`	1	token.actions.githubusercontent.com	aud only (no sub)	S3, CloudFront, and Lambda deploy	**Finding**, SF-SEC-003
`gha-deploy-staging`	1	token.actions.githubusercontent.com	aud only (no sub)	PowerUserAccess	**Finding**, SF-SEC-003
`okta-admin / okta-developer (legacy Okta SAML app)`	6	Okta SAML provider (pre-Identity Center federation)	SAML:aud	AdministratorAccess / PowerUserAccess	**Finding**, SF-SEC-013

The trusts we expected are well built: the vendor roles require an external ID, the EKS and most GitHub roles pin the token's `sub` to a namespace and service account or to a repository and environment. The ones that are findings share a shape. **A trust that checks who the caller is by one weak field, or by none:** 4 GitHub roles check only the token's audience (SF-SEC-003), a contractor's account has administrator access with no

condition (SF-SEC-020), and Kestrel's old management account still has administrator in four Kestrel accounts (SF-SEC-019). One more, `gha-cache-writer`, trusts any repository in the StrataForge GitHub organization; that is narrower than the findings and we have recorded it as expected, with a recommendation to pin it to the build repositories.

Privilege-escalation paths

The review found 12 paths that let a principal with limited permissions reach administrator. We built them from the attached policies and the trust relationships: a principal that can pass a role to a service it can also create a resource in, or that can change a policy, a trust, or a key, can act as the more powerful principal.

Privilege-escalation paths PE-01 to PE-04, with the number of principals that hold each.

ID	Technique	Who	Account	Reaches	How
PE-01	iam:PassRole + lambda:CreateFunction + lambda:InvokeFunction	FleetDeveloper permission set (212)	`fleet-dev`	fleet-dev-admin-exec (Action * on *)	A developer creates a function with the admin execution role and invokes it.
PE-02	iam:PassRole + lambda:CreateFunction + lambda:InvokeFunction	FleetDeveloper permission set (212)	`fleet-staging`	cdk-hnb659fds-cfn-exec-role (AdministratorAccess)	The CDK CloudFormation execution role can be passed to any Lambda function in staging.
PE-03	iam:PassRole + ec2:RunInstances	crucible-research IAM users (58)	`crucible-research`	crucible-batch-admin-profile (Action * on *, read on crucible-data)	A researcher launches an instance with the admin instance profile and reads the role's credentials over IMDSv1.
PE-04	iam:CreatePolicyVersion	jenkins-deployer user (1)	`suspended-legacy-ci`	Account administrator	The CI user can rewrite its own managed policy as Action * on *.

Privilege-escalation paths PE-05 to PE-08, with the number of principals that hold each.

ID	Technique	Who	Account	Reaches	How
PE-05	iam:PutUserPolicy on self, then sts:AssumeRole into Kestrel accounts	Kestrel CI users (34)	`kestrel-legacy-mgmt`	OrganizationAccountAccessRole in four Kestrel accounts (AdministratorAccess)	Any Kestrel CI key can grant itself admin and, through the pre-acquisition trust, administer kestrel-prod and kestrel-data.
PE-06	iam:UpdateAssumeRolePolicy	platform-automation-role (1)	`sf-shared-services`	Any role in sf-shared-services, including OrganizationAccountAccessRole	The automation role can rewrite the trust policy of an admin role to trust itself.
PE-07	iam:CreateAccessKey on other users	sap-basis-admins group (9)	`erp-sap-prod`	sap-backup-admin user (Action * on *)	A Basis administrator can mint a key for the backup admin user.
PE-08	cloudformation:CreateStack + iam:PassRole	FleetDeveloper permission set (212)	`fleet-staging`	cdk-hnb659fds-cfn-exec-role (AdministratorAccess)	A stack with an AWS::IAM::Role resource yields an admin role.

Privilege-escalation paths PE-09 to PE-12, with the number of principals that hold each.

ID	Technique	Who	Account	Reaches	How
PE-09	ssm:SendCommand to admin-profiled nodes	FleetOnCall permission set (64)	`fleet-api-prod`	fleet-api-node-role (Action * on *)	On-call engineers can run commands on EKS nodes whose role is admin.
PE-10	codebuild:CreateProject + iam:PassRole	ci-maintainers group (7)	`sf-cicd`	ci-admin-service-role (AdministratorAccess)	A maintainer runs a build as the admin service role.
PE-11	glue:CreateJob + iam:PassRole	DataAnalyst permission set (38)	`fleet-data-prod`	glue-etl-admin (Action * on *)	An analyst runs a Glue job as an admin role.
PE-12	iam:AttachUserPolicy on self	hackathon IAM users (26)	`sandbox-hackathon`	Account administrator	Every 2024 hackathon user can attach AdministratorAccess to itself.

The widest is the `FleetDeveloper` permission set: 3 of the paths start there, held by 212 people. All 12 are the finding in SF-SEC-018, and the fix is to remove the permissions that make each path possible rather than trust that no one will use them.

Unused roles

Of 5,730 workload roles, **1,922 have not been used in 90 days or more:** 460 have never been used, and 691 have been idle more than a year. An unused role is permissions nobody is watching. It is a Low finding on its own (SF-SEC-048) because nothing is wrong until someone finds it, and the fix is mechanical: IAM Access Analyzer's unused-access findings list each one.

The ten accounts with the most roles unused for 90 days or more.

Account	Unused roles
`fleet-eu-prod`	84
`crucible-training-prod`	79
`fleet-data-prod`	78
`fleet-analytics-prod`	75
`corp-data-warehouse`	69
`erp-sap-prod`	68
`fleet-remote-ops-prod`	66
`digital-twin-prod`	66
`corp-apps`	66
`fleet-ingest-prod`	65

Administrator policies

The review found 14 customer managed policies that grant `Action: *` on `Resource: *`. Each is attached to workload roles, and some also to people. The fix, cutting each to what the workload does, is SF-SEC-017.

Administrator policies, the roles they are attached to, and how many IAM users hold them.

Policy	Account	Roles	People
fleet-api-node-admin	fleet-api-prod	fleet-api-node-role, fleet-api-batch-role	0
ingest-ops-admin	fleet-ingest-prod	telemetry-replay-role	0
crucible-batch-admin	crucible-sim-prod	crucible-sim-batch-role, crucible-sim-node-role	0
research-batch-admin	crucible-research	crucible-batch-admin-profile	0
sap-backup-admin	erp-sap-prod	sap-backup-role	1
mes-integration-admin	mfg-mes-prod	mes-gateway-role	0
kestrel-api-admin	kestrel-prod	perception-api-node-role, perception-api-task-role	9
kestrel-ci-admin	kestrel-legacy-mgmt	jenkins-agent-role	14
kestrel-data-admin	kestrel-data	dataset-loader-role	4
jenkins-deployer	suspended-legacy-ci	jenkins-controller-role	11
platform-automation	sf-shared-services	platform-automation-role	0
hackathon-admin	sandbox-hackathon	hackathon-ec2-role	7
dev-environments-admin	fleet-dev	fleet-dev-admin-exec	0
sandbox-power	sandbox-robotics	sandbox-robotics-exec	0

What we recommend for identity

1. Deactivate and then delete the 161 unused access keys now, and retire the rest in week 4 by moving CI to OIDC roles and services to instance or task roles (SF-SEC-012).
2. Require MFA for every console user, and move the last 439 people to Identity Center (SF-SEC-013).
3. Close the escalation paths and the administrator policies (SF-SEC-017, SF-SEC-018).
4. Remove the contractor and legacy Kestrel trusts (SF-SEC-019, SF-SEC-020).
5. Delete member-account root credentials centrally, and alarm break-glass use (SF-SEC-010, SF-SEC-029, SF-SEC-040).

Deliverable 4: SOC 2 compliance map

Of the 38 criteria mapped, 15 have a gap on the AWS side today, 9 are partly met, and 3 are met. The map uses SOC 2 (2017 Trust Services Criteria, with the revised points of focus of 2022). It covers the Common Criteria (CC) and the Availability (A) and Confidentiality (C) criteria, and for each says what the AWS side looks like today, which findings stand in the way, and the AWS evidence an auditor will ask for.

What this map is and is not

It is a readiness map of the AWS side of StrataForge's controls. It is not an audit opinion, an attestation, or a certification, and Vantalect is not your auditor. The criteria that depend on people, policy, and process outside AWS (the CC1 to CC3 control environment and risk assessment, for example) are marked "Not assessed: organizational", because AWS configuration cannot show them.

How the criteria divide by status.

Status	Criteria	Meaning
Gap	15	An open finding means the control does not operate as the criterion expects
Partially met	9	The control exists but has a gap in coverage or evidence
Met on the AWS side	3	The AWS-side evidence exists today
Inherited from AWS	2	AWS's own SOC 2 report covers it (physical access and media disposal, which the carve-out puts with AWS)
Not assessed: organizational	9	Outside what AWS configuration shows

The criteria

The SOC 2 criteria, their status on the AWS side, and the evidence for each.

Criterion	Title	Status	Open findings	The AWS-side evidence an auditor will ask for
CC1.1	Commitment to integrity and ethical values	Not assessed: organizational	None	Not an AWS-side control
CC1.2	Board independence and oversight	Not assessed: organizational	None	Not an AWS-side control
CC1.3	Structures, reporting lines, and authorities	Partially met	SF-SEC-039, SF-SEC-047	AWS Organizations OU structure with an accountable owner for every account (Team and Owner tags).
CC1.4	Commitment to competence	Not assessed: organizational	None	Not an AWS-side control
CC1.5	Accountability for internal control	Not assessed: organizational	None	Not an AWS-side control
CC2.1	Relevant, quality information	Gap	SF-SEC-022, SF-SEC-047	AWS Config aggregator inventory and a tagging coverage report that names an owner for each resource.
CC2.2	Internal communication of control information	Partially met	SF-SEC-038	Runbooks and PagerDuty routing for Security Hub CSPM and GuardDuty findings, with on-call acknowledgments.
CC2.3	Communication with external parties	Partially met	SF-SEC-050	An alternate SECURITY contact on every AWS account.
CC3.1	Specified objectives	Not assessed: organizational	None	Not an AWS-side control
CC3.2	Identification and analysis of risk	Met on the AWS side	None	This findings register, its ratings, and the risk-acceptance log.
CC3.3	Consideration of fraud risk	Not assessed: organizational	None	Not an AWS-side control

Criterion	Title	Status	Open findings	The AWS-side evidence an auditor will ask for
CC3.4	Changes that affect internal control	Gap	SF-SEC-019, SF-SEC-039	Integration records for the Kestrel acquisition: Control Tower enrollment and removal of pre-acquisition trusts.
CC4.1	Ongoing and separate evaluations	Gap	4, worst High	Security Hub CSPM scores per account and standard, AWS Config compliance, and CloudTrail digest validation.
CC4.2	Communication of deficiencies	Met on the AWS side	None	The remediation roadmap with owners and dates, and the week-six re-scan.
CC5.1	Control activities that mitigate risk	Not assessed: organizational	None	Not an AWS-side control
CC5.2	General controls over technology	Gap	SF-SEC-024, SF-SEC-026, SF-SEC-039	SCP, RCP, and declarative policy documents with their attachment targets; Control Tower enrollment status.
CC5.3	Policies and procedures	Not assessed: organizational	None	Not an AWS-side control
CC6.1	Logical access security, including encryption	Gap	18, worst Critical	IAM Identity Center permission sets and assignments, the IAM credential report, KMS key policies, and encryption defaults.
CC6.2	Provisioning and removal of access	Gap	6, worst Critical	Okta-to-Identity Center SCIM provisioning records, the credential report, and key retirement records.
CC6.3	Least privilege and access changes	Gap	6, worst Critical	Access review exports, IAM Access Analyzer unused-access findings, and permissions boundaries.
CC6.4	Physical access	Inherited from AWS	None	AWS's SOC 2 report covers the data centers (carved out).
CC6.5	Disposal of protected assets	Inherited from AWS	None	AWS media sanitization is covered by AWS's SOC 2 report.
CC6.6	Protection against threats from outside the system boundary	Gap	18, worst Critical	Security group rules open to 0.0.0.0/0 (only the documented ones), EKS endpoint settings, WAF associations, IoT policies.
CC6.7	Restricted transmission and movement of data	Partially met	SF-SEC-034, SF-SEC-036, SF-SEC-052	The RCP that enforces TLS, load balancer listener policies, and ElastiCache in-transit encryption.

Criterion	Title	Status	Open findings	The AWS-side evidence an auditor will ask for
CC6.8	Prevention and detection of malicious software	Gap	SF-SEC-023, SF-SEC-033, SF-SEC-042	GuardDuty Malware Protection and Runtime Monitoring coverage, and Amazon Inspector coverage.
CC7.1	Detection of configuration changes and vulnerabilities	Gap	5, worst High	Inspector findings against remediation SLAs, AWS Config rule compliance, and Security Hub CSPM control status.
CC7.2	Monitoring for anomalies	Gap	8, worst High	GuardDuty in every account and Region, the organization trail with validation, and alarms on root and break-glass use.
CC7.3	Evaluation of security events	Gap	SF-SEC-023, SF-SEC-038	Triage records for paged findings.
CC7.4	Incident response	Partially met	SF-SEC-038	Runbooks, and records of their use and of the quarterly exercise.
CC7.5	Recovery from incidents	Partially met	SF-SEC-045	Restore tests from AWS Backup and the logically air-gapped vault.
CC8.1	Change management	Gap	6, worst Critical	Reviewed pull requests for infrastructure changes, two-person approval of firmware releases, CI role trust conditions.
CC9.1	Mitigation of business-disruption risk	Partially met	SF-SEC-045	AWS Backup plans, cross-Region copies, and backup retention settings.
CC9.2	Vendor and business-partner risk	Gap	SF-SEC-012, SF-SEC-020	An inventory of vendor access: cross-account roles with ExternalId and no vendor IAM users with long-lived keys.
A1.1	Capacity management	Not assessed: organizational	None	Not an AWS-side control
A1.2	Environmental protections, backups, and recovery infrastructure	Partially met	SF-SEC-045	Multi-AZ settings, backup retention, and the logically air-gapped vault for production data.
A1.3	Recovery plan testing	Partially met	SF-SEC-045	Restore-test records for production databases.

Criterion	Title	Status	Open findings	The AWS-side evidence an auditor will ask for
C1.1	Identification and protection of confidential information	Gap	10, worst Critical	Data classification tags, encryption settings, and public-access guardrails for buckets and snapshots.
C1.2	Disposal of confidential information	Met on the AWS side	None	S3 lifecycle rules and snapshot retention settings.

The criteria with the most open findings are the ones an auditor will test first: logical access (CC6.1, 18 findings), protection against outside threats (CC6.6, 18), and confidential information (C1.1, 10). Closing the Critical findings in week 3 removes the most visible items under all three. A SOC 2 Type 2 report covers a period, so the dates a finding was found and closed matter, and the roadmap and the week-six re-scan record both.

Deliverable 5: governance review

StrataForge runs 48 accounts in 10 organizational units on a Control Tower landing zone from April 19, 2021, and the landing zone has drifted. Its guardrail service control policies have been detached since March 14, 2023, 7 accounts are outside Control Tower, and the organization's only service control policies stop nothing. This review covers the structure, the policies, Control Tower, how accounts are created, tagging, delegated administration, and logging and detection coverage, as of July 15, 2026. StrataForge's separate AWS GovCloud (US) organization, which serves its defense programs, was out of scope.

Organizational units

The ten organizational units, their Control Tower registration, and the accounts enrolled in it.

Organizational unit	Accounts	Registered with Control Tower	Enrolled	Owners
Root	1	**No**	0 of 1	Elena Vasquez
Security	3	Yes	3 of 3	Nadia Petrova
Infrastructure	5	Yes	5 of 5	Elena Vasquez, Grace Liu
Workloads/Fleet/Prod	10	Yes	10 of 10	Lucas Ferreira, Sam Okafor
Workloads/Fleet/NonProd	7	Yes	7 of 7	Sam Okafor
Workloads/Crucible	6	Yes	6 of 6	Hiro Tanaka
Workloads/Enterprise	6	Yes	6 of 6	Ana Kowalski, Lucas Ferreira, Mei Chen
Workloads/Kestrel	5	**No**	0 of 5	Jonah Reyes
Sandbox	4	Yes	3 of 4	Elena Vasquez, Hiro Tanaka, Sam Okafor
Suspended	1	**No**	0 of 1	Elena Vasquez

The structure is sensible: security and infrastructure accounts at the top, workloads grouped by business unit, a sandbox OU, and a Suspended OU. The faults are in what is attached to it. The Workloads/Kestrel and Suspended OUs are not registered with Control Tower at all, so none of their accounts receive its guardrails, and the management account sits at the root by design.

Service control policies, resource control policies, and declarative policies today

The policies attached to the organization today. There are no resource control policies and no declarative policies.

Policy	Type	Attached to
`FullAWSAccess`	AWS managed	Root, every OU, every account
`DenyLeaveOrganization`	customer managed, created June 10, 2019	Root

`FullAWSAccess` allows everything, and `DenyLeaveOrganization` stops an account leaving. Together they deny nothing that matters, so any account administrator can disable GuardDuty, delete a trail, or turn off encryption by default. The landing zone's own guardrail SCPs were detached to unblock a failed pipeline deployment and never reattached, as the drift table below shows (SF-SEC-026).

The policies we recommend

The fix in SF-SEC-026 (and the TLS and public-access guardrails in SF-SEC-024 and SF-SEC-052) puts these in place in week 3. They stay inside the current limits: up to 10 SCPs of 10,240 characters, 5 RCPs of 5,120 characters, and 10 declarative policies of 10,000 characters per root, OU, or account.

The guardrails we recommend adding at the top of the organization.

Policy	Type	Attached to	What it does
`sf-protect-security-services`	Service control policy	Root	Denies Stopping or deleting CloudTrail trails and event selectors, AWS Config recorders, GuardDuty, Security Hub CSPM, IAM Access Analyzer, and Inspector, except the platform and security roles
`sf-deny-member-root`	Service control policy	Root	Denies Every action by a member account's root user, except in a root session started with sts:AssumeRoot (the condition `Null: {aws:AssumedRoot: true}`), which is how root tasks run
`sf-protect-account-defaults`	Service control policy	Root	Denies Turning off EBS default encryption; changing S3, snapshot, or AMI Block Public Access, RDS snapshot sharing, or KMS key rotation outside the platform roles; and creating IAM users, access keys, or console passwords outside the PlatformIdentityAdmin role
`sf-require-workload-boundary`	Service control policy	Root	Denies Creating a role or user, or setting a permissions boundary, without the sf-workload-boundary boundary, and editing, deleting, or removing the boundary itself, outside the platform and identity administrator roles
`sf-deny-unused-regions`	Service control policy	Workloads, Sandbox, Suspended	Denies Every action in the 11 unused Regions, except global services, the security services, and the platform roles
`sf-rcp-enforce-tls`	Resource control policy	Root	Denies Requests to S3, KMS, SQS, Secrets Manager, and STS without TLS (aws:SecureTransport false)
`sf-ec2-imdsv2-default`	EC2 declarative policy	Root	Sets Instance metadata defaults for new launches: IMDSv2 required, hop limit 2
`sf-ec2-block-public-sharing`	EC2 declarative policy	Root	Sets Snapshot block public access (block all sharing) and AMI block public access (block new sharing)

Control Tower

Control Tower governs 6 of the 17 enabled Regions (us-east-1, us-west-2, eu-central-1, ap-northeast-1, eu-west-1, ap-southeast-1). Drift is the most useful view of how it came to be this way:

Control Tower drift and what caused it.

Since	Type of drift	What happened
March 14, 2023	SCP detached from a registered OU	The landing zone's guardrail SCPs were detached from all six registered OUs on 03/14/2023 to unblock a failed pipeline deployment and were never reattached.
April 15, 2024	Account moved between OUs	sandbox-hackathon was created in the Organizations console in April 2024 and placed in Sandbox outside Account Factory.
June 3, 2024	Account moved between OUs	suspended-legacy-ci was moved to Suspended in the Organizations console in 2024; its Jenkins estate was retired but the account was never emptied or closed.
November 6, 2023	Landing zone update available	The landing zone has not been updated since 2023; Control Tower reports it two versions behind.

The accounts outside Control Tower are the five Kestrel accounts (invited from Kestrel Vision's organization), `sandbox-hackathon` (created in the console), and `suspended-legacy-ci`. The week-5 repair (SF-SEC-039) re-registers the OUs and enrolls them.

How accounts are created

Accounts by how they were created.

How the account was created	Accounts	Examples
Organizations API, before Control Tower (later enrolled)	30	`sf-log-archive`, `sf-security-tooling`, `sf-forensics`, `sf-network-hub`
Control Tower Account Factory	11	`sf-identity`, `sf-backup`, `fleet-jp-prod`, `digital-twin-prod`
Invited from Kestrel Vision's organization (November 2025)	5	`kestrel-prod`, `kestrel-staging`, `kestrel-dev`, `kestrel-data`
Management account	1	`sf-management`
Organizations console, outside Account Factory	1	`sandbox-hackathon`

Only 11 accounts came from Account Factory. Most of the rest were created through the Organizations API before Control Tower existed and enrolled later, and that is fine; the accounts that were never enrolled are where the baseline is missing.

Delegated administrators

Delegated administrators for organization-wide security services.

Service	Delegated administrator today	Recommended
`guardduty.amazonaws.com`	`sf-security-tooling`	`sf-security-tooling`
`securityhub.amazonaws.com`	`sf-security-tooling`	`sf-security-tooling`
`inspector2.amazonaws.com`	`sf-security-tooling`	`sf-security-tooling`
`sso.amazonaws.com`	`sf-identity`	`sf-identity`
`config.amazonaws.com`	None	`sf-security-tooling` (new)
`access-analyzer.amazonaws.com`	None	`sf-security-tooling` (new)

Security services are administered from `sf-security-tooling` and identity from `sf-identity`, so the management account is kept out of daily use, which is correct. AWS Config and IAM Access Analyzer have no delegated administrator today, unlike the other security services. Week 5 delegates both to `sf-security-tooling`, so every security service is administered from one place and the management account stays out of daily use (SF-SEC-022, SF-SEC-044).

Tagging

Resources that carry no `Team` tag, by type. Databases, functions, and clusters are fully tagged.

Resource type	Total	Untagged	Tagged
Instances	2,861	497	82.6%
Volumes	4,712	1,239	73.7%
Snapshots	4,445	1,032	76.8%
Buckets	317	77	75.7%
Databases	148	0	100.0%
Functions	1,790	0	100.0%
Load balancers	175	5	97.1%
Clusters	16	0	100.0%

There is no tag policy in the organization. The four tags the organization requires are `Team`, `Application`, `Environment`, `DataClassification`, and they are enforced by convention only. The gap is concentrated in a few accounts, mostly Kestrel and sandbox accounts.

The ten accounts with the highest share of untagged resources.

Account	Untagged	Total	Untagged share
`kestrel-legacy-mgmt`	31	31	100.0%
`sandbox-hackathon`	44	44	100.0%
`kestrel-data`	11	13	84.6%
`kestrel-staging`	20	34	58.8%
`sandbox-ml`	23	42	54.8%
`sandbox-robotics`	105	202	52.0%
`kestrel-dev`	93	188	49.5%
`sandbox-platform`	96	200	48.0%
`kestrel-prod`	152	360	42.2%
`sf-network-hub`	6	16	37.5%

A resource with no `Team` tag is a resource nobody will be paged for, and an incident that reaches one has no owner to call. SF-SEC-047 adds the tag policy and a Config rule that finds what is missing.

Logging and detection coverage

Account-Regions covered of 816 by service, today and at the target. Week 5 reaches the target.

Service	Covered today	Share	Target
CloudTrail (organization trail)	816	100.0%	816
GuardDuty	258	31.6%	816
Security Hub CSPM	129	15.8%	816
AWS Config	246	30.1%	816
IAM Access Analyzer	48	5.9%	816
Amazon Inspector	8	1.0%	816

The only detection that runs everywhere is the organization trail, and it dates from 2019: it has no log file validation and no KMS encryption, and its bucket can be written to and deleted from the management account (SF-SEC-025). Everything else runs only in the six Regions with workloads, and GuardDuty is off in every Kestrel account (SF-SEC-023). The week-one scan has the same matrix by kind of account.

Deliverable 6: the remediation roadmap

The roadmap sequences all 54 findings into three weeks of work and a backlog: 309 hours in week 3, 174 in week 4, 116 in week 5, and 566 that remain for StrataForge after week 6. Every item has an owner, an effort, and a date. Risk comes first: every Critical is in week 3, identity follows in week 4, and detection and response go in during week 5, once the noisiest findings are closed and the alerts that remain are worth paging someone for.

The roadmap by week. Backlog hours include the parts of week-3 to week-5 findings that wait on a date or a dependency.

Week	Theme	Starts	Findings	Hours
3	Criticals and guardrails	Jul 27	24	309
4	Identity	Aug 3	9	174
5	Detection and response	Aug 10	11	116
6	Re-scan and handover	Aug 17		
After	Backlog, owned by StrataForge	After Aug 21	10	566

Hours are estimates of the engineering time a team would spend doing each change by hand. In the Hardening, Vantalect writes the pull requests with agents, under review, inside its 110 hours, and the owners review, merge, and roll out, so the hours size the work and are not the time StrataForge's engineers must find. The change reviews in weeks 3 to 5 reset the estimates. Week 3 is the heaviest. If a team cannot carry its share, the item moves to the backlog; for a Critical, StrataForge can instead accept the risk in writing, and the guarantee counts an accepted Critical as closed.

Hours by owner

Effort hours by finding owner and week.

Owner	Week 3	Week 4	Week 5	Backlog	All
Sam Okafor	124	19	8	343	494
Elena Vasquez	118	40	52	173	383
Jonah Reyes	39	6	0	0	45
Hiro Tanaka	10	0	0	12	22
Grace Liu	8	97	0	38	143
Mei Chen	10	4	0	0	14
Nadia Petrova	0	8	56	0	64

The plan, week by week

Week 3 starts July 27, 2026 and ends July 31, 2026; week 4 starts August 3, 2026 and ends August 7, 2026; week 5 starts August 10, 2026 and ends August 14, 2026. The backlog is after August 21, 2026, owned by StrataForge, and each item has its own date.

Every finding in the order it is worked, by week.

Week	Order	Finding	Severity	Title	Owner	Hours	Depends on	Due
Week 3	1	SF-SEC-001	Critical	A 1,127-day-old IAM user key can sign and publish firmware to every robot	Sam Okafor	24	None	Jul 31
Week 3	2	SF-SEC-002	Critical	One IoT policy lets any of 61,400 robots' certificates publish to any robot's command topic	Sam Okafor	40	None	Jul 31
Week 3	3	SF-SEC-003	Critical	4 GitHub Actions deploy roles can be assumed from any GitHub repository	Elena Vasquez	6	None	Jul 31
Week 3	4	SF-SEC-007	Critical	A Kestrel MySQL snapshot with annotation records is restorable by any AWS account	Jonah Reyes	3	None	Jul 31
Week 3	5	SF-SEC-005	Critical	5 EBS snapshots in kestrel-prod are public, and snapshot block public access is off in all 816 account-Regions	Jonah Reyes	6	SF-SEC-026	Jul 31
Week 3	6	SF-SEC-004	Critical	Jupyter is open to the internet on 51 research and sandbox instances, some with no token	Hiro Tanaka	10	None	Jul 31
Week 3	7	SF-SEC-006	Critical	4 Kestrel dataset buckets are public: 468.8 million objects of labeled images from customer sites	Jonah Reyes	14	None	Jul 31
Week 3	8	SF-SEC-008	Critical	Kestrel's Jenkins controller is open to the internet on 8080 and 22	Jonah Reyes	16	None	Jul 31
Week 3	9	SF-SEC-010	Critical	Root users of 3 Kestrel accounts have access keys and no MFA	Grace Liu	8	None	Jul 31
Week 3	10	SF-SEC-009	Critical	RDP is open to the internet on 32 PLM CAD workstations that hold product designs	Mei Chen	10	None	Jul 31

Week	OrderFinding	Severity	Title	Owner	HoursDepends on	Due
Week 3	11SF-SEC-011	High	docs-legacy.strataforge.example points at a deleted S3 website bucket that anyone can claim	Elena Vasquez	3None	Jul 31
Week 3	12SF-SEC-027	High	SSH bastions accept connections from the whole internet	Elena Vasquez	6None	Jul 31
Week 3	13SF-SEC-021	High	The staging OTA test key lets the production pipeline sign, and pilot robots still trust it	Sam Okafor	8None	Jul 31
Week 3	14SF-SEC-024	High	S3 Block Public Access is off at the account level in 31 accounts and on 47 buckets	Elena Vasquez	10None	Jul 31
Week 3	15SF-SEC-026	High	Service control policies stop nothing: only FullAWSAccess and DenyLeaveOrganization are attached	Elena Vasquez	24None	Jul 31
Week 3	16SF-SEC-016	High	All 16 EKS cluster API endpoints accept connections from 0.0.0.0/0	Sam Okafor	21 (+17 in theNone backlog)	Jul 31
Week 3	17SF-SEC-014	High	1,631 instances still allow IMDSv1	Elena Vasquez	31 (+25 in theNone backlog)	Jul 31
Week 3	18SF-SEC-028	High	SSH is open to the internet on 76 development instances	Sam Okafor	18None	Jul 31
Week 3	19SF-SEC-030	High	23 robot certificates are shared by 1,840 robots at 3 customer sites	Sam Okafor	13 (+19 in theNone backlog)	Jul 31
Week 3	20SF-SEC-032	Medium	EBS encryption by default is off in 762 account-Regions; 1,245 volumes are unencrypted	Elena Vasquez	8 (+70 in theNone backlog)	Jul 31
Week 3	21SF-SEC-041	Medium	SSH and RDP are open to the internet on 27 sandbox instances	Elena Vasquez	10None	Jul 31
Week 3	22SF-SEC-053	Low	78 customer managed keys do not rotate	Elena Vasquez	4None	Jul 31
Week 3	23SF-SEC-052	Low	220 buckets accept requests without TLS	Elena Vasquez	6None	Jul 31

Week	OrderFinding	Severity	Title	Owner	HoursDepends on	Due
Week 3	24SF-SEC-050	Low	Account baseline gaps: no security contact in 42 accounts, no support role in 20, CloudShell full access on 195 identities	Elena Vasquez	10None	Jul 31
Week 4	25SF-SEC-013	High	108 IAM users sign in to the console without MFA, 2 of them in the management account	Grace Liu	22SF-SEC-010, SF-SEC-029	Aug 7
Week 4	26SF-SEC-012	High	1,009 active access keys are older than 90 days; 400 are over three years old	Grace Liu	41 (+22 in theSF-SEC-003, SF-backlog)SEC-008	Aug 7
Week 4	27SF-SEC-020	High	A former contractor's AWS account holds AdministratorAccess in mfg-mes-prod with no ExternalId	Mei Chen	4None	Aug 7
Week 4	28SF-SEC-019	High	Kestrel's former management account still administers four Kestrel member accounts	Jonah Reyes	6SF-SEC-008	Aug 7
Week 4	29SF-SEC-018	High	12 privilege-escalation paths lead to administrator in 11 accounts	Grace Liu	28None	Aug 7
Week 4	30SF-SEC-015	High	144 Lambda functions have secret-named environment variables	Sam Okafor	19 (+13 in theNone backlog)	Aug 7
Week 4	31SF-SEC-017	High	14 customer managed policies grant Action * on Resource * to 17 workload roles and 46 IAM users	Elena Vasquez	40None	Aug 7
Week 4	32SF-SEC-029	High	4 more member-account root users have no MFA; 40 use virtual MFA	Grace Liu	6None	Aug 7
Week 4	33SF-SEC-040	Medium	Break-glass access is not alarmed and was last tested in April 2025	Nadia Petrova	8None	Aug 7
Week 5	34SF-SEC-022	High	AWS Config records nothing in 570 account-Regions	Elena Vasquez	10SF-SEC-039	Aug 14
Week 5	35SF-SEC-023	High	GuardDuty is off in 558 account-Regions, including every Kestrel account; most protection plans are off	Nadia Petrova	12None	Aug 14
Week 5	36SF-SEC-025	High	The 2019 organization trail can be stopped from the management account, and its logs can be deleted	Nadia Petrova	20None	Aug 14

Week	OrderFinding	Severity	Title	Owner	HoursDepends on	Due
Week 5	37SF-SEC-038	Medium	Security Hub CSPM runs in only 3 Regions, with FSBP only and no CIS standard	Nadia Petrova	8SF-SEC-022	Aug 14
Week 5	38SF-SEC-039	Medium	7 accounts are outside Control Tower, and the landing zone has drifted	Elena Vasquez	20None	Aug 14
Week 5	39SF-SEC-044	Medium	IAM Access Analyzer covers only us-east-1; 768 account-Regions have no external access analyzer	Nadia Petrova	4None	Aug 14
Week 5	40SF-SEC-042	Medium	Amazon Inspector is off almost everywhere; 132 ECR repositories have no image scanning	Nadia Petrova	8None	Aug 14
Week 5	41SF-SEC-043	Medium	750 default VPCs, most in Regions nobody uses, have open default security groups and network ACLs	Elena Vasquez	10 (+6 in theNone backlog)	Aug 14
Week 5	42SF-SEC-049	Low	MFA delete is off on every bucket	Nadia Petrova	4SF-SEC-025	Aug 14
Week 5	43SF-SEC-051	Low	Server access logging is off on 233 buckets	Elena Vasquez	12None	Aug 14
Week 5	44SF-SEC-054	Low	EKS control-plane logging is off on 15 of 16 clusters, and every cluster still uses the aws-auth ConfigMap	Sam Okafor	8 (+16 in theNone backlog)	Aug 14
Backlog	45SF-SEC-037	Medium	5 EFS file systems are not encrypted at rest	Hiro Tanaka	12None	November 20, 2026
Backlog	46SF-SEC-034	Medium	39 internet-facing ALBs have no AWS WAF; 84 ALBs do not redirect HTTP to HTTPS	Sam Okafor	16None	October 30, 2026
Backlog	47SF-SEC-033	Medium	556 running instances are not managed by Systems Manager; 2,380 run AMIs over a year old	Elena Vasquez	20None	October 30, 2026
Backlog	48SF-SEC-036	Medium	9 ElastiCache replication groups lack encryption at rest or in transit	Sam Okafor	33None	November 20, 2026
Backlog	49SF-SEC-035	Medium	22 database instances store data unencrypted	Sam Okafor	74None	November 20, 2026

Week	Order	Finding	Severity	Title	Owner	Hours	Depends on	Due
Backlog	50	SF-SEC-031	Medium	1,247 Lambda functions run on deprecated runtimes	Sam Okafor	143	None	January 29, 2027
Backlog	51	SF-SEC-045	Medium	31 databases keep one day of backups; 112 resilience checks fail	Sam Okafor	12	None	October 30, 2026
Backlog	52	SF-SEC-046	Low	130 customer managed policies use service-wide wildcards such as s3:*	Elena Vasquez	40	None	December 18, 2026
Backlog	53	SF-SEC-047	Low	2,850 of 14,464 resources have no Team tag, so no owner can be found for them	Elena Vasquez	12	None	December 18, 2026
Backlog	54	SF-SEC-048	Low	1,922 workload roles have not been used in 90 days or more	Grace Liu	16	None	September 4, 2026

What waits, and why

Of the findings, 8 are only partly done by week 6. Each has a part that waits on a date or a dependency outside the engagement: a maintenance window in a factory, a customer site, or an upgrade of agents and SDKs. The part that waits is in the backlog with its own date.

The parts of findings that move to the backlog.

Finding	What remains	Hours	Due
SF-SEC-016	Make the non-production endpoints private once CI runners reach them through the Transit Gateway	17	September 30, 2026
SF-SEC-014	Switch the instances that still make IMDSv1 calls after their agents and SDKs are upgraded	25	October 30, 2026
SF-SEC-030	Re-provision the robots at the Bremen and Ota sites during their maintenance windows	19	September 30, 2026
SF-SEC-032	Migrate the remaining unencrypted volumes through encrypted snapshot copies	70	December 18, 2026
SF-SEC-012	Replace the factory gateway users' keys with IAM Roles Anywhere during factory maintenance windows	22	November 20, 2026
SF-SEC-015	Move the fleet-dev functions' secrets to Secrets Manager	13	September 30, 2026
SF-SEC-043	Replace the open default network ACLs in the working Regions' VPCs	6	October 30, 2026
SF-SEC-054	Move every cluster to access entries and upgrade 1.31 and 1.32 clusters	16	November 13, 2026

© 2026 Vantalect, LLC. Sample deliverable; StrataForge Robotics is fictional.

AWS, Amazon Web Services, AWS Graviton, and the AWS service names are trademarks of Amazon.com, Inc. or its affiliates. Kubernetes is a registered trademark of The Linux Foundation. GitHub is a trademark of GitHub, Inc. NVIDIA and Isaac Sim are trademarks of NVIDIA Corporation. SAP and SAP S/4HANA are trademarks of SAP SE. Okta is a trademark of Okta, Inc. PagerDuty is a trademark of PagerDuty, Inc. SOC 2 is a registered trademark of the American Institute of Certified Public Accountants. CIS Benchmarks is a trademark of the Center for Internet Security, Inc. Vantalect is not affiliated with or endorsed by any of them. StrataForge Robotics is fictional.