Secure *AWS* Multi-Account Baseline

Terraform + Compliance-as-Code

Portfolio
by Amina Jiyu An
Github:
https://github.com/amina0806

Executive Summary

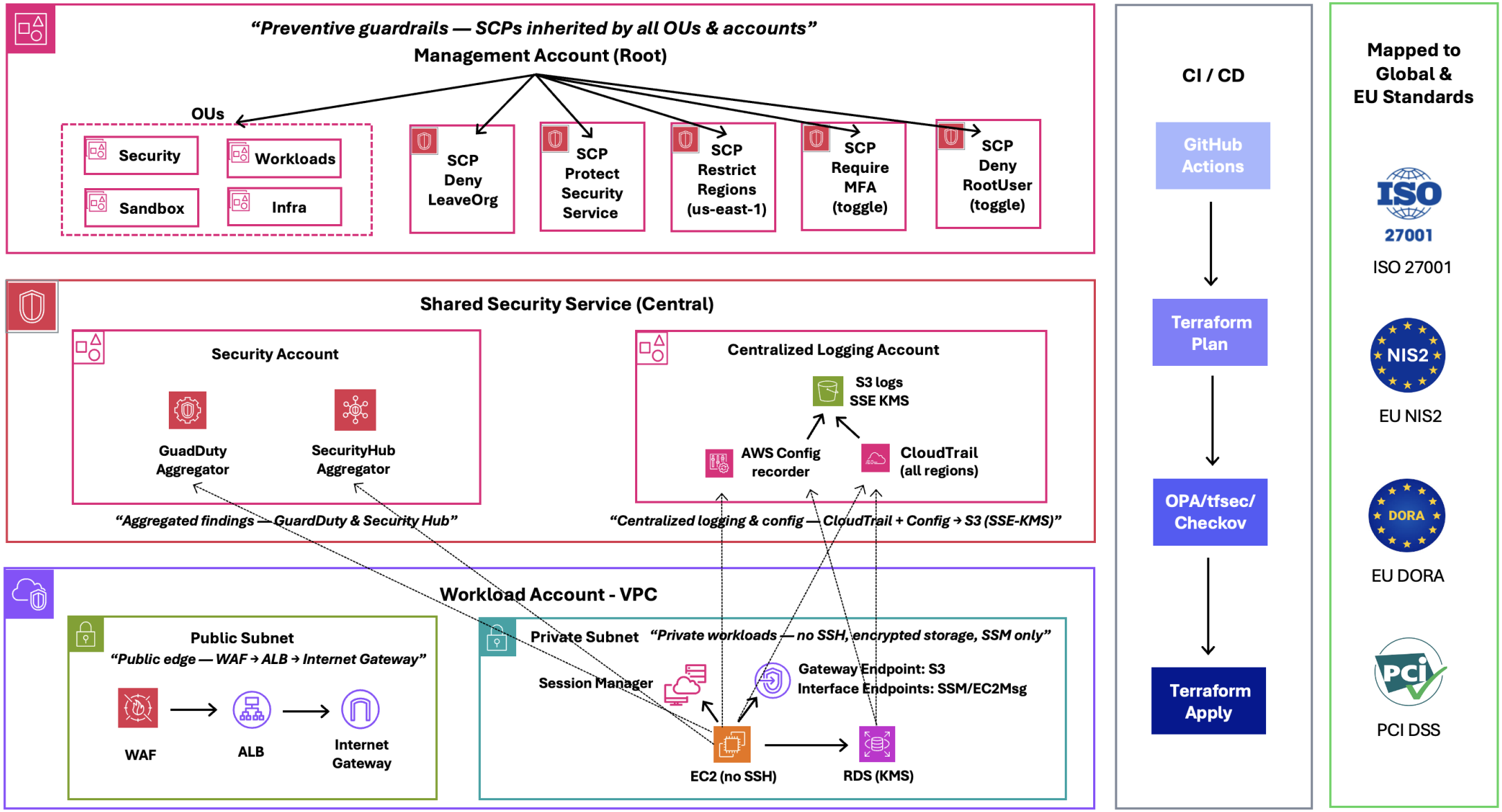
Enterprise AWS Secure Baseline (Terraform + Policy-as-Code)

"This is not theory. Every slide is a proof I built, validated, and enforced in AWS."

This portfolio demonstrates how I designed and enforced a secure AWS environment at **enterprise scale**, combining preventive, detective, and governance controls. Every component is mapped to **international and EU regulatory compliance frameworks** (ISO/IEC 27001,PCI DSS, EU NIS2, EU DORA), proving awareness of both global standards and local regulatory requirements.

Multi-account Governance	Centralized Logging Encryption	Compliance Mapping	Threat detection & CSPM	Policy-as-Code (OPA,tfsec,Checkov)
Multi-account governance with AWS Orgs &SCPs) → ISO 27001 A.5.1 | NIS2 Article 21 | DORA Article 5 (ICT Governance)	CloudTrail + S3/KMS logs → ISO 27001 A.12.4/A.8.15 | NIS2 Article 21(2)(e) | DORA Article 9 (Logging & Monitoring)	AWS Config Conformance Packs → ISO 27001 A.12.1/A.5.14 | NIS2 Article 21(2)(d) | DORA Article 11	GuardDuty + Security Hub → ISO 27001 A.12.6/A.8.8 | NIS2 Article 21(2)(f) | DORA Article 10	Enforce encryption & IAM boundaries → ISO 27001 A.14.2/A.8.28 | NIS2 Article 21(2)(b) | DORA Article 6)

Architecture Diagram



Preventive = SCPs • Detective = CloudTrail/Config/GD/SecHub • Foundational = VPC+Encryption+SSM • Governance = Terraform+OPA

Validated against ISO 27001, EU NIS2, EU DORA, PCI DSS controls

Step1: State Backend

What this proves

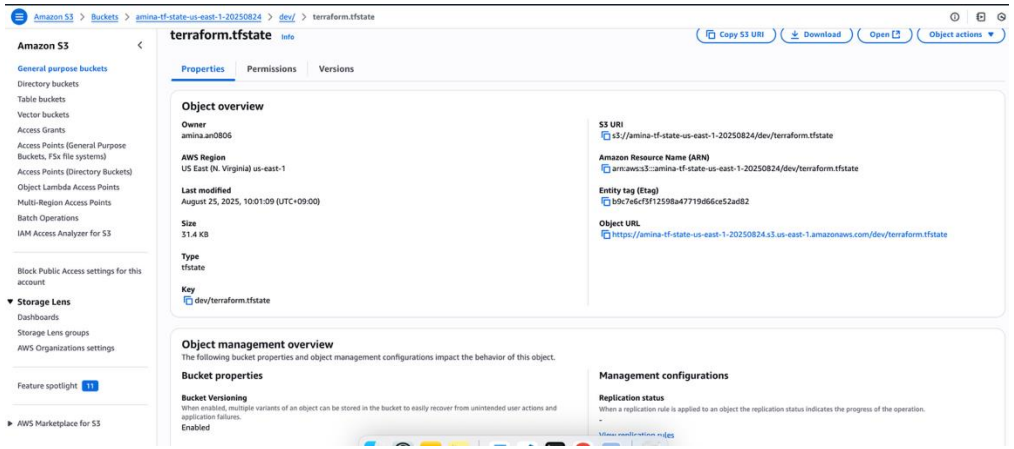
- Secure Terraform state management across accounts.
- **Encryption (SSE-KMS)** protects state confidentiality.
- DynamoDB locking prevents concurrent writes / corruption.

Controls

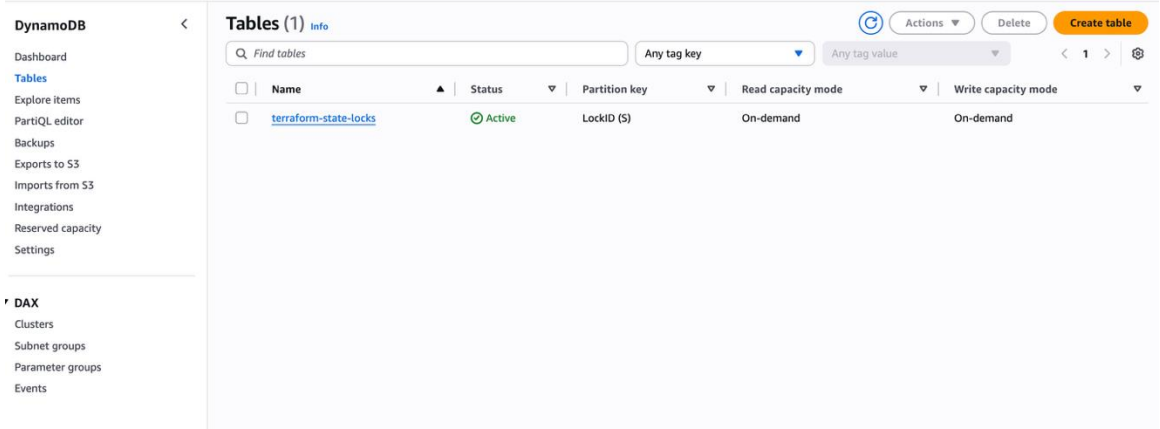
- **ISO 27001:** A.8.20, A.8.23, A.8.16 → 2022: 8.24, 5.23, 5.15
- **NIS2:** Article 21(2)(e) — Protection and encryption of data
- **DORA:** Article 9 — ICT systems and security requirements

Proofs / Screenshots

S3 bucket (SSE-KMS enabled)



DynamoDB state locking



Note: Ensures **tamper-resistant, segregated state** → critical for enterprise IaC.

Step2: Centralized Logging

What this proves

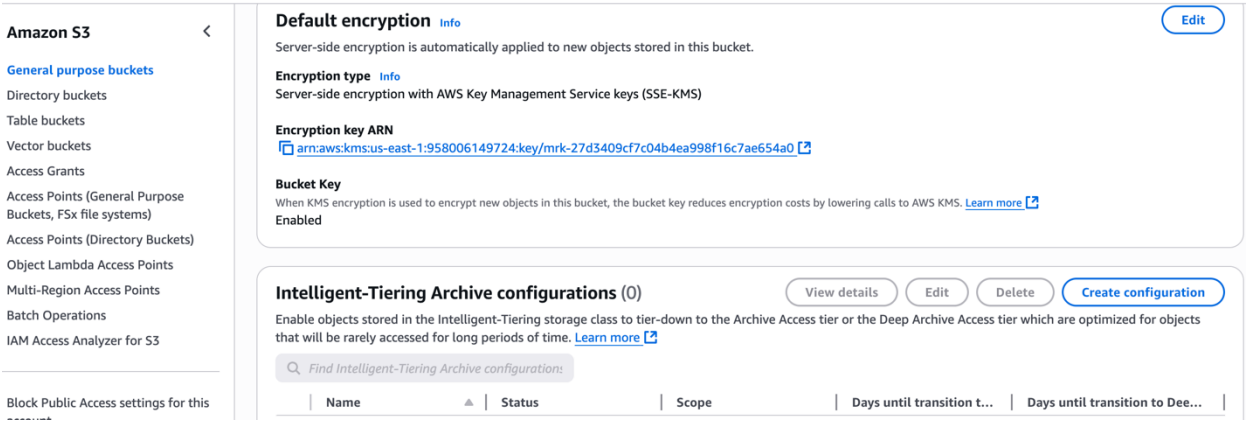
- Enterprise-wide **visibility** into all AWS activity.
- CloudTrail & AWS Config logs are **centralized, encrypted, immutable**.
- S3 bucket with **KMS CMK + versioning** → no accidental/intentional log deletion.

Controls

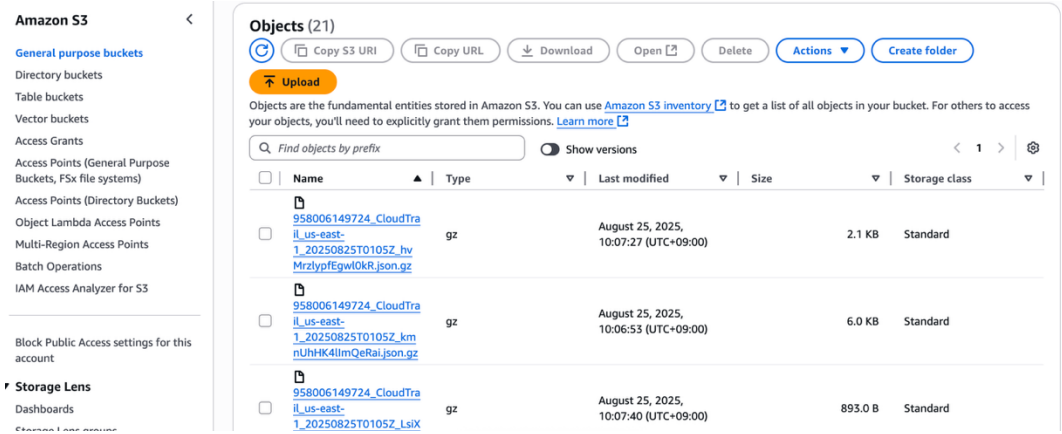
- **ISO 27001:** A.12.4 → 2022: 8.15
- **EU NIS2:** Art. 21(2)(d) Event logging & monitoring, Art. 23 Incident reporting
- **EU DORA:** Art. 8 Monitoring & detection, Art. 23 ICT incident reporting

Proofs / Screenshots

Log bucket encryption (SSE-KMS)



CloudTrail Logs in S3



Note: Logging is the foundation for monitoring & audit evidence.

Step3: AWS Config & Conformance Packs

What this proves

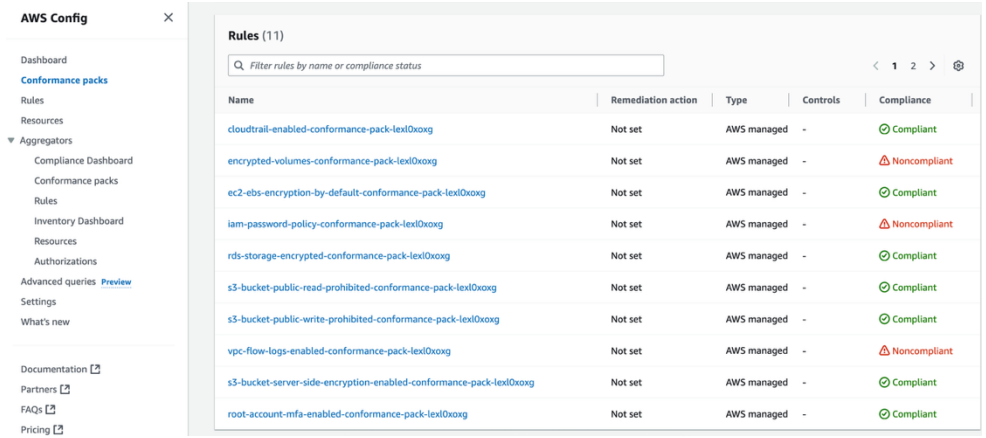
- **Detects misconfigurations** → flags non-compliance in near real-time.
- Conformance Pack with 11 security baseline rules (passwords, MFA, encryption, logs)

Controls

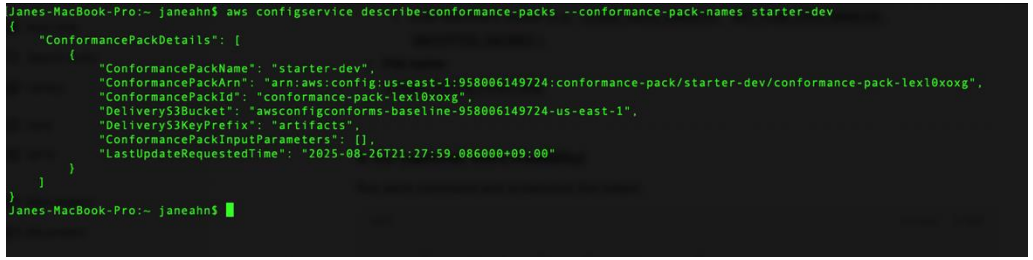
- **ISO 27001:** A.12.1, A.18.2.2 → 2022: 5.14, 5.36
- **EU NIS2:** Art. 21(2)(a) Governance & policies, Art. 21(2)(c) Risk assessment
- **EU DORA:** Art. 8 ICT risk management (continuous compliance monitoring), Art. 8 Governance of ICT assets

Proofs / Screenshots

Config Rules Evaluations



CLI Conformance Pack



```
Janes-MacBook-Pro:~ janeahn$ aws configservice describe-conformance-packs --conformance-pack-names starter-dev
{
    "ConformancePackDetails": [
        {
            "ConformancePackName": "starter-dev",
            "ConformancePackArn": "arn:aws:config:us-east-1:958006149724:conformance-pack/starter-dev/conformance-pack-lexl0xoxg",
            "ConformancePackId": "conformance-pack-lexl0xoxg",
            "DeliveryS3Bucket": "awsconfigconforms-baseline-958006149724-us-east-1",
            "DeliveryS3KeyPrefix": "artifacts",
            "ConformancePackInputParameters": [],
            "LastUpdateRequestedTime": "2025-08-26T21:27:59.086000+09:00"
        }
    ]
}
Janes-MacBook-Pro:~ janeahn$
```

Note: Provides ongoing evidence for audits. Moves from reactive audits → proactive continuous compliance

Step4: Security Hub & GuardDuty

What this proves

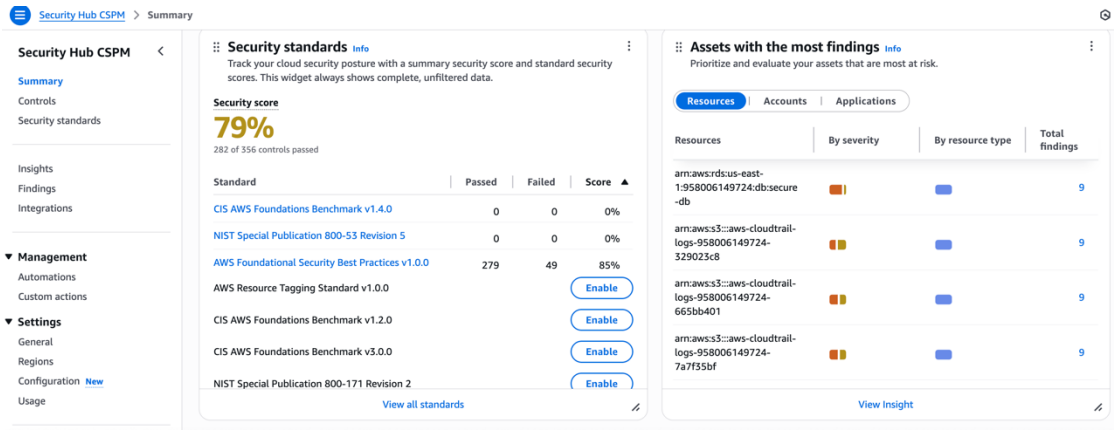
- Unified **threat detection + compliance aggregation.**
- Security Hub consolidates findings (CIS, PCI DSS).
- GuardDuty detects **anomalous network and account behavior.**

Controls

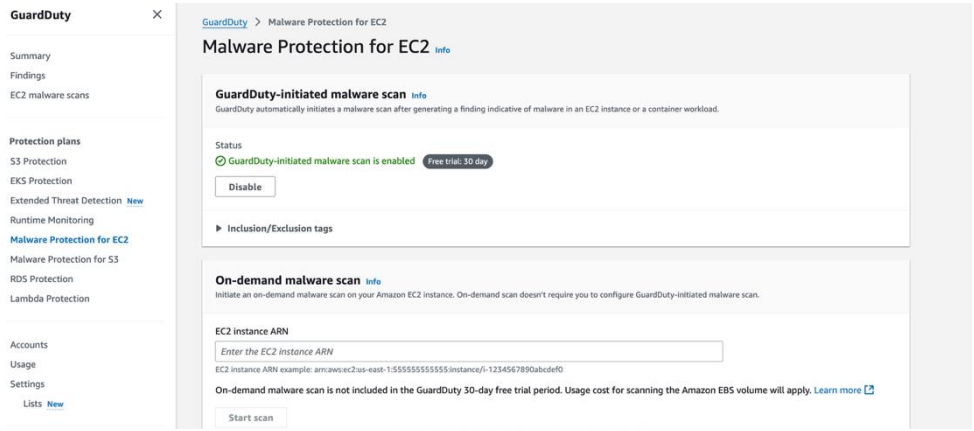
- **ISO 27001:** A.12.6, A.16.1 → 2022: 8.8, 5.25
- **EU NIS2:** Art. 21(2)(d) Event monitoring,
  Art. 21(2)(g) Incident handling, Art. 23 Incident reporting
- **EU DORA:** Art. 8 Monitoring & detection,
  Art. 23 ICT incident reporting

Proofs / Screenshots

Security Hub Summary



GuardDuty Detector ON



Note: Provides central view of risk posture across all accounts

Step5: Policy-as-Code (OPA, tfsec, Checkov)

What this proves

- **Automated governance** before provisioning.
- Prevents deployment of insecure resources (unencrypted S3, missing MFA, etc.).
- CI/CD gate → code must pass tfsec, Checkov, OPA rules **before apply.**

Controls

- **ISO 27001:** A.14.2, A.12.1.2, A.18.2.3 → 2022: 8.28, 5.14, 5.35
- **EU DORA:** Article 8 ICT risk management (secure configuration, CI/CD enforcement), Article 30 Third-party risk management (dependency checks, SBOM)

Proofs / Screenshots

❌ OPA eval fail (GuardDuty missing)

```
Janes-MacBook-Pro:tf-aws-secure-baseline janeahn$ cat > plan-no-guardduty.json <<'EOF'
> {
>   "resource_changes": []
> }
> EOF
Janes-MacBook-Pro:tf-aws-secure-baseline janeahn$ opa eval -d policies-as-code/opa -i plan-no-guardduty.json 'data.terraform.security.m
tty
[
  "GuardDuty is not being enabled in this plan (missing aws_guardduty_detector).",
```

✅ OPA eval pass (all checks)

```
Janes-MacBook-Pro:tf-aws-secure-baseline janeahn$ opa eval -d policies-as-code/opa -i plan-pass.json 'data.terraform.security.result' -f pretty
{
  "count": 0,
  "messages": [],
  "passed": true
}
Janes-MacBook-Pro:tf-aws-secure-baseline janeahn$
```

✅ OPA unit tests pass

```
Janes-MacBook-Pro:tf-aws-secure-baseline janeahn$ opa test  policies-as-code/opa -v
policies-as-code/opa/policies_test.rego:
data.terraform.security.test_guardduty_missing_detector_denies: PASS (2.876834ms)
data.terraform.security.test_iam_group_missing_boundary_denies: PASS (3.512292ms)
data.terraform.security.test_guardduty_disabled_denies: PASS (3.549917ms)
data.terraform.security.test_securityhub_both_missing_denies: PASS (3.554584ms)
data.terraform.security.test_securityhub_account_only_denies_subscription: PASS (3.612166ms)
data.terraform.security.test_iam_role_missing_boundary_denies: PASS (3.591125ms)
data.terraform.security.test_iam_user_missing_boundary_denies: PASS (3.576917ms)
data.terraform.security.test_s3_requires_sse_denies: PASS (626.75µs)
data.terraform.security.test_s3_with_sse_allows: PASS (4.415333ms)
data.terraform.security.test_guardduty_enabled_allows: PASS (1.988041ms)
--------------------------------------------------------------------------------
PASS: 10/10
```

Note: Shifts compliance left → security embedded in development pipeline.

Step6: Organizations & SCPs

What this proves

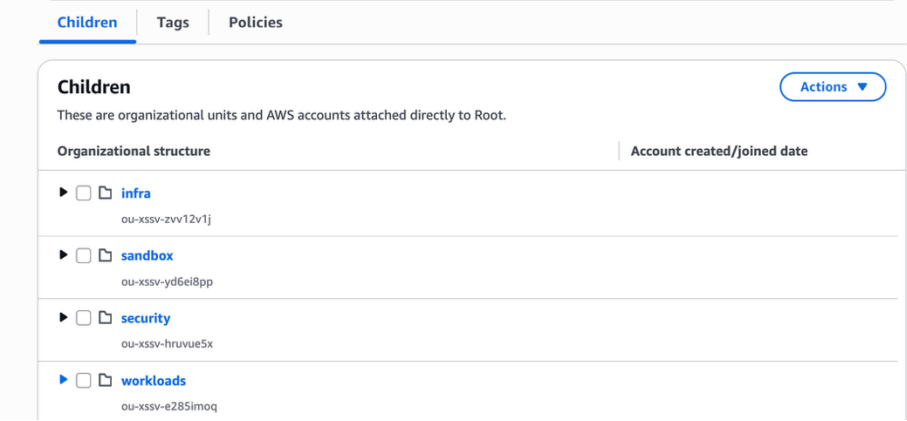
- **Preventive guardrails** enforced Org-wide.
- **SCPs restrict dangerous actions**: Deny leaving Org , Protect CloudTrail/Config/SecHub/GD ,Restrict regions (only us-east-1), Require MFA for IAM writes, Deny root user access (toggle)

Controls

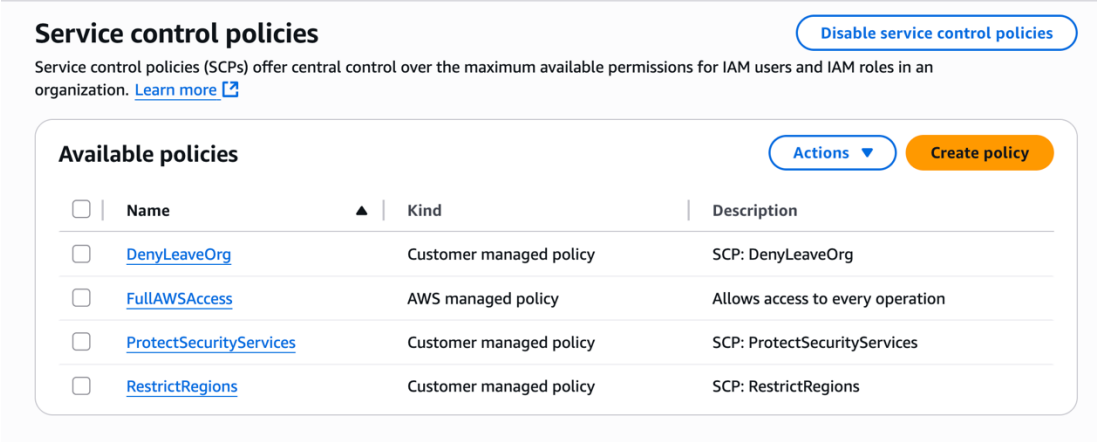
- **ISO 27001:** A.5.1.1, A.9.2.3, A.9.2.1 → 2022: 5.1, 5.18, 5.17
- **EU NIS2:** Article 21(2)(a) Governance & policies, Article 21(2)(b) Access control
- **EU DORA**: Article 8 ICT risk management (preventive governance), Article 30 Third-party governance

Proofs / Screenshots

Org OUs (security, infra, workloads, sandbox)



Root with SCPs Attached



Note: These guardrails prevent violations at source — stronger than detective controls.

Compliance Mapping

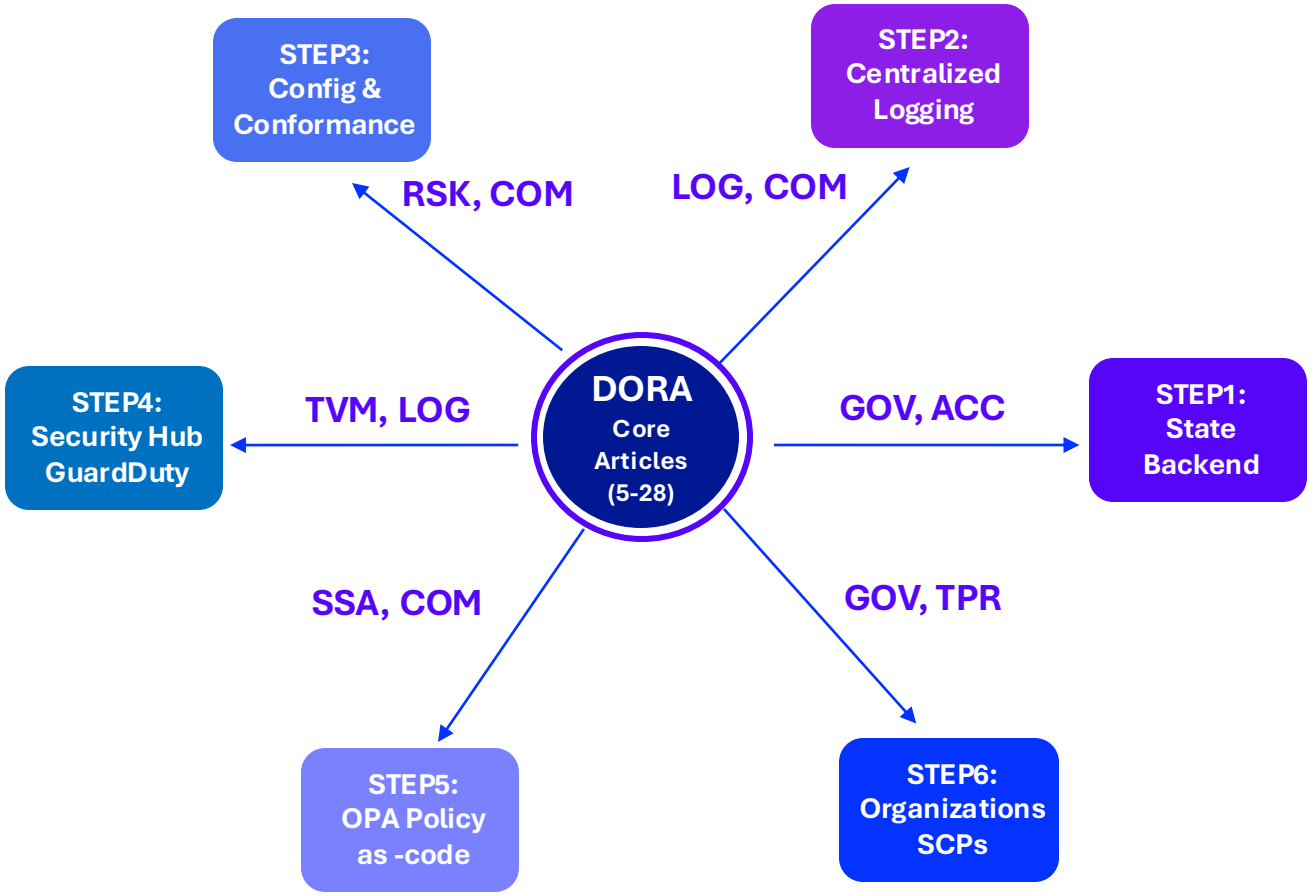
Step	Implementation Example	ISO/IEC 27001	NIST CSF	PCI DSS	EU NIS2	EU DORA
1 — State Backend	S3 backend SSE-KMS, DynamoDB lock	A.8.20 → 8.24 (crypto), A.8.16 → 5.15 (access control)	PR.DS-1 (Data-at-rest protected)	Req. 3.5 (Encryption of cardholder data)	Art. 21(2)(b) — Secure access & asset management	Art. 8 — ICT risk management (data security)
2 — Centralized Logging	CloudTrail, CloudWatch, KMS	A.12.4 → 8.15 (logging), A.8.20 → 8.24	DE.AE-1 (Anomalous activity detected)	Req. 10 (Logging & monitoring)	Art. 21(2)(d) — Event logging & monitoring	Art. 23 — Incident detection & reporting
3 — Config & Conformance	Config rules, Conformance packs	A.12.1 → 5.14, A.18.2.2 → 5.36	ID.RA-1 (Risks identified)	Req. 11.5 (File integrity monitoring)	Art. 21(2)© — Risk assessment & treatment	Art. 8 — ICT risk management, compliance
4 — Security Hub & GuardDuty	Threat detection, incident dashboard	A.12.4 → 8.15, A.12.6 → 8.8	DE.CM-1 (Continuous monitoring)	Req. 12.10 (Incident response)	Art. 21(2)(d) — Threat detection & response	Art. 23 — ICT incident handling, reporting
5 — OPA Policy-as-Code	Terraform plan eval, CI/CD enforcement	A.12.6 → 8.8, A.18.2.2 → 5.36, A.9.2.3 → 5.18	PR.IP-3 (Secure dev lifecycle)	Req. 6.3 (Secure development practices)	Art. 21(2)(a) — Governance & policies	Art. 8 — ICT controls automation
6 — Organizations & SCPs	DenyLeaveOrg, Protect Security Services, RestrictRegions	A.5.1.1 → 5.1, A.12.4 → 8.15, A.9.2.3 → 5.18	RS.MI-1 (Mitigation executed)	Req. 7.2 (Restrict access to cardholder data)	Art. 21(2)(b) — Access control, least privilege	Art. 8 — ICT governance; Art. 30 — Third-party/vendor risk

Legend

- **NIS2**: Art. 21 = Cybersecurity risk management (access control, monitoring, governance) | Art. 23 = Incident detection & reporting
- **DORA**: Art. 8 = ICT risk management & controls | Art. 23 = ICT incident reporting | Art. 30 = Third-party/vendor management
- **NIST CSF**: ID = Identify | PR = Protect | DE = Detect | RS = Respond | RC = Recover
- **PCI DSS**: Req. = Requirement

DORA Mapping Highlights

Mapping Portfolio Steps to DORA Articles & Core Domains



DORA Core Domains (Article Mapping)

GOV = Governance & Oversight • **RSK** = ICT Risk Management • **COM** = Compliance & Testing
• **LOG** = Logging & Incident Reporting • **TVM** = Threat & Vulnerability Management • **SSA** = Secure Systems & Applications
• **ACC** = Access Control • **TPR** = Third-Party & Outsourcing Risk

CI/CD Enforcement with Policy-as-Code

What this proves

- GitHub Actions pipeline runs **security checks automatically**.
- Every pull request triggers **tfsec, Checkov, OPA** before merge.
- Pipeline ensures **"no code is applied without passing security gates."**

Controls

- **ISO 27001:** A.14.2 Secure coding, A.12.1.2 Change management, A.18.2.3 Technical compliance review → 2022: **8.28, 5.14, 5.35**
- **EU DORA:** Art. 8 ICT risk management (secure configuration and testing in CI/CD pipelines), Art. 30 Third-party risk management (audit and code integrity requirements)

Proofs / Screenshots

GitHub Actions YAML workflow

```yaml
name: terraform-security-checks

on:
  pull_request:
    paths:
      - "**/*.tf"
      - "policies-as-code/**"
      - ".github/workflows/plan.yml"
  workflow_dispatch:

jobs:
  security_checks:
    runs-on: ubuntu-latest
    env:
      # Default guess; we'll auto-correct this below if the folder doesn't exist
      WORK_DIR: envs/dev
      OPA_DIR: policies-as-code/opa
```

CI Badge Green

terraform-security-checks passing

Enterprise AWS Secure Baseline (Terraform + PaC)

This project demonstrates **how to design and enforce a secure AWS environment at enterprise scale.** It includes:

- Multi-account setup with AWS Organizations & Service Control Policies
- Centralized logging (CloudTrail, CloudWatch, S3 + KMS)
- AWS Config Conformance Packs for compliance monitoring
- Security Hub & GuardDuty as Cloud Security Posture Management (CSPM) tools
- Policy-as-Code (OPA/Rego) to enforce encryption, IAM boundaries, and security service activation

Compliance Mapping: **ISO/IEC 27001 Annex A (2013 & 2022), Saudi NCA ECC, UAE NESA IAS**

ISO/IEC 27001 Annex A — Control Mapping (2013 → 2022)

Key takeaway: IaC merges are blocked until all security checks pass — proving NIS2/DORA-ready DevSecOps governance.

Conclusion

“Every slide is a proof I built, validated, and enforced in AWS.”

- ✓ **Compliance → ISO 27001, NIS2, DORA, PCI DSS**
- ✓ **Security → IAM/PAM, SCPs, Logging, Security Hub**
- ✓ **DevSecOps → Policy-as-Code in CI/CD pipelines**

Amina Jiyu An

Cloud Security & Compliance Engineer