AWS Well-Architected Framework

Digital Sovereignty Lens

Digital Sovereignty Lens: AWS Well-Architected Framework

Table of Contents

Digital Sovereignty Lens - AWS Well-Architected Framework

Publication date: **January 26, 2026** (???)

This paper describes the Digital Sovereignty Lens for the AWS Well-Architected Framework. It provides AWS customers with a set of Well-Architected best practices and guidance on digital sovereignty.

As organizations worldwide accelerate their cloud adoption, many face increasing regulatory pressures and stakeholder demands to maintain control over their digital assets and data.

While there is no single definition of digital sovereignty and specific requirements may vary by country, it is broadly understood as a term used by nation states to *assert sovereignty* over the *digital assets* they *own or regulate*.

Digital sovereignty in practice

In practice, digital sovereignty manifests as requirements aligned to three broad areas.

1. **Compliance:** We find nations asserting digital sovereignty through legislations, standards, or by empowering their regulatory agencies to come up with compliance requirements. Compliance requirements originate from:
   - Cybersecurity related directives, standards, and guidelines (for example, NIS 2 is a directive, and NIST SP 800-53 Rev. 5 is a standard and a guideline).
   - Data privacy legislations (for example, Regulation (EU) 2016/679, also known as the European Union General Data Protection Regulation (GDPR)).
   - Industry or sectoral regulations (for example, the Health Insurance Portability and Accountability Act (HIPAA) is a US federal law that protects sensitive patient health information (PHI)).
   - Emerging regulations (for example, the European Union AI Act).

   Compliance is a key driver when it comes to sovereign workloads.
2. **Control**: Beyond standard compliance, some nations now mandate additional controls, driving towards more stringent data security and data residency related outcomes. These requirements translate to technical measures including:
   - Confidential compute solutions (for example, AWS Nitro Enclaves)

  - Externally managed key stores (for example, AWS KMS external key stores)
  - Dedicated data security solutions
  - Privacy enhancing technologies
  - Data residency controls
  - Hybrid compute environments at the edge (for example, AWS Outposts)

3. **Continuity**: Some nations seek to exercise more independence and choice while making technology decisions. They don't want to be locked into proprietary technologies or punitive license terms. This also extends to having uninterrupted access to infrastructure, services and skills required to support their digital footprint. Achieving continuity through greater self-sufficiency and resiliency is therefore a key outcome of digital sovereignty initiatives.

Together, the *three Cs* form a mental model of the outcomes expected from a digital sovereignty initiative. Be aware that there is an alternative viewpoint on digital sovereignty, described as a combination of data sovereignty, operational sovereignty, and technology sovereignty. The three Cs align with that viewpoint but are anchored towards clear business-focused outcomes rather than on abstract notions.

Scope

Sovereign workloads must be compliance-aligned, auditable, transparent, interoperable, portable, and survivable. However, not every system needs to exhibit each of those qualities. For example, you may not want to invest in interoperability and portability initially choosing instead to focus on the compliance and control related measures.

The Digital Sovereignty Lens provides a framework to assist in designing workloads that address sovereignty requirements while still using the flexibility of the AWS Cloud.

Lens availability

Custom lenses extend the best practice guidance provided by AWS Well-Architected Tool. AWS WA Tool allows you to create your own custom lenses, or to use lenses created by others that have been shared with you.

To begin reviewing your digital sovereignty workload, download and import the Digital Sovereignty Lens into AWS Well-Architected Tool from the public AWS Well-Architected custom lens GitHub repository.

Definitions

This document contains common definitions and terminology used across the AWS Well-Architected Digital Sovereignty Lens pillars.

- **Data residency:** The physical or geographic location where data is stored and processed. In sovereign workloads, this typically means restricting data storage and processing to specific AWS Regions within approved jurisdictions. For example, an organization subject to European regulations might require customer data to remain within European Union (EU) regions (like eu-west-1 or eu-central-1) and never replicate to regions outside the EU.
- **Data sovereignty:** The concept that data is subject to the laws and governance structures of the nation or region where it is collected or stored. This extends beyond physical location to include legal jurisdiction, data access rights, and regulatory adherence. For architects, this means designing systems where data handling, encryption key management, and operational access comply with local laws, even when using global cloud services.
- **General Data Protection Regulation (GDPR):** EU regulation on data protection and privacy that applies to an organization processing personal data of EU residents. Key requirements include data minimization, right to erasure, data portability, and mandatory breach notification. Solutions must implement technical controls for consent management, data deletion, and cross-border transfer restrictions.
- **Health Insurance Portability and Accountability Act (HIPAA):** US legislation providing data privacy and security provisions for safeguarding medical information. Requires administrative, physical, and technical safeguards including encryption, access controls, audit logging, and business associate agreements. Cloud workloads handling PHI must implement HIPAA-eligible services and maintain comprehensive audit trails.
- **Payment Card Industry Data Security Standard (PCI-DSS):** Information security standard for organizations that handle credit cards from major card brands. Requires network segmentation, encryption of cardholder data, regular security testing, access controls, and monitoring.
- **California Consumer Privacy Act (CCPA):** California state statute intended to enhance privacy rights and consumer protection for California residents. Grants consumers rights to know what personal information is collected, delete their data, and opt out of data sales. Technical implementations require data discovery, classification, and automated deletion capabilities.
- **Identity and access management (IAM):** Framework for managing identities and access permissions. In sovereign contexts, IAM must enforce identity, role, and locatiion-based access controls with detailed audit logs of access attempts and actions.

- **Attribute-based access control (ABAC):** Access control model that uses attributes (tags) attached to resources, users, and environment context to make authorization decisions. Enables fine-grained permissions based on conditions like data classification, user department, project, or geographic location. For sovereign workloads, ABAC allows dynamic enforcement of data residency by denying access to resources tagged with specific regions or sensitivity levels. For example, you can create policies that allow users to access only resources tagged with their home region or data classification level.
- **Role-based access control (RBAC):** Access control model based on user roles. Simplifies permission management by grouping users into roles (such as database administrator, security auditor, or application developer) and assigning permissions to roles. Critical for sovereign workloads to enforce separation of duties and limit access to sensitive data.
- **Multi-factor authentication (MFA):** Security process requiring multiple verification methods (something you know, something you have, something you are) before granting access. Essential for sovereign workloads to block unauthorized access, especially for privileged operations or access to sensitive data.
- **Least privilege:** Security principle of granting only the minimum necessary permissions required to perform a specific task. In practice, this means using IAM policies, policy boundaries, organization level controls, temporary credentials, and just-in-time access rather than broad administrative permissions.
- **Zero trust:** Security model that requires verification for every access request, regardless of whether it originates inside or outside the network perimeter. Assumes breach and verifies explicitly using identity, device information, location, and other contextual information. For sovereign architectures, this means continuous authentication and authorization for data access.
- **Preventative controls:** Controls designed to block an event from occurring before it happens. Examples include service control policies (SCPs) that deny launching resources in non-approved regions, IAM policies that block cross-region data replication, or network ACLs that block traffic to unauthorized destinations.
- **Proactive controls:** Controls designed to block the creation of noncompliant resources during deployment. Examples include AWS CloudFormation Guard rules that validate infrastructure as code template, or AWS Config conformance packs that check resource configurations before provisioning.
- **Detective controls:** Controls designed to detect, log, and alert after an event has occurred. Examples include AWS Config rules that identify noncompliant resources, AWS CloudTrail logs that record API calls, or Amazon GuardDuty findings that detect suspicious activity. These provide visibility and enable rapid response to compliance violations.

- **Responsive controls:** Controls designed to drive remediation of adverse events or deviations from security baselines. Examples include AWS Systems Manager Automation documents that automatically remediate noncompliant resources, AWS Lambda functions that respond to security findings, or AWS Config remediation actions that restore compliant configurations.
- **Defense in depth:** Layered security approach implementing multiple controls at different levels (network, application, data, and identity). If one control fails, others provide backup protection. For sovereign workloads, this might include network isolation, encryption, access controls, and monitoring working together.
- **Key management service (KMS):** Service for creating and controlling encryption keys used to encrypt data. In sovereign contexts, a KMS must use customer-managed keys (CMKs) with key material that never leaves the approved region and key policies that restrict access to authorized personnel in approved locations.
- **Hardware security module (HSM):** Physical device that provides tamper-resistant storage and cryptographic operations for encryption keys.
- **Confidential computing:** Technology that protects data during processing using isolated compute environments such as secure enclaves (like AWS Nitro Enclaves). Enables processing of sensitive data while keeping it encrypted in memory, protecting against privileged users and malicious software.
- **Data loss prevention (DLP):** Strategy and tools for blocking unauthorized data transfers or exfiltration. Includes content inspection, policy enforcement, and blocking of sensitive data transmission through email, web uploads, or API calls. Critical for blocking accidental or malicious data leakage from sovereign boundaries.
- **Immutable storage:** Storage that cannot be modified or deleted for a specified retention period. Implemented using features like S3 Object Lock or legal hold. Essential for regulatory adherence, forensic investigation, and protecting against malicious or un-intended deletion.
- **Write once read many (WORM):** Storage configuration that allows data to be written once but read multiple times, blocking modification or deletion. Used for regulatory adherence, legal holds, and archival storage where data integrity is paramount.
- **Privacy-enhancing technologies:** Technical measures to protect personal data while enabling its use. Includes techniques like tokenization, pseudonymization, differential privacy, homomorphic encryption, and secure multi-party computation. Allows organizations to derive value from data while minimizing privacy risks.
- **Protected health information (PHI):** Health information that can be linked to a specific individual, including medical records, treatment information, and payment data. Subject to HIPAA regulations requiring strict access controls, encryption, and audit logging.

- **Personally identifiable information (PII):** Data that can identify a specific individual, such as names, email addresses, social security numbers, or IP addresses. Requires special handling under various privacy regulations including GDPR, CCPA, and sector-specific laws.
- **Data classification:** The process of categorizing data based on sensitivity levels (such as public, internal, confidential, or restricted). In sovereign architectures, classification drives technical controls like encryption requirements, access restrictions, geographic boundaries, and retention policies. For example, PII might require encryption at rest and in transit, while public data might have fewer restrictions.
- **Data lineage:** The ability to track data movement and transformations throughout its lifecycle, from creation through processing, storage, and eventual deletion. This is critical for sovereignty as it provides auditable evidence that data never left approved boundaries and was processed according to regulatory requirements.
- **Content delivery network (CDN):** Distributed network of servers that deliver content to users based on geographic location. For sovereign workloads, CDNs must be configured to serve content only from approved regions and block caching of sensitive data in unauthorized locations.
- **Failover:** Process of switching to a redundant or standby system upon failure of the primary system. In sovereign architectures, failover targets must be in approved regions, and failover procedures must adhere to data residency requirements throughout the recovery process.
- **Mean Time to Recovery (MTTR):** Average time required to repair a failed component or system and restore service. Critical metric for business continuity planning.
- **Business continuity:** Capability of an organization to continue delivery of products or services at acceptable predefined levels following a disruptive incident. For sovereign workloads, continuity plans must account for regional constraints, locality of operational teams, and regulatory requirements during disaster scenarios.
- **Audit trail:** Chronological record of system activities that provides documentary evidence of operations, procedures, or events. Must be tamper-proof, comprehensive, and retained according to regulatory requirements. Includes logs of data access, configuration changes, authentication events, and administrative actions.
- **Data protection authority (DPA):** Regulatory body overseeing data protection regulations within a jurisdiction (such as CNIL in France or ICO in the UK). Organizations must report data breaches to the relevant DPA and may be subject to audits and enforcement actions.
- **Third-party risk management (TPRM):** Process of identifying, assessing, and mitigating risks associated with third-party vendors and service providers. Critical for sovereign workloads to verify that vendors meet data residency requirements, have appropriate security controls, and comply with relevant regulations.

- **Open protocols:** Publicly documented communication standards that enable interoperability between systems from different vendors. Examples include HTTPS, SMTP, and SAML. Important for sovereign architectures to avoid vendor lock-in and enable portability across cloud providers or on-premises infrastructure.
- **Open data formats:** Publicly documented file and data formats that can be read and written by multiple tools and platforms without proprietary restrictions. Examples include JSON, XML, CSV, Parquet, and ORC. Critical for sovereign workloads to maintain data portability and interoperability.
- **Open table formats:** Vendor-neutral table formats that provide ACID transactions, schema evolution, and time travel capabilities for data lakes. Examples include Apache Iceberg, Apache Hudi, and Delta Lake (with open-source version). These formats enable interoperability across different compute engines (like Spark, Presto, or Flink) and cloud platforms, allowing organizations to avoid lock-in to proprietary data warehouse software. For sovereign architectures, open table formats facilitate data portability, and provide flexibility to change analytics platforms while maintaining data sovereignty requirements.

Design principles

The Digital Sovereignty Lens outlines five general design principles targeted to address a set of key challenges organizations encounter when navigating sovereignty concerns. The key challenges explain why organizations take certain actions, answering the question: Why are we doing what we are doing? This is followed by key practices that organizations should consider in order to meet those challenges.

Detailed best practices mapped to the design principles can be found under the individual pillars.

Note that sovereign-by-design principles build on top of the existing design principles defined under the operational excellence, security, reliability and performance efficiency pillars of the AWS Well-Architected Framework.

Design principles

- Apply standardized enforceable controls
- Establish adequate security posture in line with data sensitivity levels
- Design for continuous compliance
- Design for interoperability and portability
- Design for survivability

Apply standardized enforceable controls

Nations use compliance as a mechanism to assert sovereignty over the digital assets they own or regulate. As a result, workloads must comply with local data privacy directives, industry-specific regulations and cybersecurity standards. By applying standardized enforceable controls, you establish consistent guardrails across your environments and provide demonstrable evidence to auditors.

Key challenges:

- Interpreting complex compliance needs and translating them to a set of effective technical controls requires deep domain knowledge, is time-consuming, and is prone to errors.
- Organizations develop compliance spreadsheets and policy documents that outline standards for developers. However, paper-based policies create significant enforcement challenges.

Inconsistent and erroneous implementations frequently emerge across different teams, creating potential security and regulatory risks.

- The scope and complexity of regulations have been increasing over time, coupled with the challenge that many jurisdictions have either overlapping or contradictory regulatory requirements. The *cost of compliance* is therefore increasing.

Key practices:

Consider the following key practices to meet the challenges outlined above.

- **Document regulatory requirements:** Create a compliance-aligned matrix or traceability document that maps specific regulatory requirements to the technical controls and operational practices. This provides an auditable trail tying your practices directly to mandates.
- **Set up automated controls:** Use software or services that provide standardized compliance controls out-of-the-box. Standardized controls are less likely to be misinterpreted and reduce the risk of erroneous or inconsistent implementations. The deployment of standardized, automated controls is known as policy as code (PaC). Implement PaC to codify and enforce regulatory requirements throughout your CI/CD pipelines, and continuously enforce those requirements through automated guardrails, both during software development and infrastructure provisioning. When combined with modern DevOps practices, policy as code plays a major role in reducing both the risks and costs associated with compliance.
- **Detect and mitigate compliance violations early:** Embed preventative and proactive controls into CI/CD pipelines and infrastructure as code workflows. This stops non-conforming resources from being deployed in the first place.
- **Balance compliance with business agility:** Embedding too many preventative and proactive controls too early in the software development lifecycle can slow down delivery. Start with detective controls, then gradually change the evaluation modes to be more preventative and proactive in nature, as the runtime characteristics of the feature being delivered starts becoming clearer.

Establish adequate security posture in line with data sensitivity levels

With an ever-increasing number of data-intensive workloads, organizations want full visibility of where data is stored, control how and with whom data is shared, and audit who has access and for what duration.

Key challenges:

- Managing data at scale requires improving the discoverability, understandability, accessibility, and trustworthiness of individual datasets. At the same time, most jurisdictions require adherence to strict data privacy legislations, data protection requirements, and specific data residency mandates.
- Inadequate protection of citizens' data exposes businesses to severe financial and reputational damage when personally identifiable information (PII), protected health information (PHI), or confidential information is compromised. Organizations remain cautious about migrating to the cloud due to perceived control concerns, often selecting on-premises solutions that deliver less speed and flexibility than cloud alternatives.
- Organizations apply a defense in depth strategy to authorize access to data, a best practice that aligns with secure by design principles. However, as use cases multiply, the number of policies also increases rapidly. This proliferation of policies creates technical debt, policy overlaps, and security gaps that pose challenges for data access governance.

Key practices:

Consider the following key practices to meet the challenges outlined above.

- **Locate and classify sensitive data:** Locate sensitive data using automated discovery, classification, and cataloging, augmented with human-in-the-loop processes. Build an exhaustive data inventory.
- **Apply security and privacy controls:** Define trust boundaries by implementing strict access permissions and network controls. Restrict sensitive information sharing to verified accounts and environments. Apply data obfuscation techniques like tokenization or masking to balance data utility with security controls. Verify access models by analyzing who has access to what, and reconcile this with actual operational and business needs.
- **Track data flows:** Track data movement both across and within trust boundaries. Monitor traffic within network segments and across private and public facing network interfaces. Use data lineage software to track information flow through data pipelines and storage systems.
- **Validate and improve controls:** Use threat modeling to verify the effectiveness and coverage of security and privacy controls applied to protect sensitive data. Source and integrate threat intelligence to augment threat detection.
- **Build evidence:** Retain network flow logs, usage logs, access logs, application logs, and security findings for the long-term aligning with regulatory needs. Use immutable storage options (write

once read many (WORM)) to protect the chain of evidence. Retain access-related audit trails long-term.
- **Build in privacy and transparency:** Implement consent management systems to track user preferences for personal data use. Collect only what is required, and clearly inform users about how data is collected, processed, and retained. Create data retention policies that comply with regulations and delete data when no longer needed. Build systems that support user rights protection as mandated by data privacy legislations, including data access, correction, deletion, and portability. Consider conducting data protection impact assessments (DPIAs) to identify privacy risks involved in processing personal data.

Design for continuous compliance

Maintaining an effective digital sovereignty posture involves continuous discovery and remediation of non-compliant resources. Adversarial actors often exploit compliance gaps to exfiltrate confidential data or disrupt operations, affecting nations and their citizens.

Key challenges:

- The cost of remediating resources increases greatly when you discover non-compliance in the later stages of the software development lifecycle. Late-stage design changes and dependency replacements disrupt development and introduce potential risks.
- Point-in-time attestations and certifications are just a snapshot and don't represent the real state of compliance. A workload that meets regulatory requirements today may fall out of compliance the next day.
- Auditors seek evidence in the form of screenshots, reports, and audit trails. This is disruptive and forces teams to divert valuable engineering time towards gathering evidence.
- Compliance professionals and engineering teams often face resistance in their efforts to return to requirement adherence because the next audit may not be due for months. This leaves teams responsible for owning and managing risks over an extended time period.
- Applications fail compliance audits despite meeting their direct requirements when underlying systems contain security gaps or when dependencies develop new vulnerabilities. For example, a web application that meets compliance requirements may fall out of compliance when its database software misses critical security patches.

Key practices:

Consider the following key practices to meet the challenges outlined above.

- **Introduce a culture of compliance:** Integrate compliance into software development activities and operational processes rather than adding it later. Apply compliance checks throughout the entire development lifecycle to identify and mitigate risks before incidents occur. Cultivate a culture of continuous compliance, maintaining systems and processes that are consistently audit-ready and aligned with regulatory standards. Implement regular compliance training for staff. Without a continuous compliance-aligned mindset, organizations risk accumulating technical debt, security vulnerabilities, and regulatory violations over time. Compliance can't be an isolated, periodic effort. It requires comprehensive integration across people, processes, and technology to be sustainable and effective in the long term.
- **Be audit ready:** Monitor compliance status continuously to reduce audit disruption and improve security posture. Use automated tools to gather compliance-related evidence on a continuous basis. Use standardized reporting templates and generate reports on-demand.
- **Build in remediations:** Prioritize building automated remediations. When automated remediations are in place, return to compliance is quick and predictable. This removes the need for seeking approvals or unplanned expenses.
- **Empower local teams:** Local teams are best placed to understand regional regulatory needs. To improve regulatory adherence, make local engineering teams self-sufficient, engage with local law firms and regulatory authorities to reduce ambiguity around legislations, and hire local talent to better understand regional sensitivities.
- **Establish vendor compliance:** Define specific compliance requirements for vendors based on data handling, security protocols, and regional regulations. Require vendors to demonstrate adherence to these requirements through certification and regular reporting on security measures. Track vendor adherence through automated monitoring tools, regular audits, and dashboards.

Design for interoperability and portability

For some organizations, being self-sufficient and reducing lock-ins with their technology partners gives them more flexibility, translating to more choices. Interoperability and portability play a key role in this endeavor.

Although interoperability and portability sound similar, they address different concerns. Portability refers to how smoothly a workload and its data can be moved from one infrastructure to another. Interoperability focuses on how many code and configuration changes are needed for the workload to function, after it has been ported.

Key challenges:

- Many jurisdictions are developing policies to maintain uninterrupted access to the infrastructure, services, and skills needed for continued digital operations. These initiatives aim to promote local technology providers, accelerate AI innovation, and grow local economies. To comply with these requirements, organizations must future-proof their technology stacks by designing systems with portability and interoperability in mind. Portable and interoperable systems provide the flexibility to switch service providers when regulatory requirements or business needs change.
- Organizations operating in jurisdictions that face ongoing conflicts or large-scale infrastructure disruptions require strategies to maintain business continuity. One strategy involves deploying digital infrastructure to allied countries and jurisdictions where operations can continue uninterrupted. Interoperable and portable technology stacks provide organizations with multiple deployment options and the ability to respond quickly when geopolitical or infrastructure conditions change.
- Organizations don't want to be locked into technology that can quickly become legacy or where the provider is prone to making sudden and drastic changes to pricing or licensing terms. An interoperable and portable codebase allows them much greater maneuverability in such situations.

Key practices:

Consider the following key practices to meet the challenges outlined above.

- **Define interoperability goals:** Align your interoperability requirements with relevant regulations like the European Digital Operational Resilience Act (DORA), data residency needs, and local circumstances. Having clear goals guides your approach.
- **Build abstractions:** Incorporate abstractions into your code and configurations to deploy workloads consistently across regions and infrastructure types. This approach reduces the need for significant rework when deploying to new environments. Consider aligning with open-source technologies, open standards, and open data formats to widen your portability and interoperability options. Alternatively, consider using infrastructure-agnostic independent software vendor (ISV) solutions.
- **Plan ahead for data portability:** Evaluate data center exit policies, the costs involved, tools required, and the available network bandwidth to build a data migration plan if needed.
- **Test for interoperability:** Test your workloads across multiple environments to verify interoperability. Run identical test suites in each environment to verify consistent functionality across deployment locations.

Design for survivability

Organizations exposed to geopolitical risks and large-scale infrastructure disruptions want to be able to maintain business continuity by deploying their digital infrastructure to other regions, countries or jurisdictions.

Key challenges:

- In large organizations, applications have specific Recovery Time Objective (RTO) and Recovery Point Objective (RPO) targets that may conflict with broader business continuity goals. Technology teams and business stakeholders frequently develop separate definitions of *minimum restorable service* and activation timelines, creating disconnected recovery plans. This misalignment becomes particularly problematic when multiple technology systems and engineering teams support a single business function.
- Restoring workloads and entire systems from backups may take hours or even days. This may require several manual steps and many post-restoration validations before the system can serve live traffic again. Many systems are not designed to be restored using automated scripts. This is especially true for legacy non-idempotent systems that rely on remote procedure call (RPC) style synchronous interactions. When in-flight transactions fail, engineers must perform manual rollbacks and complex data cleanups to restore system integrity.
- Organizations neglect full disaster recovery (DR) testing due to its complexity and resource demands. DR testing requires cross-team coordination, third-party service integration, data synchronization, and legacy systems availability. Even when DR tests are conducted and teams log discovered issues in a risk register, they may not be prioritized and addressed. This can lead to fragile systems and hidden vulnerabilities.

Key practices:

Consider the following key practices to meet the challenges outlined above.

- **Align with business continuity goals:** Adopt a defined disaster recovery strategy, aligning with organizational business continuity goals. Define what a minimum restorable service is. Define timelines for full service restoration.
- **Design systems with DR as a stated goal:** Understand system dependencies and fault isolation boundaries, and document your recovery path. Test for disaster recovery. Prioritize and test critical paths of recovery. Record outcomes of DR testing. Use root cause analysis to identify causes of failure and develop mitigations. Develop a roadmap to sunset legacy systems that pose risks to your recovery objectives.

- **Prepare for DR:** Identify key stakeholders. In addition to stakeholders involved in performing the actual recovery from a disaster (such as engineers, technical support, and executives), you should also have a list of key internal stakeholders, a list of critical vendors, third-party suppliers, and even key customers who might be most affected.
- **Report irregularities:** Incorporate regulatory requirements for incident response, and breach notification into your disaster recovery and business continuity plans.
- **Plan for contingencies:** Develop business continuity strategies for system failures. Implement offline and semi-offline operation modes to maintain critical functions during severe disruptions like undersea cable outages. Create procedures to restore workloads after offline periods of operation.

Scenarios

Digital sovereignty encompasses the principles and practices that customers use to maintain control over their data, infrastructure, and operations within specific jurisdictions. The following scenarios represent common situations where digital sovereignty requirements apply. Review these scenarios to identify which are relevant to your organization or solution.

1. **Design and develop nationally-critical digital infrastructure:** Digital infrastructure serving the daily needs of citizens must be secure, resilient, and highly available. Examples include digital payment platforms, border control systems, securities and commodities trading platforms, healthcare systems, and other public service platforms.

   Disruptions to nationally important infrastructure can cause significant inconvenience to citizens. When designing and developing these workloads, consider the *three Cs*: compliance, control, and continuity. First, identify and baseline your regulatory requirements. Second, implement necessary controls to protect personally identifiable information (PII) and protected health information (PHI). Third, define the minimum restorable service levels needed to maintain business continuity. These measures support both regulatory adherence and reliable service delivery.
2. **Set up and operate workloads in new jurisdictions:** Businesses expand their global footprint by launching new digital products worldwide. These products require adaptation to meet regional regulatory standards, data protection requirements, and adhere to cultural sensitivities.

   Consider setting up self-sufficient engineering teams in each jurisdiction. These teams should manage their own infrastructure, avoiding the need for cross-border remote access. Recruit talent from the target jurisdiction, as these professionals bring insights into local customs and regulations. Partner with regional legal and privacy experts to improve compliance with jurisdiction-specific requirements.
3. **Protect citizens' data by complying with local data export controls and privacy legislations:** Sovereign nations own citizens' data. Businesses collecting personal data must comply with regional data privacy legislations and only allow authorized cross-border data transfers.

   Handling PII and PHI requires careful planning. Define trust boundaries and authorize and record data transfers across your trust boundaries. Protect data at rest and in transit, and apply supplemental measures if required. Apply privacy by design principles to manage data at scale. Create a culture of compliance within the organization and be consistently ready for audits.

Every jurisdiction has explicit controls over export of data. However, some jurisdictions allow data to be exported using bi-lateral adequacy arrangements or similar mechanisms. Read the Navigating GDPR Compliance on AWS to understand your options regarding data transfer between the European Union (EU) to the United States (US).

4. **Restrict who can provide operational support, to what extent, and from where:** Allow only authorized personnel, resident within specific geographies and subject to local laws, to provide operational support.

   Providing operational support from approved locations can improve your compliance posture in relation to regional data privacy laws and related data export controls. Build a strong identity foundation and apply principles of least privilege. Decide who needs to access what, why do they need it, from where, and for how long.

   Understand residency and citizenship requirements, audit standards, and other legal and economic implications involving the location of your support center and staff. Regularly review and update access policies based on changing requirements and in compliance with local laws.

5. **Reduce the impact of unexpected disruptions:** Assess potential impacts of reduced access to critical software and services, physical infrastructure, and technical skills required to support your digital footprint. Create resiliency and continuity plans to protect against disruptions from international trade disputes, intellectual-property right conflicts, and unforeseen geopolitical events.

   Consider where you want your disaster recovery (DR) site to be located. Your DR plan must support business continuity when faced with disruptions brought about by regional wars and conflicts, power grid failures, network outages, cybersecurity issues, and climate-related events.

   Consider aligning with open source technologies, open standards (publicly available specifications free from proprietary restrictions), and open data formats to enhance portability and interoperability options. Build abstractions into your code and configurations to deploy workloads consistently across AWS Regions, AWS edge locations, or other managed infrastructure. This approach reduces significant rework if you need to replace dependencies. Alternatively, use infrastructure-agnostic independent software vendor (ISV) solutions to gain more independence and flexibility when porting workloads to different environments.

6. **Align with policy-led technology initiatives:** Nations aspire to be self-sufficient by supporting and nurturing a thriving landscape of local technology providers. This creates more choice and competition in the market, creating room for innovation and stimulates local economic growth.

However, some nations or their designated regulators may mandate the use of domestically developed technology stacks to accelerate this process.

Follow regulatory mandates and align with national and regional policy initiatives. Engage with regulators and policy makers to understand the long and mid-term approach of putting their decisions into practice.

7. **Navigate data disclosure requests:** Organizations are often required to assist law enforcement agencies to disclose information related to subjects of interests. Data disclosure requests may originate from local as well as foreign enforcement agencies.

   Develop a formal process to accept and track data disclosure requests using secure digital channels. Based on the request, derive the minimum amount of data required to comply with an order, the most secure method of data transfer, and the timelines involved. Work with your technology service providers and delegate their share of responsibilities.
8. **Implement new technology through digital sovereignty initiatives:** Find ways of taking advantage of emerging technology without compromising on sovereign controls. Organizations that fully grasp sovereignty-related requirements can enter new regulated markets with confidence, build customer trust, adapt quickly to changing regulatory landscapes, and maintain operational resilience when faced with unforeseen disruptions.

   Consider setting up a center of excellence (CoE) or a *practice area* to address sovereignty-related requirements and challenges. Include experts from legal, technical, compliance, and security domains. A digital sovereignty practice must be equipped to handle multidisciplinary challenges and respond in a variety of ways, not just with technical solutions.

Operational excellence

Sovereign-by-design workloads must be compliant, auditable, transparent, secure, and privacy-aware. This section provides an overview of the design principles, questions, and operational best practices required for implementing and validating those qualities.

Topics

Definitions

The following are operational excellence-specific definitions that complement this lens' general definitions.

- **Preventative controls**: These controls are designed to block an event from occurring. For example, this service control policy (SCP) denies launch of an EC2 instance type unless they are of type t2.micro.
- **Proactive controls**: These controls are designed to block the creation of noncompliant resources. For example, this AWS CloudFormation Guard rule blocks an EC2 instance from being deployed if its security group allows SSH ingress.
- **Detective controls**: These controls are designed to detect, log, and alert after an event has occurred. For example, AWS Config provides a wide selection of detective controls for you to use.
- **Responsive controls**: These controls are designed to drive remediation of adverse events or deviations from your security baseline.
- **Privacy-aware workload**: A privacy-aware workload is designed and implemented with privacy as a core principle throughout its entire lifecycle. It proactively protects user data and respects individual privacy rights through technical, organizational, and design measures.

Design principles

- **Design to be compliant:** Set fulfilling sovereign compliance requirements as a design goal. Sovereign compliance in this context refers to national cybersecurity standards or frameworks, data privacy legislations, and industry regulations with technology implications, such as the EU Digital Operational Resilience Act (DORA). These requirements may also be technology-specific, such as requirements related to the use of cloud infrastructure services. Define your compliance metrics and build solutions that:
  - Apply automated controls to block noncompliant resources from being provisioned.
  - Detect drifts from baseline and forward findings to a single pane of glass.
  - Send notifications to stakeholders and downstream applications.
  - Run automated remediation to restore adherence. Noncompliant resources incur remediation costs and lead to security vulnerabilities. These vulnerabilities expose systems to intrusions and cyberattacks. The consequences can include prolonged outages, reputational damage, and regulatory penalties.
- **Maintain a compliance-aligned posture with multilayered controls:** Maintain a compliance-aligned posture using preventative, proactive, and detective controls. Moreover, controls are best implemented and maintained using a federated operating model. For example, central teams can deploy preventative controls at an organizational level, while product teams who are closer to the data can set up a mix of proactive and detective controls aligned with data protection and data privacy needs of the application.
- **Compliance is a shared responsibility:** Compliance requires application and solution owners across your organization to address their responsibilities in detecting and remediating noncompliant resources. It demands deep domain, regulatory, and technical knowledge, which is best scaled by sharing responsibilities through a federated operating model.
- **Make audits less disruptive:** Auditors require conclusive evidence demonstrating compliance with technical and operational requirements. This includes reports, documents, screenshots, code, and configurations. Audit activities must maintain business continuity without requiring developer intervention. To minimize disruption, implement automatic collection and aggregation of audit-related evidence and provide auditors with the ability to generate reports on-demand.

Organization

DSOPS01: How does your organization address compliance requirements?

Compliance is a shared responsibility requiring collaboration among specialized teams and delegation at an organization level. For example, security consultants establish data privacy and data protection policies. DevSecOps teams develop reusable compliance controls that codify those policies. Developers then select and deploy compliance controls to meet requirements specific to their workloads. This workflow enables each team to contribute their expertise while maintaining clear boundaries of responsibility.

Best practices

- DSOPS01-BP01 Establish a compliance function
- DSOPS01-BP02 Delegate compliance responsibilities
- DSOPS01-BP03 Implement compliance training and awareness
- DSOPS01-BP04 Establish vendor compliance management

DSOPS01-BP01 Establish a compliance function

Organizations operating in the cloud need an effective compliance function to meet regulatory requirements, protect data, and maintain stakeholder trust.

Desired outcome: A unified, organization-wide compliance framework that continuously monitors, enforces, and validates adherence to relevant standards and regulations.

Common anti-patterns:

- Lack of clear compliance policies and procedures.
- Teams prioritize compliance only when there is an impending audit.
- Current compliance posture is not fully visible. Extensive search and discovery is required to identify compliance gaps.
- Limited involvement of compliance teams in defining security policies or during operational readiness review (ORR) of workloads.

Benefits of establishing this best practice:

- Streamlined processes and technology enable rapid adaptation to new laws, minimizing disruption.
- Clear documentation and centralized systems simplify audits.
- Better decision-making based on objective compliance risk awareness. Trained employees and predictive technologies (for example, artificial intelligence (AI)) can identify risks early and reduce violations.
- Reduced possibility of penalties leading to increased trust from customers, partners, and stakeholders.
- Highly visible security posture through consistent application of compliance controls and long-term storage of activity logs.
- Reliability in compliance can differentiate your organization in regulated industries, giving you improved confidence to expand into new territories and industries.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Start by developing a compliance-first culture where security and privacy are embedded into every aspect of the organization's technology usage. This requires executive sponsorship, clear governance structures, and a shift from reactive to proactive compliance management. Balance automation with human oversight, and use AWS tools while recognizing where third-party solutions may be necessary.

Goals of a compliance function:

- Establish clear ownership, accountability, and decision-making processes before implementing technical controls.
- Prioritize controls based on data sensitivity and regulatory impact.
- Solution architecture and designs must embed automated compliance checking into CI/CD pipelines. Regularly validate those checks using automated test cases.
- Make audits self-service. Gather evidence for audits continuously. Analyze audit logs and security findings in near real-time. Make reports available to auditors on-demand.
- Document and communicate compliance boundaries between your teams and your technology service providers.
- Scale compliance by setting organization-wide guardrails while empowering teams with the ability to define their own controls.

Implementation steps

Setting up and managing a compliance function that scales across the organization is a multi-step, multi-year endeavor. The following guidance outlines a few of the most important steps you might need to consider.

1. **Establish a compliance governance structure:**
    - Form a Cloud Compliance Center of Excellence (Cloud Compliance CoE or CCCoE) with executive sponsorship. Include experts from legal, operations, cybersecurity, data security, business, and technology domains. Alternatively, work towards adding compliance capabilities into your existing Cloud Center of Excellence.
    - Document core compliance processes. Outline your entire compliance lifecycle, spanning discovery, baselining, development, operationalization, monitoring, remediation, and evolution of regulatory requirements and associated controls.
    - Define compliance roles (for example, compliance officer, security architects or consultants, business unit champions, developers, and auditors).
    - Create a responsible, accountable, consulted, informed (RACI) matrix for compliance responsibilities across the organization. An example RACI matrix (for illustration only) is as follows:

Activity	CCoE	Business Unit Champions	Dev Teams	Security Consultants	Auditors
Define compliance standards	A/R	C	I	C	I
Implement controls	A	R	R	C	I
Monitor compliance	R	R	I	R	I
Remediate issues	A	R	R	C	I

Activity	CCoE	Business Unit Champions	Dev Teams	Security Consultants	Auditors
Report compliance status	R	C	I	C	A
Conduct audits	I	I	C	C	R

R = Responsible, A = Accountable, C = Consulted, I = Informed

- Develop compliance policies and standards aligned with business objectives. For example, if your business goal is to onboard healthcare customers and hold protected health information (PHI) data, consider Health Insurance Portability and Accountability Act (HIPAA) regulations.
- Create regionalized standard operating procedures to support data subject requests, reporting data breaches and more broadly, managing incidents originating from non-compliance. For example, here is guidance for data subject requests issued by the UK Information Commissioner's Office.
- Define your audit support process. Consider questions such as:
  - What does our certification roadmap look like?
  - How should we resource specific audit processes?
  - How will evidence be captured and preserved?
  - How will re-certification and re-attestation be managed?

2. **Conduct compliance requirements analysis**:
   - Inventory applicable regulations and standards (like General Data Protection Regulation (GDPR), HIPAA, Payment Card Industry Data Security Standard (PCI-DSS), Service Organization Control 2 (SOC 2), and International Organization for Standardization 27001 (ISO 27001)).
   - Build a compliance baseline at an organizational and Region level. Allow teams to extend baselines by adding requirements unique to their workloads.
   - Map baseline requirements to AWS services and shared responsibility boundaries.
   - Identify gaps between current state and regulatory requirements.
   - Consider report generation requirements and evidences needed to support audit exercises.
3. **Evaluate and integrate compliance tooling**:

- Select services and tools aligned with your compliance requirements. For more detail, see Choosing AWS security, identity, and governance services.
- Develop prototypes and architecture outlines to gain a holistic understanding of how your selected tools work. Understand how they integrate with and support your compliance workflow.
- Assess and configure integrations requirements with your existing architecture. AWS security and compliance services are integrated from the outset. However, additional effort may be needed to integrate third-party tooling. This includes IT service management (ITSM), cloud security posture management (CSPM), security information and event management (SIEM), endpoint detection and response (EDR), and extended detection and response (XDR) tools.
- Aim to operationalize centralized compliance dashboards. Consider adopting a CSPM product (for example, AWS Security Hub CSPM), augmented with a product that provides data security posture management (DSPM) capabilities.

4. **Establish engineering best practices**:
   - Build and test automated compliance controls. Write AWS CloudFormation Guard rules to provision custom controls. Use the CloudFormation CLI to validate and unit test your guard rules.
   - Build and test automated remediations. Systems Manager Automation runbooks provide a large catalog of remediations ready for use.
   - Monitor your compliance status. Many AWS security and compliance services provide built-in customizable dashboards. For example, AWS Security Hub allows you to create custom insights to track issues relevant to your workload.
   - Enable tools to automate audits. Consider adopting AWS Audit Manager. It continually audits your AWS usage to simplify risk and compliance management. When you enable an Audit Manager standard framework it automatically collects logs, and security findings on your behalf.
5. **Start shortlisting data sources for compliance monitoring**: AWS Config can directly detect compliance drifts as a result of resource misconfigurations (reported as findings). However, additional effort is required to detect non-compliance due to operational loopholes or code and design flaws. Logs play a central role in this process. Candidate sources include network flow logs, access logs, service and application usage logs, plus security events, and security findings. The shortlisted compliance tooling should automatically ingest data from multiple data sources and derive insights with minimal custom integration code.
6. **Set up service onboarding procedures**: Set best practices and enforce guardrails to onboard new AWS Services in line with your security and compliance best practices. Start with security best practices provided with AWS service documentation. AWS Artifact contains additional

information regarding the compliance status of individual AWS services. AWS Control Tower provides more than 700 guardrails in the form of preventative, proactive, and detective controls to set up consistent and secure cloud environments across AWS Regions and accounts.

7. **Set up compliance training programs**:
   - Launch training courses to address upskilling needs, especially around new tools and techniques.
   - Launch role-based training courses. Consider starting with the following topics:
     - Understanding data classification (for example, public, internal, confidential).
     - Best practices of handling sensitive data.
     - Understanding regulatory requirements and standards (for example, GDPR, HIPAA, PCI-DSS, California Consumer Privacy Act (CCPA)).
   - Create mentorship, and advocacy programs to drive awareness.
8. **Develop response procedures**:
   - Design and develop runbooks for compliance violations and security incidents. Consider using AWS Systems Manager Incident Manager for response coordination.
   - Establish evidence preservation and extraction procedures to respond to regulatory inquiries. This includes data subject requests, data disclosure requests from law enforcement agencies, or reporting incidents of data breach. Consider building a cyber forensics capability over time to deal with such requirements in a repeatable and predictable manner.
9. **Set objective success criteria**: For example,
   - Reduce the total number of resources not meeting compliance requirements, or reduce *critical* and *high* issues.
   - Reduce the time between remediation and the first detection of a compliance issue.
   - Reduce effort by decreasing the number of days spent supporting audits.
   - When you enable Security Hub, it automatically generates a dashboard that provides a summary of findings by severity. You can drill-down to individual findings and extract the data required to produce some of the statistics listed above. It also possible to build automations to extract this data. For example, the Security Hub Compliance Analyzer extracts findings to an Amazon S3 Bucket, and then parses out the relevant information.
10. **Implement continuous improvements**:
    - Monitor health of key compliance metrics.
    - Identify optimization opportunities. Gather feedback and implement improvements.
    - Improve control testing procedures.
    - Improve automation runbooks and their coverage.

Resources

Related best practices:

- [AG.ACG.1] Adopt a risk-based compliance framework
- [AG.ACG.2] Implement controlled procedures for introducing new services and features
- [AG.ACG.3] Automate deployment of detective controls
- [AG.ACG.4] Strengthen security posture with ubiquitous preventative guardrails
- [AG.ACG.6] Implement auto-remediation for non-compliant findings
- [O.DIP.1] Aggregate logs and events across workloads
- [O.DIP.2] Centralize logs for enhanced security investigations
- OPS01-BP03 Evaluate governance requirements
- OPS01-BP04 Evaluate compliance requirements

Related documents:

- Scaling a governance, risk, and compliance program for the cloud, emerging technologies, and innovation
- Evolving GRC to Maximize Your Business Benefits from the Cloud
- Optimizing cloud governance on AWS: Integrating the NIST Cybersecurity Framework, AWS Cloud Adoption Framework, and AWS Well-Architected
- Decision Guide: Choosing AWS security, identity, and governance services
- Introducing AWS Audit Manager Common Controls Library
- AWS Security Incident Response Technical Guide
- Training programs available through AWS Skill Builder, and AWS Workshops.
- Building Security from the Ground up with Secure by Design

DSOPS01-BP02 Delegate compliance responsibilities

Effective compliance at scale requires distributing responsibilities across teams while maintaining centralized oversight and standards. Traditional centralized compliance models create bottlenecks that slow business velocity and limit an organization's ability to respond quickly to regulatory changes.

By delegating compliance responsibilities to individual teams while providing them with common guardrails and automated tools, organizations can achieve both compliance consistency and operational agility.

Desired outcome: Teams inherit common organizational controls but are also empowered to address regulatory requirements independently.

Common anti-patterns:

- Organizations entrust responsibility and accountability of addressing compliance requirements to a handful of experts only.
- The belief that centralizing compliance functions leads to more consistent and efficient implementations.
- Teams view compliance as a post-deployment concern.

Benefits of establishing this best practice:

- Every team is responsible, capable, and adequately empowered to detect and remediate compliance issues independently.
- Teams inherit common guardrails while being empowered to add or customize workload-specific guardrails.
- Reduced compliance bottlenecks through distributed decision-making authority.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Compliance is a shared responsibility and is scaled through empowerment. This is a five step process. It involves deploying a suite of integrated compliance and security tooling, defining key roles and responsibilities, setting up guardrails, delegating responsibilities, and supporting continuous training.

Implementation steps

1. **Deploy foundational tools**: Start by setting up foundational security and compliance tooling to create a robust compliance governance framework from day one. The decision guide Choosing AWS security, identity, and governance services assists customers in selecting services that best fit their needs. For more detail, see AWS Security Reference Architecture (AWS SRA). The AWS SRA provides detailed guidelines for deploying a full complement of AWS security services in a multi-account environment. It recommends setting up a dedicated Security Tooling Account to co-locate your security tooling at one place.
2. **Define roles and responsibilities**: Having a set of clearly defined roles is essential to delegating compliance responsibilities at scale. Here are a few example roles:

   - Central cloud team roles. Responsibilities may include:
     - Defining organization-wide compliance policies (for example, encryption or logging).
     - Deploying common guardrails and provisioning services to individual member accounts.
     - Monitoring compliance centrally (using for example, AWS Security Hub Cloud Security Posture Management).
   - Account team roles. Responsibilities may include:
     - Implementing account-specific controls (for example, IAM policies or resource tagging).
     - Remediating resources that don't meet compliance requirements.
     - Reporting compliance status to the central team.
3. **Establish guardrails**: Guardrails provide developers with an approved method of self-provisioning and self-managing resources and environments, but within pre-set limits. For example, guardrails allow you to pre-set the type, size, capacity, and the environment within which a resource can be provisioned. Using AWS Control Tower, you can activate common guardrails from your management account and attach those guardrails widely and consistently to member accounts. Here are some of the ways you set those guardrails:
   - **Service control policies (SCP):** By implementing SCPs, organizations can set preventative guardrails at an Organizational Unit (OU) level, at the level of an Account, or both. SCPs are written with the same syntax used to define IAM policies. They apply by inheritance, meaning that accounts under an OU automatically inherit the same guardrails as set for that OU. Each OU inherits the same guardrails as set in the root organizational account. SCPs do not grant permissions to the IAM users and IAM roles. An SCP defines a permission guardrail, or sets limits, on the actions that the IAM users and IAM roles in your organization can perform. Common use cases include:
     - Restrict access to unapproved AWS Regions.
     - Block deletion of compliance-related resources.
     - Enforce encryption requirements.
     - Block unauthorized API calls.
     - Mandate resource tagging.
   - **Resource control policies (RCP):** While SCPs control which AWS services and actions can be used, RCPs focus on controlling the configuration of specific resources and resource types. RCPs are applied at the organization root, organizational unit (OU), or at the account level, allowing for granular control over resource configurations. Common use cases of RCP include:
     - Restrict access to only HTTPS connections to your resources.
     - Consistent Amazon S3 bucket policy controls (for example, enforce KMS Encryption).
   - **AWS CloudFormation guard rules:** AWS CloudFormation guard rules are an effective way to establish what a well-configured resource looks like and how it should behave. Develop guard

rules and use cfn-validate and cfn-test to verify CloudFormation templates locally as well with your CI/CD pipelines.

- **AWS Firewall Manager policies:** Use AWS Firewall Manager to administer and maintain ingress and egress controls across multiple accounts and resources.
- **Additional network controls:** Consider AWS Web Application Firewall (AWS WAF), AWS Shield Advanced, Amazon Virtual Private Cloud (VPC) security groups, network ACLs, AWS Network Firewall, and Amazon Route 53 Resolver DNS Firewall to control traffic across your environments.
- **Backup and retention policies:** Use AWS Backup to set up organization-wide backup and retention policies aligning with your regulatory requirements.
- **Tagging policies:** Tags play an important role in defining the context under which a resource operates. By enforcing tags, you can apply selective behavioral controls without interrupting feature delivery. For more detail, see Best Practices for Tagging AWS Resources.

4. **Delegate responsibilities**: While SCPs, RCPs, and firewall policies apply consistent guardrails, they are effectively a form of denylisting. To empower teams, we must also think of allowlisting. To do this, set up delegated admin roles by developing IAM policies and setting up IAM permission boundaries:
   - Delegated admin roles enable organizations to distribute governance responsibilities while maintaining centralized oversight and compliance standards. This allows security, compliance, and infrastructure teams to independently manage their respective domains. For example, security teams can manage Amazon GuardDuty findings across the organization, while compliance teams can monitor Config rules and Security Hub controls. AWS Services integrated with AWS Organizations support delegated administration roles.
   - Identity and access management (IAM) policies and resource-based policies provide the foundation for delegation by establishing granular access controls and automated policy enforcement. As a strategy, consider setting up broad allow lists in development and test environments, and then scoping them down as you move up to production environments. You can use AWS IAM Access Analyzer to observe permission usage patterns across development and test environments. By monitoring actual access patterns and generating policy recommendations based on observed usage, you can progressively scope down permissions without impacting functionality. To do this:
     - Enable IAM Access Analyzer across environments.
     - Review findings regularly as part of the code promotion process.
     - Integrate analyzer recommendations into your CI/CD pipeline and progressively reduce permissions.
     - Document justifications for deliberately broad permissions.

    - Use the generated policies as templates for production roles.
  - IAM permission boundaries act as a guardrail to define the maximum permissions an IAM entity (user or role) can have, regardless of the permissions granted by their identity-based policies. Think of them as a security fence that blocks privilege escalation by setting an upper limit on what actions can be performed. Permission boundaries don't grant permissions themselves; instead, they work in conjunction with identity-based policies where the effective permissions become the intersection of what's allowed by both the identity-based policy and the permission boundary. This provides defense-in-depth and enables secure delegation of IAM management tasks. Common use cases include:
    - Blocking IAM privilege escalation.
    - Limiting developer permissions in production or other accounts holding sensitive data. For example, you can create your own permission boundaries.

5. **Train teams and document processes**: Prepare teams to become self-sufficient by providing the necessary knowledge and tools required to become successful.
   - Train teams on compliance tools.
   - Provide on-demand sandbox environments for teams to gain hands-on experience in building simple instruction driven solutions. Consider subscribing to immersive learning offerings from AWS Skill Builder such as AWS Cloud Quest, SimuLearn and Industry Quest.
   - Publish compliance playbooks with step-by-step guidance around technical and operational capabilities required to support existing and emerging regulations. Allow teams to find playbooks on common challenges such as securing personally identifiable information (PII) and financial data.

Example workflow: The following is an example of a workflow you can expect after effective delegation of responsibilities.

1. A member account deploys an unencrypted S3 bucket.
2. AWS Config rule s3-bucket-server-side-encryption-enabled triggers a non-compliance alert.
3. AWS Security Hub aggregates the finding and notifies the account owner using Amazon Simple Notification Service (Amazon SNS).
4. The account team uses a pre-approved AWS Systems Manager runbook to enable encryption automatically.
5. Compliance status updates in the central dashboard.

By combining automation, delegation, and education, this model scales seamlessly across your AWS organization.

Resources

Related best practices:

- [AG.SAD.2] Delegate identity and access management responsibilities
- OPS03-BP02 Team members are empowered to take action when outcomes are at risk
- OPS03-BP03 Escalation is encouraged
- OPS03-BP04 Communications are timely, clear, and actionable
- OPS03-BP06 Team members are encouraged to maintain and grow their skill sets

Related documents:

- Two-Pizza Teams Are Just the Start, Part 1: Accountability and Empowerment Are Key to High-Performing Agile Organizations
- Two-Pizza Teams Are Just the Start, Part 2: Accountability and Empowerment Are Key to High-Performing Agile Organizations
- Compliance validation for AWS Organizations
- Delegating responsibility to others using permissions boundaries
- Setting a delegated administrator account in Security Hub
- Managing GuardDuty accounts with AWS Organizations
- Managing multiple Macie accounts with AWS Organizations
- Using Organizations to manage behavior graph accounts
- Delegated administrator for IAM Access Analyzer
- Designating a delegated administrator account for Amazon Inspector
- Adding a delegated administrator in AWS Audit Manager

Related videos:

- AWS re:Inforce 2022 - Deep dive into compliance and auditing at scale (GRC402)

Related examples:

- When and where to use IAM permissions boundaries

DSOPS01-BP03 Implement compliance training and awareness

Compliance-trained software professionals can design and build systems that adhere to regulations (for example, GDPR, HIPAA, and PCI-DSS). Without awareness, even well-intentioned technical decisions can lead to vulnerabilities or solutions that don't meet regulatory requirements, exposing organizations to potential risk.

Desired outcome: Software professionals are equipped with knowledge and skills required to consistently design, develop, and maintain systems that meet compliance requirements.

Common anti-patterns:

- Generic compliance training without context or applicability.
- Focusing on compliance training completion rates rather than effectiveness.
- Failing to update training and guidance when regulations change.
- Creating siloed compliance knowledge within specialized teams.
- Lack of opportunities to practice compliance measures in a hands-on setting, making it harder to apply design principles and best practices in real-world scenarios.

Benefits of establishing this best practice:

- Improved security posture across systems and services.
- Reduced need for late-stage remediation or redesigns.
- Reduced risk of violations and associated penalties.
- Fosters accountability for security and privacy by default.
- Increased customer trust through demonstrable regulatory adherence and security practices.
- More efficient development cycles by addressing compliance requirements early.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Develop a structured, role-specific program combining foundational training, hands-on practice, and continuous reinforcement. Align content with organizational risk profiles, integrate compliance into development workflows, and use automated tooling to address regulatory needs in real-world scenarios.

Implementation steps

1. **Assess**: Baseline your learning needs.
   - Conduct surveys to determine current understanding of regulatory requirements across teams. Consult with subject matter experts to identify knowledge and skills gaps.
   - Identify the regulatory standards that apply to your organization.
     - Use existing documentation such as business impact analysis (BIA), data protection impact analysis (DPIA) documents, and risk registers.
     - Use existing policy documentation such as security, data handling, privacy, data classification, and data retention policies.
     - Get read-only access to production or production-like environments to study existing compliance related controls.
   - Map requirements to specific technical roles and responsibilities.
2. **Define success criteria**: Training programs must lead to successful outcomes (for example, a 25-50% reduction in compliance violations within six months or a 20-30% improvement in developer confidence when dealing with specific compliance standards).
3. **Develop content**: Create training content and reusable assets that cover identified regulatory standards and address skills and knowledge gaps.
   - Create role-specific learning paths for targeting specific technical and operational roles.
   - Focus on secure architecture and design patterns, augmented with data protection and data privacy-related best practices.
   - Cover secure coding standards, safe data handling, and incident response procedures.
   - Cover reporting procedures (for example, security incident reporting, available escalation paths, key points of contact, and whistleblower helplines).
   - Cover AWS services and independent software vendor (ISV) products related to compliance and security.
   - Create searchable knowledge bases of compliance requirements. Consider incorporating LLM-powered search augmented with Amazon Bedrock Knowledge Bases and AWS MCP Servers.
   - Create a library of reusable reference architectures, prescriptive guidance, solution accelerators, and code examples that developers can readily apply. AWS Architecture Center and AWS Samples provide a starting point.
4. **Deliver**: Provide regular training sessions using a combination of talks, discussions, and hands-on exercises.
   - Deploy interactive e-learning modules to deliver training content.
   - Conduct hands-on workshops for practical application of skills (for example, breach response drills).
   - Arrange expert-led sessions for complex topics.

5. **Apply**: Incorporate compliance considerations into existing design and development processes.
   - Deploy compliance and vulnerability checking linters and IDE plugins.
   - Apply a compliance checklist when evaluating architecture decisions and during code reviews.
   - Incorporate automated compliance validation into Continuous Integration and Continuous Deployment (CI/CD) pipelines.
   - Establish office hours with subject matter experts.
6. **Track, refine, and reinforce**: Regularly monitor and evaluate the effectiveness of training programs.
   - Track violations in code reviews and security testing.
   - Track audit outcomes and incident trends to refine training content.
   - Measure effectiveness through knowledge assessments.
   - Share updates on evolving standards through newsletters or other team messaging channels.
   - Reinforce training content through code reviews and security testing.
7. **Scale up**: Once you achieve your initial success metrics, scale the program.
   - Deploy learning management systems (LMSs) across the organization.
   - Drive professional certification and accreditation programs. Consider providing subscriptions to online learning solutions to enable your colleagues to prepare.
   - Consider setting up moderated team channels so that teams can discuss and share best practices and challenges.

Resources

Related best practices:

- OPS03-BP06 Team members are encouraged to maintain and grow their skill sets
- OPS11-BP04 Perform knowledge management
- SEC11-BP01 Train for application security

Related resources:

- Comprehensive resource for AWS compliance offerings
- Get started on security training with content built by AWS experts
- Automated compliance checks against industry standards
- Policy-as-code evaluation tool
- Resource compliance monitoring and remediation
- Continuous audit evidence collection

- Compliance-focused reference deployments
- Set up and govern compliant multi-account environments

DSOPS01-BP04 Establish vendor compliance management

Vendor compliance management is essential for maintaining security and regulatory adherence across your supply chain.

As organizations increasingly rely on third-party vendors and services, establishing partner compliance becomes essential. These partners must meet the same cybersecurity standards, data privacy requirements, and industry-specific regulations for effective risk mitigation.

Without proper vendor compliance management, organizations can expose themselves to significant security vulnerabilities, regulatory violations, and potential data breaches through their extended environment.

Desired outcome: A systematic, documented, and continuously-monitored vendor compliance program is established for third-party services. The program verifies that services integrated with your AWS environment meet or exceed your organization's security and regulatory requirements.

Common anti-patterns:

- Evaluating vendors just during onboarding without ongoing monitoring.
- Using the same assessment approach for vendors regardless of risk level or data access.
- Failing to include specific compliance requirements in vendor agreements.
- Not maintaining evidence of vendor compliance for audit purposes.
- Relying on spreadsheets and email for managing vendor compliance.
- Not establishing protocols for addressing vendor compliance failures.

Benefits of establishing this best practice:

- Comprehensive visibility into vendor security postures minimizes the likelihood of breaches through third-party connections.
- A structured approach to vendor management simplifies evidence collection during audits and regulatory assessments.
- Demonstrates to regulators and customers that proper oversight of the entire supply chain is in place.

- Identifying vendor compliance gaps early protects against service disruptions.
- Risk-based vendor classification enables focused attention on the most critical third-party relationships.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Establishing vendor compliance management requires a systematic approach to identifying, assessing, and monitoring third-party risks within your AWS environment. This process should be proportionate to the risks involved, with greater scrutiny applied to vendors that access sensitive data or provide critical services.

Implementation steps

1. **Create a vendor inventory**:
   - Document third-party services connected to your AWS environment.
   - Classify vendors based on data sensitivity and operational criticality.
   - Map vendors to applicable compliance frameworks (for example, HIPAA or GDPR).
2. **Design assessment frameworks**:
   - Develop tiered questionnaires based on vendor risk levels.
   - Create technical assessment procedures (penetration tests, architecture reviews).
   - Establish minimum acceptable compliance standards for each vendor and enforce these standards through contractual obligations.
3. **Implement assessment processes**:
   - Conduct pre-engagement assessments for new vendors.
   - Perform regular reassessments based on risk classification.
   - Verify vendor certifications and attestations (for example, SOC 2 or ISO 27001).
4. **Build contractual protections**:
   - Include specific compliance requirements in vendor agreements.
   - Define data handling and security obligations.
   - Establish right-to-audit provisions and breach notification requirements.
5. **Deploy continuous monitoring**:
   - Implement automated compliance monitoring for third-party integrations. For example, AWS Security Hub CSPM can receive findings from third-party partner product integrations. This provides real-time visibility and reduces the need to collect and maintain documentary evidence from third parties.

    - Set up alerts for compliance deviations.

6. **Establish governance structures**:
    - Define clear roles and responsibilities for vendor oversight.
    - Create a vendor compliance review board for high-risk providers.
    - Develop escalation paths for compliance violations.
7. **Document and report**:
    - Maintain centralized records of vendor assessments.
    - Create compliance dashboards for executive visibility.
    - Prepare regular status reports for stakeholders.

Resources

Related best practices:

- Permissions management best practices
- Data classification best practices
- Incident response requirements and procedures

Related documents:

- AWS Audit Manager: Integrations with third-party Governance, Risk and Compliance (GRC) products

Related videos:

- AWS re:Inforce 2022 - Simplify the vendor risk assessment process with Vendor Insights
- AWS re:Invent 2025 - Scale Security Operations with AWS Security Incident Response Service (SEC329)
- AWS Security Incident Response Investigative Agent Demo

Prepare

DSOPS02: How do you design workloads for compliance?

Designing compliant workloads is a critical consideration for AWS customers, especially in an era where regulatory landscapes are increasingly complex and stringent.

Considerations:

- Non-compliance can lead to severe financial penalties, legal repercussions, and reputational damage, which can have long-lasting effects on a business.
- With the ever-evolving nature of cybersecurity threats and data privacy concerns, regulatory adherence is not a one-time task but an ongoing process that requires constant vigilance and adaptability.
- As businesses increasingly adopt cloud services, the shared responsibility model between AWS and its customers necessitates a clear understanding of where compliance obligations lie.
- Finally, rather than treating compliance as an afterthought or a periodic checkpoint, organizations should consider building compliance into their workload architectures from the start.

Therefore, understanding how to design compliant workloads is essential for AWS customers aiming to navigate the complexities of modern regulatory environments with cloud computing.

DSOPS03: How do you design your workload for continuous auditability?

In highly regulated industries, continuous auditability enables real-time compliance visibility, reduces risk of breaches or non-compliance penalties, and builds trust with regulators. It shifts compliance from a reactive, manual process to an automated, proactive practice embedded in the workload's design.

Best practices

- DSOPS02-BP01 Baseline your compliance requirements
- DSOPS02-BP02 Establish an automated path to compliance
- DSOPS02-BP03 Develop and track key compliance metrics
- DSOPS03-BP01 Plan and prepare for audits
- DSOPS03-BP02 Automate evidence collection and reporting

DSOPS02-BP01 Baseline your compliance requirements

Baselining your compliance requirements is crucial for making sure that your organization maintains a consistent and secure posture. It assists in proactively identifying gaps, reducing risks, and maintaining adherence to regulatory standards.

Desired outcome: Your organization has a clear, authoritative understanding of compliance requirements that guides design decisions, accelerates workload deployment, and reduces risk across AWS environments.

Common anti-patterns:

- Making decisions based on informal understanding rather than documented requirements.
- Applying the same compliance controls across workloads without considering specific regulatory requirements.
- Only identifying requirements after a compliance issue or audit finding occurs.
- Creating compliance baselines once and rarely updating them as regulations evolve.

Benefits of establishing this best practice:

- Risk reduction through proactive identification and mitigation of compliance gaps.
- Cost optimization by implementing appropriate controls without over-engineering solutions.
- Faster deployment of compliant workloads using pre-approved architectural patterns.
- Improved audit readiness with clear documentation and evidence of compliance consideration.
- Enhanced stakeholder confidence through demonstrated commitment to regulatory adherence.
- Streamlined decision-making with clear compliance criteria for design choices.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Begin by conducting a comprehensive assessment of your organization's compliance landscape, including industry-specific regulations, jurisdictional requirements, and internal policies. Engage legal, compliance, and business stakeholders to holistically cover applicable requirements. Document each requirement with sufficient detail to enable technical implementation decisions, including specific controls, evidence requirements, and assessment criteria.

Implementation steps

1. **Define a compliance matrix**: Create a compliance matrix, mapping regulatory requirements to specific technical controls and processes. As examples, refer to the Secure Controls Framework (SCF) and download the latest secure-controls-framework-scf-[version].xlsx file. Use the AWS Customer Compliance Guides (CCGs) and the attached excel file in the document to find a full complement of AWS built controls categorized by services, compliance standards, and security topics. The two resources could serve as a potential starting point to map named compliance frameworks to equivalent preventative, proactive, and detective controls listed under AWS Control Tower, controls managed by AWS Security Hub CSPM and managed rules provided by AWS Config.

   The following table is an example of a compliance mapping matrix. The first four columns state the *organization control id*, *framework control id*, *category* and *subcategory*. The *policy directive* column translates a framework requirement (known as Complementary Customer Criteria in C5 terminology) to instructions describing how this requirement should be met when using a specific technology capability. The technology capability in this example is object storage. Relational database, message queue, volume storage, and network attached storage are few of the other examples. The *candidate controls* column list the technical controls (expressed as policy as code) required to meet the policy directive. The control mapping shown below is only for illustration purposes. It is not accurate and is just one among several possible interpretations. The *validation rules* column informs developers on the pass criteria.

   For illustration, the example uses the Cloud Computing Compliance Criteria Catalogue – C5 published in 2020 by the Federal Office for Information Security (BSI) of Germany. C5 serves as a foundation in the area of cloud security for providers, customers, and auditors. In the example shown here, a compliance requirement is named CRY-03. For full text refer to Section 5.8, CRY-03 of Criteria Catalogue C5:2020.

Organization Control Id	C5-Control Id	Category	Subcategory	Policy Directive	Candidate Controls	Validation Rules
ENC-001-R	CRY-03 Encryption of sensitive data for storage	Encryption at rest	Object storage	**(1)** Do not configure object storage solutions with default cloud provider managed encryption keys. **(2)** Generate key material outside of the cloud provider's environment using an independent toolchain. Use the cloud provider's key management tools and the key	CT.S3.PR.10: Require an Amazon S3 bucket to have server-side encryption configured using an AWS KMS key. CT.S3.PV.6: Require object uploads to Amazon S3 buckets to use server-side encryption with an AWS KMS key (SSE-KMS) .SH.S3.17: S3 buckets should be encrypted	For up to restricted data sensitivity levels, the following rules apply: **(1)** SSE-KMS is enabled as default. **(2)** KMS key used was built using imported key material. **(3)** Users pass the x-amz-server-side-encryption-aws-kms-key-id header element during s3:PutObject operation

Organization Control Id	C5-Control Id	Category	Subcategory	Policy Directive	Candidate Controls	Validation Rules
				material you built, to generate new keys. Save the generated keys to the cloud provider's key vault. Validate that the key vault is at least FIPS 140-3 Level 3 compliant. Also use secure channels during key exchanges. **(3)** Consider using dedicated HSMs and external key stores for data classified	at rest with AWS KMS keys. Add more controls	s for buckets which are not configured with SSE-KMS. For confidential and highly-confidential data sensitivity levels, follow separate guidance on provided with ENC-001-CH

Organization Control Id	C5-Control Id	Category	Subcategory	Policy Directive	Candidate Controls	Validation Rules
				confidential or above. This allows you to store keys outside of cloud provider's key vault and specify where the encryption/decryption operations are performed.		

- CT.S3.PR.10 is a proactive control implemented in the form of an AWS CloudFormation guard rule. It acts as a check-point during resource provisioning and is invoked when you use CloudFormation templates or the CloudFormation CDK to provision S3 buckets. However, if your aim is to block non-compliant S3 buckets from being provisioned when using other infrastructure as code (IaC) tools, this control alone would not be sufficient. To achieve a similar outcome, first build a CloudFormation hook and add the guard rule specification described as part of CT.S3.PR.10. Then provision your S3 bucket using an IaC provider such as Terraform, or Pulumi that supports the AWS Cloud Control API (CCAPI). For more detail, see Introducing AWS CloudFormation Hooks invoked via AWS Cloud Control API (CCAPI).

- CT.S3.PV.6 is a resource control policy (RCP). It requires users to supply a KMS key ID as part of the header in S3:PutObject requests for buckets that do not have SSE-KMS configured as the default encryption mode. SH.S3.17 is an AWS Config managed rule that checks whether an S3 bucket is encrypted with an KMS key (SSE-KMS or DSSE-KMS). All three controls can be enabled at an Organization Unit (OU) or an account level using AWS Control Tower.

Developing and keeping compliance mapping matrix documents and databases up to date requires input from more than a one team. In the example shown here, security and compliance SMEs may be providing inputs for the organization control id, framework control id, category, subcategory, and policy directives columns while SecOps and DevSecOps teams may be responsible for filling up the candidate controls and validation rules columns.

2. **Document jurisdiction specific requirements**: Document jurisdiction specific (for example, regional union, country, trading bloc, province) cybersecurity, data privacy and data export controls requirements. Incorporate these additional requirements into your compliance matrix.
   - Document guidelines around data export controls. This includes bi-lateral or multi-lateral adequacy arrangements you may need to adhere to.
   - Document legal requirements for workforce location and citizenship-related restrictions. Operational roles in domains such as defence, military and law enforcement may require additional vetting.
   - Document cryptographic standards and policies.
   - Document mandatory public disclosure requirements. For example, tracking how many data disclosure requests you received from law enforcement agencies, and how many did you fulfil.
   - Document data breach reporting procedures. This should include what to report, whom to report to and within what timelines.
   - Document intellectual property rights requirements.
   - Validate requirements through reviews with local legal experts and your own compliance teams. Seek clarifications from regulators and designated national competent authorities.
   - Additionally, consider documenting the potential economic impact of meeting regulations (cost of regulation). This guides yearly planning and budgeting.
3. **Build a comprehensive knowledge base:** Make compliance mappings searchable, accessible, and understandable to improve awareness and reusability.
   - Include descriptive text and guidance.
   - Make prior certifications and attestations readily available to the wider organization.
   - Use advanced analytics methods. For example, large language models (LLMs), retrieval-augmented generation (RAG) search, and knowledge graphs to make compliance-related literature more accessible.

4. **Conduct gap analysis**: Discover potential gaps in your compliance posture.
   - Conduct data protection impact analysis (DPIAs) where there is a high risk of non-compliance due to the location and sensitivity of the data involved.
   - A quick way to spot compliance issues would be to enable AWS Security Hub CSPM. Then, enable a Security Hub standard to automatically start collecting data about non-compliant resources. Compliance findings are displayed on the Security Hub CSPM dashboard.
5. **Log and manage risks**: Maintain a risk register. Log and manage compliance-related risks that cannot be addressed by the current design. Document compensatory measures applied to partially or fully mitigate risks.
6. **Review compliance baselines**: Update baselines periodically to address emerging threats and regulatory changes. Re-assess compliance requirements per jurisdiction and industry. Stay tuned to updates coming from regulatory authorities.

Resources

Related best practices:

- OPS01-BP03 Evaluate governance requirements
- OPS01-BP04 Evaluate compliance requirements
- SEC02-BP01: Use strong identity foundation

Related documents:

- Implementing a compliance and reporting strategy for NIST SP 800-53 Rev. 5
- Scaling a governance, risk, and compliance program for the cloud
- AWS Security Reference Architecture (SRA)
- AWS Risk and Compliance Whitepaper
- Exploring the new AWS European Sovereign Cloud: Sovereign Reference Framework
- Architecting for HIPAA Security and Compliance on Amazon Web Services
- AWS Config Conformance Packs
- AWS Security Hub Compliance Standards
- AWS Artifact User Guide
- AWS Audit Manager User Guide

Related videos:

- Global Security & Compliance Acceleration Program
- Global Security & Compliance Acceleration (GSCA) Bundles
- AWS re:Inforce 2022 - Quantifying your compliance posture with conformance packs (GRC211)

DSOPS02-BP02 Establish an automated path to compliance

Establish an automated path to effectively meet cybersecurity standards, data privacy legislation, and industry-specific regulations like HIPAA and PCI-DSS. Automation reduces the risk of human error, speeds up the compliance process, and allows for continuous monitoring and adaptation to changing regulations.

Desired outcome: Compliance is built into every change through automated validation, detection, and remediation, enabling faster deployments and audit-ready documentation.

Common anti-patterns:

- Relying on periodic manual audits and spreadsheet-based tracking instead of continuous automated monitoring.
- Performing compliance validation only during audit periods rather than continuously.
- Failing to use a centralized tool for compliance reporting resulting in blind spots.
- Generating compliance alerts without automated remediation or clear escalation procedures.

Benefits of establishing this best practice:

- Reduce human error in compliance monitoring, detection, analysis, and remediation.
- Maintain real-time regulatory posture with automated evidence collection and reporting, enabling audit-ready responses and regulatory confidence.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

To establish an automated path to compliance, codify your compliance requirements into compliance as code (CaC) policies. Integrate compliance validation into your CI/CD pipelines, and build compliance in to every change.

The most important components of this strategy are:

1. Defining compliance policies as code
2. Unit testing and validating compliance policies
3. Integrating compliance policies with CI/CD pipelines
4. Creating automated remediation workflows to address common compliance violations
5. Regularly testing compliance as code policies through controlled violations and performing automated remediation actions in sandbox, development or test environments

Implementation steps

AWS provides several capabilities required to establish a mature compliance posture. Consider using the three lines of defense model developed by Institute of Internal Auditors (IIA) to select and provision AWS services that best fit your needs.

In the three lines model, the first-line function manages risk, the second-line function oversees risk, and the third-line function provides objective and independent assurance of risk management. Aligning with this model, consider the following:

1. **First line: risk management:** Manage risks by applying secure by design (SBD) principles, provisioning automated guardrails, and implementing automated remediations.
   - Start by defining compliance metrics (including deviations, thresholds, and risk tolerance levels) working together with your security consultants, data protection office, and workload owners. Metrics guide decisions on automation priorities and tool selection.
   - Apply a set of controls expressed as compliance as code and aligned to best practices and compliance frameworks. Examples include AWS Foundational Security Best Practices and NIST SP 800-53 Rev. 5. AWS Control Tower provides 700 plus preventative, proactive, and detective controls mapped to several frameworks. When you enable Control Tower controls, it automatically starts enforcing compliance as code policies and also detects compliance drifts.
   - Adopt a multi-account strategy similar to the guidance provided in the AWS Security Reference Architecture. With account-level isolation, you can better control the impact of potential compliance issues.
   - Provide pre-approved, compliance-aligned infrastructure templates (for example, VPCs or EC2 instances) to enable developers to provision standardized resources. Consider using Service Catalog to enable provisioning, administration, and management of standardized AWS CloudFormation or Terraform Cloud products.
   - Reduce potential information exposure risks by applying principle of least privilege and just-in-time access.

- Provide automated runbooks to remediate compliance violations. The AWS Systems Manager Automation Runbook Reference provides a catalog of runbooks. You can also create your own runbooks.

2. **Second line: risk oversight:** Aim to identify compliance drifts on a continuous basis and prioritize remediations based on calculated risk scores. Use AWS Security Hub CSPM or an equivalent Cloud Security Posture Management (CSPM) solution to gain continuous risk oversight.
   - Enable Security Hub CSPM to accept findings from AWS services and third-party providers.
   - Enable consolidated controls view and consolidated controls findings in Security Hub CSPM. This reduces findings noise by producing a single finding for a control, even if the control applies to multiple enabled standards.
   - Log API activity with AWS CloudTrail across accounts and store logs in a central S3 bucket. Log network flows, especially logs generated at the edge of the network. Require applications to log user actions. For example, user X changed document Y at time Z. When you log user actions (commands) and correlate them with system generated logs (for example firewall logs), you gain a holistic understanding of activity in your workload.
   - Enable encryption and log integrity validation. If you are using Amazon Security Lake as a long-term storage for your security and compliance logs, apply these measures to protect data. Consider enforcing the write once read many (WORM) model when storing logs in S3 to protect chain of evidence.
3. **Third line: risk assessment and compliance reporting:** Aim to support continuous risk assessment of your entire environment by automatically collecting, aggregating, and analyzing logs.
   - Source logs audit logs, network flow logs, firewall logs and application. Example sources include:
     - Amazon CloudTrail Data and Management Event Logs for API activity tracking
     - VPC Flow Logs for network traffic analysis
     - AWS WAF Logs for web application firewall monitoring
     - Amazon EKS Audit Logs for Kubernetes cluster activity
     - Route 53 resolver query logs for DNS query monitoring
     - Application-specific logs collected through Amazon CloudWatch or another application performance monitoring (APM) tool
   - Consolidate logs using Amazon Security Lake, or an equivalent security-focused data lake to enable efficient querying and analysis across terabytes of log data. Security Lake normalizes log files to a single open format known as the Open Cybersecurity Schema Framework

(OCSF). This standardization allows you to connect Security Lake with your existing security information and event management (SIEM) tooling as well as Amazon OpenSearch Service.

- Generate audit-ready reports on-demand with AWS Audit Manager. Audit Manager streamlines compliance reporting by collecting evidence aligned with several frameworks such as NIST 800-53 Rev 5 and PCI DSS 4.0. It ingests results from AWS Config managed rule evaluations, CloudTrail management event logs, findings from Security Hub and can also make AWS API calls to generate snapshots of your environment. Audit Manager can generate assessment reports based on assessments you create.
- Log in into your AWS Management Console and use the AWS Artifact service to access AWS security and compliance reports plus select online agreements. You can download AWS compliance reports like SOC, ISO, and PCI directly to demonstrate AWS infrastructure compliance to auditors.

Resources

Related best practices:

- OPS05-BP01 Use version control
- OPS05-BP02 Test and validate changes
- OPS05-BP03 Use configuration management systems
- OPS05-BP04 Use build and deployment management systems
- OPS05-BP05 Perform patch management
- OPS05-BP06 Share design standards
- OPS05-BP07 Implement practices to improve code quality
- OPS05-BP08 Use multiple environments
- OPS05-BP09 Make frequent, small, reversible changes
- OPS05-BP10 Fully automate integration and deployment

Related documents:

- Introduction to the Three Lines Model
- Integrate across the Three Lines Model (Part 1)
- Integrate across the Three Lines Model (Part 2)
- Implementing a compliance and reporting strategy for NIST SP 800-53 Rev. 5
- Consolidating controls in Security Hub: The new controls view and consolidated findings

Related videos:

- How to implement compliance at scale with the Three Lines of Defense model: AWS AMER Summit Aug 2021
- AWS re:Invent 2025 - From Reactive to Proactive: Infrastructure governance by design (COP352)
- AWS re:Invent 2025 - From Code to Policies: Accelerate Development w/ IAM Policy Autopilot (SEC351)

Related services:

- AWS Security Hub CSPM
- AWS Config
- AWS Systems Manager Automation
- Amazon GuardDuty
- Amazon Inspector
- Amazon Macie
- AWS Firewall Manager
- AWS Organizations
- Amazon Security Lake

DSOPS02-BP03 Develop and track key compliance metrics

In highly regulated industries, developing and tracking compliance metrics is critical for demonstrating adherence to required standards.

Desired outcome: Teams use automated compliance metrics to reduce risk, accelerate remediation, and improve overall regulatory posture.

Common anti-patterns:

- Limiting compliance metrics visibility to a small group instead of sharing with relevant stakeholders and decision makers.
- Collecting and reporting compliance metrics manually instead of through automation.
- Treating compliance metrics as informational only rather than using them to improve compliance posture or reduce risks.

Benefits of establishing this best practice:

- Improved regulatory posture through data-driven decisions.
- Enhanced stakeholder confidence through transparent compliance reporting.
- Improved resource allocation based on data-driven compliance insights.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Developing and tracking compliance metrics requires a strategic approach that aligns with your organizational risk profile and regulatory requirements. Start by identifying a set of operational and business metrics. Categorize metrics by facets such as AWS account IDs, Regions, resource types (for example, EC2, S3, and Lambda), severity, owner, tags (for example, cost center or business unit), and security standards. The following are example metrics. Metrics and targets listed here are just for illustration and neither exhaustive nor accurate.

Category	Metric	Target	Actual
Operational	Critical findings open for more than 24 hours	0	
Operational	High findings open for more than 7 days	0	
Operational	Mean time to remediation for Critical Findings	<24 Hours	
Business	Audit preparation time	50% reduction year over year (YoY)	
Business	Cost of compliance per workload	5% reduction year over year (YoY)	
Business	Time to attain full compliance for new services	Target <7 days	

Implementation steps

The following steps provide instructions to activate AWS Security Hub CSPM and set up operational metrics within Security Hub. Refer to the equivalent documentation provided by the vendor if you use a different Cloud Security Posture Management (CSPM) tool.

1. **Enable AWS Security Hub CSPM and integrate compliance standards:**
   - Enable Security Hub in your AWS accounts:

     ```
     AWS securityhub enable-security-hub --region <your-region>
     ```

   - Enable compliance standards (for example, CIS, PCI-DSS, and HIPAA) in Security Hub:

     ```
     AWS securityhub enable-import-findings-for-product --product-arn
      arn:aws:securityhub:<region>::product/aws/securityhub --region <your-region>
     AWS securityhub batch-enable-standards --standards-subscription-requests
      StandardsArn="arn:aws:securityhub:::standards/cis-aws-foundations-benchmark/
     v/1.4.0" --region <your-region>
     ```

2. **Set up operational metrics**: The following implementation steps use AWS Security Hub. With Security Hub you can create custom insights to collect a specific set of findings and track issues that are unique to your environment. To get the number of open findings at a critical severity level for the last 30 days, follow these steps:
   - Create a custom insight using the AWS CLI:

     ```
     AWS securityhub create-insight --region <your-region> \
       --name "CriticalFindingsOver30Days" \
       --filters '{"SeverityLabel": [{"Value": "CRITICAL", "Comparison": "EQUALS"}],
      "RecordState": [{"Value": "ACTIVE", "Comparison": "EQUALS"}],"WorkflowStatus":
      [{"Value": "NEW", "Comparison": "EQUALS"}], "CreatedAt": [{"DateRange": {"Value":
      30, "Unit": "DAYS"}}]}' \
       --group-by-attribute "ResourceType"
     ```

   - Use the ARN of the insight to create an Amazon Simple Notification Service (Amazon SNS) topic to receive notifications:

     ```
     # EventBridge Rule (CloudFormation snippet)
     ```

```
Type: AWS::Events::Rule
Properties:
  Name: CriticalFindingsOver30DaysRule
  EventPattern:
    |
      {"detail":{"insightName":["CriticalFindingsOver30Days"]},"detail-type":
["Security Hub Insight Results"],"source":["aws.securityhub"]}
    |
  Targets:
      - Arn: arn:aws:sns:<region>:<account-id>:critical-alerts
```

3. **Monitor and optimize**:
   - **Build dashboards**: Use services to visualize metrics like Quick Suite or Amazon CloudWatch Dashboards.
   - **Review metrics regularly**: Periodically review AWS Security Hub's Findings Overview and custom reports.

Resources

Related best practices:

- SEC02-BP01 Use strong sign-in mechanisms
- OPS01-BP03 Evaluate governance requirements
- OPS01-BP04 Evaluate compliance requirements
- OPS03-BP02 Team members are empowered to take action when outcomes are at risk
- OPS03-BP03 Escalation is encouraged
- OPS03-BP04 Communications are timely, clear, and actionable
- OPS03-BP06 Team members are encouraged to maintain and grow their skill sets
- [AG.ACG.1] Adopt a risk-based compliance framework
- [AG.ACG.3] Automate deployment of detective controls
- [AG.SAD.2] Delegate identity and access management responsibilities
- [O.DIP.2] Centralize logs for enhanced security investigations

Related documents:

- Visualize AWS Security Hub Findings using Analytics and Business Intelligence Tools
- Understanding custom insights in Security Hub CSPM
- Automate continuous compliance at scale in AWS

Related examples:

- **AWS Security Hub Automated Response & Remediation**: GitHub: aws-solutions/automated-security-response-on-aws

Related services:

- AWS Security Hub
- AWS Config
- Amazon EventBridge
- AWS Lambda
- Amazon DynamoDB
- Amazon Athena

DSOPS03-BP01 Plan and prepare for audits

For customers in highly regulated industries, proactive audit planning assists organizations as they strive to meet their audit obligations with greater certainty and regularity.

Desired outcome: A streamlined, well-planned, and evidence-based audit process that demonstrates comprehensive adherence to applicable regulations while minimizing business disruptions.

Common anti-patterns:

- Manual evidence collection when automated solutions are available.
- Scrambling to gather documentation days before an audit, leading to incomplete or inaccurate evidence.
- Treating audits as solely an IT or security team responsibility rather than a cross-functional effort.
- Not auditing vendors and SaaS tools integrated with the workload.

Benefits of establishing this best practice:

- Scope of audit exercises is known in advance leading to better planning and resource utilization.
- Well-documented audit practices demonstrate due diligence to regulators and can reduce scrutiny.

- Audits are conducted in a structured, automated, and repeatable manner.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Effective audit preparation in AWS environments requires clear scoping, establishing automated evidence collection, and activating operational processes to manage audits end-to-end.

Key steps include:

- Establishing a compliance discovery and analysis methodology that assists with narrowing down the scope of an audit exercise.
- Creating clear documentation that maps specific regulatory requirements to the technical controls and operational practices. Many organizations develop and maintain a compliance matrix document which serves as a good preliminary evidence for auditors.
- Setting up continuous evidence collection mechanisms.
- Provisioning self-serve tools for auditors.
- Developing pre-built reporting templates (mapped to specific compliance standards), that can be populated on-demand by pulling near real-time data.
- Conducting regular internal assessments, and building a feedback loop geared towards addressing gaps found.

Implementation steps

1. **Conduct readiness assessments**:
   - Perform regular internal audits using the same criteria as external auditors.
   - If required, schedule third-party pre-assessments before formal audits to gain confidence.
   - Run tabletop exercises simulating audit scenarios.
   - Practice evidence retrieval and presentation.
2. **Establish audit governance processes**:
   - Create an audit coordination team with representatives from key departments.
   - Designate an audit owner to oversee audits. Audit owners are typically governance, risk, and compliance (GRC) professionals, such as a compliance officer or a General Data Protection Regulation (GDPR) data protection officer.
   - Develop a communication plan for internal and external auditors.
   - Establish a process for managing audit findings and remediation.

3. **Prepare audit artifacts**: Start building a searchable and readily accessible repository of frequently requested audit artifacts. The following is a list of some of the items you may need to collect:
   - Inventory of software and hardware assets.
   - Security policies, data protection policies and privacy policies.
   - Data protection impact assessment (DPIA) reports.
   - Risk registers.
   - Incident management plans.
   - Business continuity plans (bcps). Disaster recovery (dr) plans.
   - Documentation related to software development lifecycle (SDLC) processes (for example, data handling procedures and change management processes).
   - Design documentation (for example, data flow diagrams, up-to-date network diagrams, and records of architecture decisions).
   - Documents related to previous security incidents. This includes root cause analysis (RCA) reports.
   - Contractual agreements entered with your technology providers. Agreements entered between AWS and AWS Customers can be found in AWS Artifact.
   - Attestations and certifications currently held.
   - Audit trails and security-related logs. For example, Amazon CloudTrail Data and Management Event Logs, VPC Flow Logs, AWS WAF Logs, Amazon EKS Audit Logs, and Route 53 resolver query logs.
   - Records of training conducted on specific compliance-related topics.
4. **Pre-provision audit tools and services**:
   - Consider using AWS Audit Manager. Audit Manager provides prebuilt frameworks (for example, PCI DSS V3.2.1) that structure and automate assessments for a given compliance standard or regulation. This potentially shortens your audit timelines, reduces inaccuracies, and lowers expenses.
   - Provision read-only auditor roles. The AWS ReadOnlyAccess managed policy is an example of one such role. It allows auditors access to compliance related services such as AWS Audit Manager, AWS Security Hub, and AWS Config.

Resources

Related best practices:

- OPS02-BP02 Processes and procedures have identified owners
- OPS02-BP03 Operations activities have identified owners responsible for their performance

- OPS02-BP04 Mechanisms exist to manage responsibilities and ownership
- OPS02-BP05 Mechanisms exist to request additions, changes, and exceptions
- OPS02-BP06 Responsibilities between teams are predefined or negotiated

Related documentation:

- How AWS Audit Manager Simplifies Audit Preparation
- Prepare for an Audit in AWS Part 1 – AWS Audit Manager, AWS Config, and AWS Artifact
- Prepare for an Audit in AWS Part 2 – General Best Practices

Related services:

- AWS Audit Manager
- AWS Artifact
- AWS Security Hub
- AWS Config

DSOPS03-BP02 Automate evidence collection and reporting

Manual evidence collection is time-consuming, error-prone, and may not provide a comprehensive view of your AWS environment. Automating these processes improve accuracy, efficiency, and audit readiness.

Desired outcome: Automated evidence collection and reporting system that continuously gathers, organizes, and presents compliance data across your workloads with minimal manual intervention.

Common anti-patterns:

- Relying solely on manual screenshots and documentation for audit evidence.
- Collecting evidence only during audit periods rather than continuously.
- Storing evidence in formats that are not readily searchable or retrievable.

Benefits of establishing this best practice:

- Reduces audit preparation time from weeks to hours or days.
- Provides real-time visibility into compliance status and security posture.
- Reduces human error and maintains consistent quality of documentation.
- Enables proactive identification and remediation of compliance gaps.

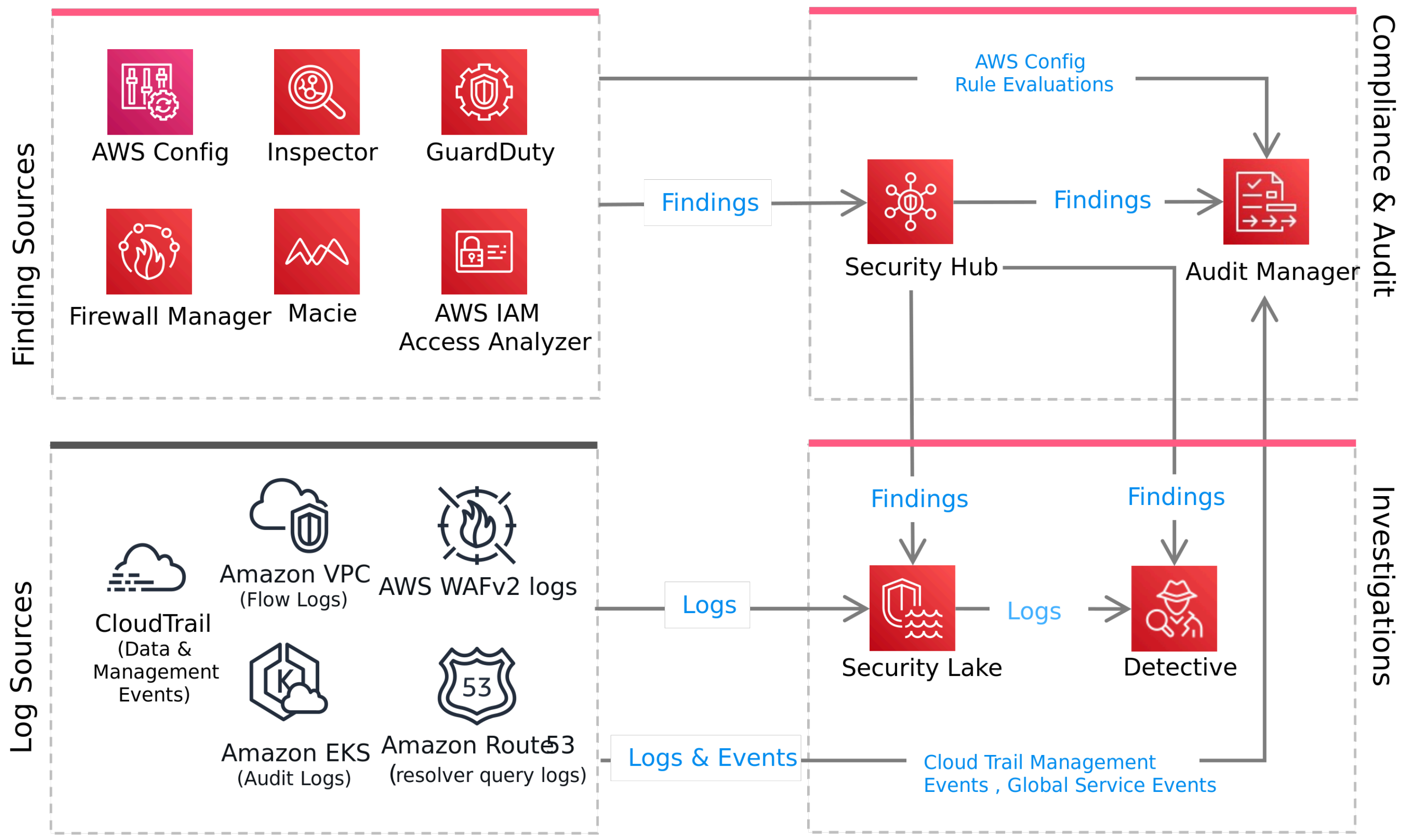
Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Implementing automated evidence collection requires a systematic approach that combines AWS services with third-party tools where necessary. Establish a centralized evidence repository with proper access controls, implement continuous monitoring and collection mechanisms, and create automated reporting workflows.

Implementation steps

Design audit-ready applications using a source, analyze, visualize, evidence (SAVE) paradigm. The following diagram illustrates how this paradigm applies in practice:



1. **Source**: Begin by identifying sources. There are two types of sources:
    - **Logs**: Collect and forward logs to a centralized location. Use a dedicated log archive account to store large volumes of logs securely, and maintain integrity of log files. Amazon Security Lake can collect logs and events from several supported AWS services. Security Lake automatically converts logs and events sourced from supported AWS services to the open-source Open Cybersecurity Schema Framework (OCSF) schema. After conversion to OCSF,

  Security Lake stores this data in an Amazon S3 bucket (one bucket per AWS Region) in your AWS account. Security Lake sources include:
  - AWS CloudTrail management and data events (S3, Lambda)
  - Amazon Elastic Kubernetes Service (Amazon EKS) Audit Logs
  - Amazon Route 53 resolver query logs
  - AWS Security Hub findings
  - Amazon Virtual Private Cloud (Amazon VPC) Flow Logs
  - AWS WAF logs
- **Findings**: AWS Config, Amazon Inspector, Amazon GuardDuty, Amazon Macie, and AWS IAM Access Analyzer can automatically detect and generate findings. These findings are derived from compliance drifts, threats, vulnerabilities, and over-permissive configurations. Consider enabling built-in security standards, with AWS Security Hub CSPM to automatically collect and correlate findings related to well-known security standards (for example, NIST 800-53 Rev. 5).

2. **Analyze**: You can use AWS services and third-party security information and event management (SIEM) tools to run analytical queries and discover new insights from logs. You can also use AWS services and third-party providers (known as finding providers) to conduct advanced analytics.
   - **Run custom analytics**: Amazon Security Lake offers several integrations with downstream analytics tools including:
     - Amazon Bedrock and Amazon SageMaker AI AI can generate AI-powered insights from Security Lake data.
     - Amazon Detective can investigate and identify the root cause of security findings or suspicious activities.
     - Amazon OpenSearch Service and Amazon OpenSearch Service ingestion pipeline can generate security insights from Security Lake data by using OpenSearch Service ingestion.
     - Since Security Lake normalizes logs into the OCSF format, you can forward data to several popular SIEM and analytics tools without having to build transformations.
   - **Use built-in intelligence**: Find providers like AWS services and third parties that use built-in intelligence to generate and send new findings to Security Hub. For AWS Security Hub, a finding is an observable record of a security check or a security-related detection. The provider analyzes your data and delivers findings, reducing the need for custom query development.

     For example, Amazon GuardDuty analyzes AWS logs and network traffic to detect threats and malicious activity in your AWS environment. It uses machine learning, anomaly detection, and integrated threat intelligence to identify unexpected and unauthorized activities like cryptocurrency mining, credential harvesting, and potentially compromised instances.

     Security Hub ingests and groups related findings to generate insights.

3. **Visualize:** Use built-in dashboards or build your own visualizations and make them available through self-service portals. Allow auditors access to self-service portals so that they can collect the evidence they need without having to rely on your teams.
   - **Use built-in dashboards**: AWS Config, Amazon GuardDuty, and Amazon Inspector provide built-in summary dashboards over findings. When you consolidate your findings to Security Hub, it can contextualize those findings, map it to known security standards, and present overall security scores. Security Hub also lets you create your own insights and build visualizations over those insights.
   - **Build your own visualizations**: Explore and interpret logs in Security Lake by combining with a query tool like Amazon Athena. Build visualizations and dashboards using business intelligence and reporting tools like Amazon Quick Suite. With Amazon OpenSearch Service, you can create a subscription that replicates data from Security Lake to your ingestion pipeline and build visualizations on top.
4. **Evidence:** A traditional audit process follows a pattern like the one below and can take multiple weeks. With an automated solution you can potentially reduce this effort.

	With automation	**Without automation**	**Automation capabilities**
Planning	1-2 weeks	<1 week	Pre-built mapping frameworks. Lists cloud provider technical controls mapping to known security standards.
Evidence requests	2-3 weeks	Immediate. No delays.	Pre-provisioned auditor roles. Auditors have real-time access to compliance dashboards.
Evidence review	3-4 weeks	1-2 weeks	Pre-organized and categorized by security standards.

	With automation	Without automation	Automation capabilities
Report writing	2-3 weeks	1-2 weeks	Automated report generation
Total	**8-12 weeks**	**3-5 weeks**	–

Consider using AWS Audit Manager to automatically collect evidence and generate reports. Audit Manager provides prebuilt frameworks that structure and automate assessments for a given compliance standard or regulation. You can create an assessment from a pre-built framework. When you create an assessment, Audit Manager automatically runs resource evaluations and collects evidences using built-in integrations with several AWS Services. The data that's collected is automatically transformed into audit-friendly evidence.

Audit Manager is extensible. For example, you can create a custom framework by wrapping over an AWS Config Conformance Pack. AWS Config is also extensible. You can develop custom rules with Config, have Audit Manager trigger those rules, collect the evidence, and produce audit-ready reports.

Resources

Related best practices:

- SEC04-BP01 Configure service and application logging
- SEC04-BP02 Analyze logs, findings, and metrics centrally
- OPS08-BP02 Analyze workload logs
- Best practice 5.4 – Secure the audit logs that record every data or resource access in analytics infrastructure

Related documents:

- Automate evidence gathering for compliance audit reports
- Audit Manager Mind Map
- How to visualize Amazon Security Lake findings with Quick Suite
- Introducing Amazon OpenSearch Service and Amazon Security Lake integration to simplify security analytics

Related videos:

- Visualizing Security Lake Data with Quick Suite: 2024 Quick Suite Learning Series
- Remediating Amazon GuardDuty and AWS Security Hub Findings
- AWS re:Invent 2025 - Observability & Security unite: Unify your data in Amazon CloudWatch (COP361)
- AWS re:Invent 2025 - Building agentic workflows for augmented observability (COP405)

Related examples:

- Workshop: AWS Cloud – An Auditors Lens
- Workshop: AWS Config Resource Compliance Dashboard - Part of Cloud Intelligence Dashboards Framework

Operate

DSOPS04: How do you monitor your current compliance status?

Monitoring compliance status is crucial as it enables organizations to proactively identify and address potential compliance gaps before they become significant issues. With continuous compliance monitoring, AWS customers can quickly detect unauthorized changes to their environment, verify that security controls remain effective, and maintain real-time awareness of their compliance posture across their cloud workloads.

DSOPS05: How do you remediate non-compliance?

Automate compliance remediation to maintain continuous regulatory alignment. Detect and correct violations as they occur instead of relying on manual intervention. This approach reduces operational overhead, minimizes human error and improves audit readiness.

Best practices

- DSOPS04-BP01 Maintain continuous visibility of your compliance status
- DSOPS04-BP02 Correlate security findings to compliance requirements
- DSOPS05-BP01 Enable independent root cause analysis and remediation
- DSOPS05-BP02 Automate compliance remediation

DSOPS04-BP01 Maintain continuous visibility of your compliance status

Proactive monitoring reduces risks of data breaches, misconfigurations, and audit failures by providing visibility into resource configurations and user activity.

Desired outcome: Improve compliance visibility with real-time alerts for deviations from regulatory requirements or internal policies.

Common anti-patterns:

- Current data collection processes exclude critical sources.
- Teams do not analyze collected data to identify compliance drifts, potentially missing opportunities for improvement.

Benefits of establishing this best practice:

- Increased visibility into compliance status across AWS resources.
- Reduced manual audit efforts and associated costs.
- Reduced risk of violations and associated penalties.
- Early detection of configuration drift and policy violations.
- Improves continual regulatory adherence rather than point-in-time assessments.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Start by enabling logging and monitoring services, then define compliance baselines using AWS tools. Prioritize the following:

1. Derive and aggregate findings from each source. Gathering useful telemetry data can improve your findings.

2. Prefer ready-made findings and known analysis methods where possible. For example, AWS Config, Amazon Inspector, Amazon GuardDuty, Amazon Macie, and AWS IAM Access Analyzer can automatically detect and generate findings resulting from compliance drifts, threats, vulnerabilities and over-permissive configurations.
3. Rank and score compliance findings by dimensions such as data sensitivity, network boundaries (for example, VPC or subnet), geo-location, trust boundary, and organizational priorities. Adjust weights as needed. Keep metrics updated with findings sourced in near real time.
4. Set alarms and send notifications to a pre-defined audience when thresholds are breached.

Implementation steps

1. **Collect logs**:
   - **Using AWS CloudTrail:** AWS CloudTrail is a service that enables governance, compliance, and operational auditing of your AWS account. It records and logs API activities and events that occur in your AWS account. These include management events, data events, network activity events and insights events.
     - Events contain information about who, what, when, how, and the outcome of the event. By default, CloudTrail logs management events only. These events are enabled by default and can be viewed through the AWS Management Console (or with the CLI) for up to 90 days. To retain data beyond 90 days, you can create your own CloudTrail trail and filter events by service, resource, or event type.
     - AWS services are integrated with CloudTrail. For example, when you provision an Amazon S3 bucket, you can enable both management and data events (for example,GetObject, DeleteObject, and PutObject).
   - **Other log sources:** Beyond CloudTrail, other important log sources include Amazon VPC Flow Logs, Amazon EKS Audit Logs, Amazon Route 53 DNS Query Logs, and AWS WAFv2 Logs. Additionally, capture logs generated by your applications and their dependent libraries and forward those to Amazon CloudWatch.
2. **Aggregate logs at a central place**:
   - **Aggregate CloudTrail logs**: AWS Control Tower creates a Log Archive account within a Security Organizational Unit (OU) when customers set up their own Landing Zone. Control Tower version 3.0 (released July 2022) automatically configures organizational logging with these capabilities.
     - An organization-level AWS CloudTrail trail is deployed in the organization's management account. This automatically captures actions of each member account.
     - The logs are stored in a central Amazon S3 bucket located in the Log Archive account.

    - By default, this trail is configured to log management events only.
    - You can also manually enable CloudTrail organization trails. For more detail, see Creating a trail for an organization.
  - **Aggregate Amazon CloudWatch logs**: You can strengthen your security and regulatory posture by collecting logs from applications that process sensitive data, including PII, PHI, and payment card data. To accomplish this, aggregate selected CloudWatch logs from your workload accounts into the centralized Log Archive account:
    - In each workload account, identify the CloudWatch log groups you want to centralize.
    - Create a CloudWatch Logs subscription filter for each log group.
    - Set the destination of the subscription filter to a CloudWatch logs destination in the Log Archive Account.
    - In the Log Archive Account, create a CloudWatch logs destination.
    - Configure the destination to point to an Amazon Data Firehose delivery stream. Create a Firehose delivery stream.
    - Configure the stream to deliver logs to an S3 bucket in the Log Archive Account.
    - Finally, for each CloudWatch log group in the workload accounts, enable the subscription to send logs to the destination in the Log Archive Account.

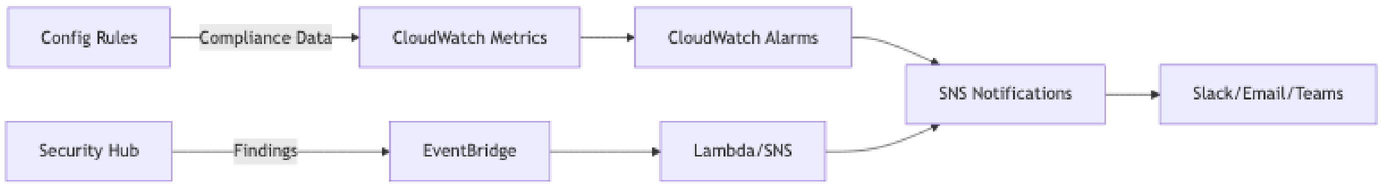
3. **Collect and aggregate findings:** Collect findings across your AWS environments by enabling compliance standards. There are two approaches, one using AWS Control Tower and other using AWS Security Hub CSPM.
   - **Using AWS Control Tower**: In a multi-account environment, we recommend starting with AWS Control Tower to set organization-wide, consistent guardrails aligned with compliance standards. AWS Control Tower consolidates 700-plus controls marked as *mandatory*, *strongly recommended*, and *elective*.
     - To assist in choosing, controls are also grouped by categories. The categories are by *common controls*, by *AWS services*, by *frameworks*, and by *groups*. If you have already set up a landing zone using Control Tower, we recommend evaluating the 240-plus controls under the *digital sovereignty* group. Alternatively, you can enable control by frameworks (for example, NIST-SP-800-53-r5, PCI-DSS-v4.0, CIS-AWS-Benchmark-v1.4, or CCCS-Medium-Cloud-Control-May-2019).
     - Technically, controls are implemented in three ways:
       - **Preventative controls:** Use service control policies (SCPs), resource control policies (RCPs), and declarative policies, which are part of AWS Organizations.
       - **Proactive controls:** Use AWS CloudFormation hooks and hooks managed by AWS Control Tower.
       - **Detective controls:** Use AWS Config.

  - Compliance findings are aggregated through AWS Security Hub CSPM and AWS Config Aggregators which reside in your audit account. Security Hub CSPM controls are identified in the AWS Control Tower console as SH.ControlID (for example, SH.CodeBuild.1). When you enable a Security Hub managed control in Control Tower, it also enables Security Hub CSPM for you.
- **Using AWS Security Hub CSPM**: In a single-account environment, consider starting from AWS Security Hub CSPM. Although it does not provide out-of-the-box preventative controls (unlike Control Tower), and is primarily meant to activate detective and proactive controls, it does provide a set of in-built security standards to map to. In AWS Security Hub, a *security standard* is a set of requirements that's based on regulatory frameworks, industry best practices, or company policies. Security Hub CSPM maps these requirements to controls, and runs security checks on the controls to assess whether the requirements of a standard are being met. The security checks result in Security Hub CSPM findings. Depending upon the integrations you enable and region availability, Security Hub CSPM receives findings from:
  - AWS Config
  - AWS Firewall Manager
  - Amazon GuardDuty
  - AWS Health
  - AWS Identity and Access Management Access Analyzer
  - Amazon Inspector
  - AWS IoT Device Defender
  - Amazon Macie
  - AWS Systems Manager Patch Manager

4. **Analyze findings**: AWS security and compliance services provide log and findings analysis capabilities through integrated machine learning and automated correlation engines.
   - Amazon CloudWatch Logs Insights enables SQL-like queries across massive log datasets to identify patterns and anomalies.
   - Amazon Detective uses graph analytics and machine learning to automatically correlate findings from GuardDuty, Security Hub, and Macie, creating visual timelines that assist security teams understand the relationships between entities, events, and potential threats.
   - Amazon Security Lake centralizes security data from multiple sources into a standardized format, enabling advanced analytics through Amazon Athena queries and integration with third-party Security Information and Event Management (SIEM) tools.
   - Amazon OpenSearch Service provides real-time search and visualization capabilities for security logs, allowing organizations to build custom dashboards and perform complex threat hunting across their entire security data landscape.

5. **Score and prioritize**: Security Hub CSPM uses a standardized scoring system to prioritize compliance findings based on severity and impact. Each finding receives a severity score from 0-100, where *informational* findings score 0, *low* findings score 1-39, *medium* findings score 40-69, *high* findings score 70-89, and *critical* findings score 90-100. The scoring considers multiple factors including the potential impact of the security issue, the exploitability of the vulnerability, and the confidence level of the detection mechanism. Security Hub also calculates a security score for each enabled standard, representing the percentage of passed security checks across the controls within that standard. This gives you a quantitative measure of your overall compliance posture that can be tracked over time and used to demonstrate improvement in security controls.

6. **Set up metrics, alarms, and notifications**: You can establish compliance monitoring through automated metrics collection, alerting, and multi-channel notifications. As an example, see the following diagram:



- AWS Config automatically publishes compliance metrics to CloudWatch, enabling you to create alarms based on the number of non-conforming resources, compliance percentage by rule, or configuration changes across your environment.
- Security Hub findings are automatically sent to Amazon EventBridge, where you can create rules to filter findings by severity, resource type, or compliance standard, then route them to AWS Lambda functions for custom processing, Amazon SNS topics for email or SMS notifications, or directly to EventBridge API destination partners such as Slack.
- Additionally, you can use CloudWatch composite alarms to create sophisticated alerting logic that combines multiple compliance metrics.

The following are example metrics. These are not exhaustive and are shown only for illustrative purposes.

Name	Category	Measurement	Engineering Value	Audit Value
Mean Time to Detection (MTTD)	Operational efficiency	Average time from when a compliance	Optimize monitoring	Shows proactive monitoring effectiveness

Name	Category	Measurement	Engineering Value	Audit Value
		violation occurs to when it's detected	coverage and alert tuning	
Mean Time to Identification (MTTI)	Operational efficiency	Average time from detection to identific ation of the root cause of compliance violations	Plug gaps in compliance data collection	Evidence of timely identific ation
Mean Time to Remediation (MTTR)	Operational efficiency	Average time from detection to full remediati on of complianc e issues. This includes duration spent in MTTI.	Identifies bottlenecks in complianc e remediation workflows	Evidence of timely correctiv e action
Critical finding backlog age	Risk and impact	How long high and critical severity findings remain unresolved	Prioritizes technical debt and resource allocation	Shows commitment to addressing high-risk issues promptly
Compliance drift rate	Risk and impact	Percentage of resources that drift from compliance over time periods	Indicates configuration management effectiveness	Demonstra tes ongoing continuous compliance efforts

Name	Category	Measurement	Engineering Value	Audit Value
Repeat violation rate	Risk and impact	Percentage of compliance issues that recur after remediati on	Identifies need for better root cause analysis and more thorough testing of remediation scripts	Shows effective ness of remediation scripts
Complianc e cost per resource	Business impact	Average cost of maintaini ng compliance per monitored resource	Optimize monitoring tool selection , configuration management practices, and compliance skills	Demonstrates cost-effective compliance management
Audit readiness score	BusinessiImpact	Percentage of complianc e evidence immediately available for audit requests	Reduces manual effort during audit preparati on	Streamlines audit processes and reduces examination time

Resources

Related best practices:

- OPS01-BP03 Evaluate governance requirements
- OPS01-BP04 Evaluate compliance requirements
- SEC01-BP03 Identify and validate control objectives
- SEC01-BP04 Stay up to date with security threats and recommendations
- SEC01-BP08 Evaluate and implement new security services and features regularly

- SEC04-BP01 Configure service and application logging
- SEC04-BP02 Capture logs, findings, and metrics in standardized locations
- SEC04-BP03 Correlate and enrich security alerts

Related documents:

- Analyzing AWS CloudTrail in Amazon CloudWatch
- How to detect and monitor Amazon Simple Storage Service (S3) access with AWS CloudTrail and Amazon CloudWatch
- Metrics for automated compliance and guardrails

Related videos:

- AWS re:Invent 2020: A security operator's guide to practical AWS CloudTrail analysis

DSOPS04-BP02 Correlate security findings to compliance requirements

Map and aggregate findings to avoid duplication and reduce information overload.

Desired outcome: Findings are correlated to regulatory requirements, enabling stakeholders to quickly identify compliance gaps and prioritize remediation efforts based on regulatory impact.

Common anti-patterns:

- Relying on generic controls without tailoring to specific regulatory requirements.
- Logs and findings are not archived for the long term. Even when stored, they are not readily searchable.
- Findings cannot be readily converted to audit evidence.

Benefits of establishing this best practice:

- Increased visibility into compliance status across AWS resources.
- Reduced risk of violations and associated penalties.
- Ability to demonstrate an adherent posture to auditors with comprehensive evidence.
- Better alignment between security, compliance, and operational teams.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Security signals generated by AWS services degrade in their value if not correlated and mapped onto specific policy and regulatory requirements. For example, why does it matter if an Amazon SQS queue is not configured to be encrypted at rest?

Security and compliance experts seek to aggregate and analyze security signals with a view to spot deviations from regulatory requirements or internal policies. This requires a deep understanding of compliance standards and is a time-consuming process. AWS Audit Manager, AWS Control Tower and AWS Security Hub are services that can reduce effort burden by separating the signal from the noise.

Implementation steps

1. **Enable AWS Control Tower controls**: Start by enabling AWS Control Tower to establish foundational guardrails across your multi-account environment, selecting compliance frameworks (such as NIST or PCI-DSS) that align with your regulatory requirements.
2. **Activate AWS Security Hub CSPM**: Activate AWS Security Hub CSPM in your audit account to centralize security findings gathered by enabling Control Tower controls and integrated security services like Amazon GuardDuty and Amazon Inspector. Configure Security Hub's compliance standards to match your regulatory frameworks, verifying that findings are automatically tagged to relevant compliance control identifiers. Consolidate Security Hub findings to reduce noise and operational overhead.
3. **Deploy AWS Audit Manager**: Deploy AWS Audit Manager to create custom assessment frameworks that map Security Hub findings and Control Tower compliance data to specific regulatory requirements and internal policies. Audit Manager continuously collects evidence from these sources. It automatically associates security findings with their corresponding regulatory controls. This approach provides a unified view where each security finding is traceable to specific regulatory requirements.
4. **Enrich Security Hub findings**: Consider creating Amazon EventBridge rules to capture Security Hub findings. You can then enrich these findings with additional information, such as the compliance control identifier, and store them in a centralized location. This allows you to correlate Security Hub findings with regulatory requirements and internal policies. You can enrich EventBridge findings with either custom Lambda functions or use EventBridge Pipes.

Resources

Related best practices:

- SEC04-BP03 Correlate and enrich security alerts
- [AG.ACG.1] Adopt a risk-based compliance framework
- [AG.ACG.3] Automate deployment of detective controls
- [AG.ACG.4] Strengthen security posture with ubiquitous preventative guardrails
- [AG.ACG.5] Automate compliance for data regulations and policies

Related documents:

- Metrics for automated compliance and guardrails
- AWS Compliance Center
- AWS Compliance Resources
- AWS Risk and Compliance Whitepaper
- AWS Shared Responsibility Model
- AWS services in scope by compliance programs

Related videos:

- AWS re:Invent 2020: A security operator's guide to practical AWS CloudTrail analysis
- AWS re:Invent 2021 - Cloud compliance, assurance, and auditing

DSOPS05-BP01 Enable independent root cause analysis and remediation

Provide engineering teams with the tools, knowledge, and permissions necessary to independently identify, analyze, and resolve compliance violations. Self-service capabilities reduce dependency on centralized security teams while accelerating remediation timelines and improving the overall security posture.

Desired outcome: Engineering teams can independently conduct root cause analysis and implement remediation for compliance violations within their scope of responsibility, using standardized tools and processes.

Common anti-patterns:

- Manual scanning of compliance reports and findings
- A limited number of fully vetted persons have access to security and compliance findings
- No standardized process for root cause analysis

- Missing automation and self-service capabilities
- Lack of historical compliance data for trend analysis

Benefits of establishing this best practice:

- Reduced Mean Time to Remediation through immediate team action
- Improved compliance posture through faster issue resolution
- Decreased operational load on security and compliance teams
- Enhanced security awareness and capability within engineering teams
- Better allocation of specialized security expertise towards resolving complex issues
- Improved team ownership and accountability for security outcomes

Level of risk exposed if this best practice is not established: Low

Implementation guidance

Effective Root Cause Analysis (RCA) and remediation requires the following steps:

1. Identifying resources having compliance issues
2. Collecting data related to those resources
3. Determining the root cause
4. Remediating the problem
5. Verifying that the remediation was successful
6. Documenting the solution

Implementation steps

1. **Identify and collect data**: Your chosen compliance and security tooling should create consolidated findings for developers to review. Consider the following.
    - **Scoped access**: Developers should only see compliance information about the resources they are responsible for.
    - **Cross-Service correlation**: Developers expect tools to correlate logs and events across multiple services, and generate a single set of actionable findings.
    - **Information relevancy**: Developers expect the following minimum information from findings.
        - Affected Resource ID, Account ID, and Region ID
        - Severity of the finding

- Source of evaluation. (For example, AWS Config Rules, Amazon Macie evaluations, Amazon GuardDuty evaluations)
- Date of evaluation
- The estimated impact of the finding
- Suggested remediation

2. **Integrate with AWS Services**: While there are several ways to identify and collect data related to non-compliant resources, we discuss 3 options below.

   - **Use AWS Security Hub and Amazon EventBridge**: Security Hub detects security and compliance issues by automatically correlating and enriching events from multiple sources. These include services providing security posture management (Security Hub CSPM), vulnerability management (Amazon Inspector), sensitive data detection (Amazon Macie), and threat detection (Amazon GuardDuty) related capabilities.

     Security Hub automatically sends new findings and updates to EventBridge as **events**. Findings appear as one of the following event types in EventBridge.

     - Security Hub Findings - Imported
     - Security Hub Findings - Custom Action
     - Security Hub Insight Results

     You can write EventBridge **rules** to match these event types. The following pattern will match the Security Hub Findings - Imported event type, extract the matching key-value pairs. For example, you can match against specific attributes like accountId, and region to selectively forward Security Hub generated events to the most appropriate recipients.

```
{
  "source": [
    "aws.securityhub"
  ],
  "detail-type": [
    "Security Hub Findings - Imported"
  ],
  "detail": {
    "findings": {
      "Region": [ "us-east-1"],
      "AccountId" : ["123456789012"]
    }
  }
}
```

EventBridge rules output JSON data structures that can be delivered to multiple targets, including Amazon SNS topics. This approach allows developers to receive scoped but detailed diagnostic information to conduct further root cause analysis.

- **Use Security Hub and Amazon Security Lake**: When you send Security Hub findings to Security Lake, it automatically constructs an AWS Glue database named amazon_security_lake_glue_db_<region_name> with associated tables. The database and tables are managed through AWS Lake Formation. By default Security Hub findings reside in a table named amazon_security_lake_table_<region_name>_sh_findings_2_0.

  Having a tabular representation of Security Hub findings offers several advantages. For example, you can perform the following actions:
  - Use Amazon Athena to query the Security Hub findings table.
  - Construct flattened views with a filtered set of columns, for example with accountid and region.
  - Additionally, apply fine-grained access control (FGAC) over your tables and views.
- **Use API-based Filtering**: Use the GetFindings API and apply filters for specific resource ARNs or tags, region, accountId and so on. Then build team-specific dashboards with Quick Suite to display those findings.

3. **Perform root cause analysis**: Once developers receive compliance notifications, they need access to root cause analysis (RCA) tools to investigate and understand the underlying issues. The specific tools and permissions required depend on the type of compliance violation and the AWS services involved.

   Two common RCA techniques are:
   - Five Whys Analysis
   - Ishikawa Diagrams

   Both techniques require the ability to collect and analyze data from multiple sources, such as logs, events, and metrics.

   When you enable Security Standards with Security Hub CSPM, Security Hub consolidates findings from multiple services, such as Config Rules, Macie, and GuardDuty, and generates a single set of actionable insights. The following example shows a non-compliant Amazon DynamoDB table, where delete protection is not enabled. The Security Hub finding provides the following information:
   - Affected Resource ID, Account ID, and Region ID
   - Severity of the finding

- Source of evaluation. (For example, Config Rules, Macie evaluations, GuardDuty evaluations). In this case the rule applied was DynamoDB.6.
- Date of evaluation
- The estimated impact of the finding
- Suggested remediation

The following snippet is from a finding triggered by the DynamoDB.6 rule

```
{
  "AwsAccountId": "XXXXXXX",
  "AwsAccountName": "XXXXXXX",
  "Compliance": {
    "Status": "FAILED",
    "SecurityControlId": "DynamoDB.6",
    "RelatedRequirements": ["NIST.800-53.r5 CA-9(1)", "NIST.800-53.r5
 CM-2","NIST.800-53.r5 CM-2(2)", "NIST.800-53.r5 CM-3","NIST.800-53.r5 SC-5(2)"
    ],
    "AssociatedStandards": [
      {
        "StandardsId": "standards/nist-800-53/v/5.0.0"
      }
    ]
  },
  "CreatedAt": "2025-01-23T02:26:29.771Z",
  "Description": "This control checks whether an Amazon DynamoDB table has deletion
 protection enabled. The control fails if a DynamoDB table doesn't have deletion
 protection enabled.",
  "FindingProviderFields": {
    ....
  },
  ...
  "Remediation": {
    "Recommendation": {
      "Text": "For information on how to correct this issue, consult the AWS Security
 Hub controls documentation.",
      "Url": "https://docs.aws.amazon.com/console/securityhub/DynamoDB.6/remediation"
    }
  },
  "Resources": [
    {
      "Details": {
        "AwsDynamoDbTable": {
          "TableId": "XXXXXXXXXXXXX",
          "TableName": "Table-Name-XXXXXXXXXX",
          "DeletionProtectionEnabled": false
        }
      },
      "Id": "arn:aws:dynamodb:region-id:XXXXXXXXXX:table/Table-Name-XXXXXXXXXX",
      "Partition": "AWS",
      "Region": "region-id",
      "Type": "AwsDynamoDbTable"
    }
  ]
  ...
}
```

Extract findings using one of the methods described in step 2 and send findings over to concerned engineering teams. Consider integrating with team messaging apps or your organizational IT Service Management (ITSM) software to enhance operational maturity.

4. **Remediation, verification, and documentation**: Once the root cause is identified, engineering teams can remediate the problem. This may involve automated remediation, manual remediation, or a combination of both. The team should also verify that the remediation was successful, and document the solution for future reference.

   AWS Systems Manager automation provides several runbooks to automatically remediate compliance violations. While there are no runbooks yet to remediate the DynamoDB.6 finding listed in this example, for more information, see this re:Post entry for remediation options.

Resources

Related best practices:

- SEC04-BP01 Configure service and application logging
- SEC04-BP02 Capture logs, findings, and metrics in standardized locations
- SEC04-BP03 Correlate and enrich security alerts
- SEC04-BP04 Initiate remediation for non-compliant resources
- SEC10-BP02 Develop incident management plans
- SEC10-BP04 Develop and test security incident response playbooks
- SEC10-BP05 Pre-provision access
- SEC10-BP06 Pre-deploy tools
- SEC10-BP07 Run simulations

- SEC10-BP08 Establish a framework for learning from incidents

Related documents:

- Visualizing AWS Config data using Amazon Athena and Quick Suite
- Deploy Conformance Packs across an Organization with Automatic Remediation
- Remediate non-compliant AWS Config rules with AWS Systems Manager Automation runbooks
- Automated Response and Remediation with AWS Security Hub
- Manage Custom AWS Config Rules with Remediation Using AWS Config Conformance Pack
- Analyzing AWS CloudTrail in Amazon CloudWatch

Related examples:

- Cloud Intelligence Dashboards - AWS Config Resource Compliance Dashboard (CRCD)

Related videos:

- AWS re:Invent 2025 - Building and validating cloud controls with generative AI (COP350)
- AWS re:Invent 2020: A security operator's guide to practical AWS CloudTrail analysis - Learn more about the AWS CloudTrail service and its value for security operations. The session dives deep into sources of data enrichment and reviews how to leverage AWS CloudTrail as part of your security operations and incident response procedures.
- Monitor AWS Resources with Scheduled Reports: 2024 Quick Suite Learning Series - Landing zone managers want a centralized view to understand compliance of different accounts and resources within their AWS Environments. With Quick Suite, you can achieve one dashboard using data exported from AWS Config summarizing information on accounts in the organization. Lastly, we can create reports with actionable insights for the owners of those accounts.

DSOPS05-BP02 Automate compliance remediation

Implement automated remediation to detect and correct compliance violations with minimal human intervention. Automation provides consistent, rapid response to known compliance issues while maintaining proper oversight and audit trails for remediation actions.

Desired outcome: Resources are automatically restored to compliance through reliable, tested, and auditable remediation processes that minimize manual intervention while maintaining oversight and rollback capabilities.

Common anti-patterns:

- Running untested remediation scripts in production
- Implementing remediations without rollback capabilities
- Missing approvals for high-risk changes
- Lack of audit trails for automated actions
- No handling of partial failures in multi-step remediations

Benefits of establishing this best practice:

- Reduced mean time to compliance restoration
- Consistent application of fixes
- Decreased operational overhead
- Improved audit readiness through automated logging
- Reduced risk of human error

Level of risk exposed if this best practice is not established: Low

Implementation guidance

After identifying and analyzing compliance issues, developers need the ability to select and apply automated remediations. The approach depends on the urgency, risk level, and organizational requirements for automation vs. manual control.

Consider the following aspects while setting up remediations.

1. **Start with low-risk remediations**: Begin with safe, reversible actions like removing public access
2. **Build a test environment**: Set up a controlled environment for testing remediations. Apply the same testing methods (unit test, integration test) as you use to test critical business functionality, to test your remediation scripts. Integrate your tests with your Continuous Integration (CI) pipeline.
3. **Implement approval gates**: Require approval before deploying high-impact changes
4. **Provide rollback capabilities**: Make sure remediations can be undone if needed
5. **Monitor remediation success**: Track remediation effectiveness and failure rates

Implementation steps

1. **Setup AWS Systems Manager Automation**: Remediations are applied using AWS Systems Manager Automation documents and can be run automatically upon triggering of compliance violations. AWS Config allows you to remediate non-compliant resources by integrating with these automation documents.

   AWS Config allows you to group both managed and custom rules into Conformance Packs. It also provides several sample Conformance Packs. These are categorized under best practices by service (for example S3, Lambda), or by standards (PCI DSS, NIST). You can customize these conformance packs per your needs and choose to enable rules applicable to you. Conformance packs templates are available from the AWS Config Rules GitHub repository.

   A powerful feature of conformance packs is you can include remediations while adding the rules that make up the pack. Consider the following code. Against the Config rule named S3BucketPublicReadProhibited, it also includes an automatic SSM remediation named AWS-DisableS3BucketPublicReadWrite. The Depends On parameter establishes the relation between the rule and the remediation. The following code is part of an example operational best practices for Amazon DynamoDB that includes sample remediations.

```
Resources:
  S3BucketPublicReadProhibited:
    Type: AWS::Config::ConfigRule
    Properties:
      ConfigRuleName: S3BucketPublicReadProhibited
      Description: >-
        Checks that your [Amazon S3](https://docs.aws.amazon.com/s3/index.html)
 buckets do not allow public read access.
        The rule checks the Block Public Access settings, the bucket policy, and the
        bucket access control list (ACL).
      Scope:
        ComplianceResourceTypes:
        - "AWS::S3::Bucket"
      Source:
        Owner: AWS
        SourceIdentifier: S3_BUCKET_PUBLIC_READ_PROHIBITED
      MaximumExecutionFrequency: Six_Hours
  S3BucketPublicReadProhibitedRemediation:
    DependsOn: S3BucketPublicReadProhibited
    Type: 'AWS::Config::RemediationConfiguration'
    Properties:
      ConfigRuleName: S3BucketPublicReadProhibited
      ResourceType: "AWS::S3::Bucket"
      TargetId: "AWS-DisableS3BucketPublicReadWrite"
      TargetType: "SSM_DOCUMENT"
      TargetVersion: "1"
      Parameters:
        AutomationAssumeRole:
          StaticValue:
            Values:
              - arn:aws:iam::<Account-Id>:role/S3OperationsAutomationsExecutionRole
     ...
```

With AWS Config you have the flexibility of defining your own custom rules (using CloudFormation Guard DSL, Lambda Functions), bundle them into custom conformance packs and trigger automated remediations.

2. **Build customized remediation workflows**: Instead of triggering automated remediations immediately, staging is another approach where compliance findings are collected (for example in a queue or a database), analyzed, and then the most appropriate response is determined. For example, AWS customer Lockheed Martin developed a custom remediation workflow, where they first check for exemptions before triggering Systems Manager Automations.

   Consider developing your own remediation workflows, when you:
   - Need to stage findings and choose between multiple remediations options,
   - Or need manual approvals prior to execution of automated runbooks.
3. **Create your own Systems Manager runbooks**: You can create your own runbooks to automate remediation tasks. Runbooks are written using YAML or JSON. You can use the visual design experience to expedite the process of creating custom runbooks. With the visual designer you can also create your own custom workflows. For example, you can drag and drop "Invoke Lambda Functions", "Start Step Function Execution", or "EventBridge Put Events" into the execution graph.
4. **Trigger remediation from AWS Security Hub findings**: Here's a typical architecture pattern customers can use to remediate non-compliant resources using AWS Security Hub and Amazon EventBridge. Refer to attachments section. It provides a framework implementation that you can use as a starting point.

Resources

Related best practices:

- SEC04-BP04 Initiate remediation for non-compliant resources
- SEC10-BP04 Develop and test security incident response playbooks

Related documents:

- Create your own Systems Manager runbooks
- Create a Systems Manager workflow
- AWS Prescriptive Guidance: Automate remediation for AWS Security Hub standard findings

Related videos:

- AWS re:Invent 2025 - From Reactive to Proactive: Infrastructure governance by design (COP352)

Evolve

DSOPS06: How are you set up to address regulatory changes?

Track regulatory changes to improve adherence to evolving data protection, privacy, and security regulations. This reduces risks of data breaches, financial penalties, and reputation damage while enabling you to navigate varying regulations across different jurisdictions for global operations.

Best practices

- DSOPS06-BP01 Track regulatory changes across operating Regions
- DSOPS06-BP02 Manage regulatory changes

DSOPS06-BP01 Track regulatory changes across operating Regions

Track, analyze, and adapt to regulatory changes across jurisdictions to maintain adherence and reduce risk.

Desired outcome: A resilient, compliance-aligned program that proactively identifies, assesses, and implements regulatory changes.

Common anti-patterns:

- Missing clear ownership and processes for regulatory change management.
- Focusing only on global standards while ignoring region-specific regulations.
- Relying solely on external consultants without building internal expertise.
- Addressing compliance issues reactively instead of proactively.

Benefits of establishing this best practice:

- Reduced risk through early identification of regulatory changes.
- Better decision-making for regional expansion and service adoption.
- Increased customer and partner trust through improved regulatory adherence.
- Lower costs by avoiding last-minute remediation.
- Improved alignment between legal, compliance, and technical teams.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Considering setting up a regulatory intelligence function to track new regulations, amendments to existing laws, and regulatory guidance from authorities across jurisdictions. Automate compliance checks and evidence collection with AWS Config, AWS Audit Manager, and AWS Security Hub. Track regulatory changes with third-party governance risk and compliance (GRC) tools. Adhere to regulatory requirements through regular testing, training, and documentation updates.

Implementation steps

1. **Establish a regulatory intelligence function:** Create a team of multi-disciplinary experts responsible for monitoring regulatory changes across geographic regions and jurisdictions where you operate. Monitor changes to globally recognized standards such as ISO, PCI, local privacy laws and cybersecurity standards. A regulatory intelligence function should also clearly articulate the impact of impending changes on existing business operations and technology solutions. Include the following:
   - Compliance officer with relevant certifications (such as certified information privacy professional (CIPP), certified information systems security professional (CISSP), or certified in risk and information systems control (CRISC))

- Legal counsel specializing in cybersecurity and data privacy legislation
- Cloud architects and security consultants with compliance expertise and certifications (such as AWS Certified Security - Specialty or Certified Cloud Security Professional (CCSP))
- Regional compliance specialists for each geographic Region or jurisdiction where you operate

2. **Establish partnerships with local legal experts:** Partner with local legal firms or compliance consultants in each operating region to gain expert insights on regulatory changes. When selecting partners, look for firms with:
   - Proven expertise in technology and data protection regulations
   - Active participation in regulatory consultations and industry working groups
   - Established relationships with local regulatory authorities
   - Track record of advising on cloud compliance matters
3. **Work with local regulatory authorities:** Engage with regulators and join consultation exercises. Use regulatory sandboxes where available to prepare for new regulations.
4. **Use AWS services and Marketplace offerings:** Integrate AWS compliance resources into your tracking system. Use AWS Artifact to access AWS compliance reports and agreements, which are updated as AWS achieves new certifications and attestations. While AWS Artifact provides compliance documentation, you'll need to supplement it with external sources to track regulatory changes themselves. Explore governance, risk, and compliance (GRC) solutions in AWS Marketplace.
5. **Implement GRC software:** Evaluate leading GRC products that complement AWS services. Solutions like OneTrust integrate with AWS to collect compliance evidence and establish traceability between requirements and controls.
6. **Update training curriculum:** Maintain training velocity and update your training courses regularly to align with changing regulations.
7. **Establish regular review cycles:**
   - Conduct quarterly reviews of compliance policies, control implementations, and workload configurations to verify they remain aligned with current regulations.
   - Watch for new regulations and regulatory changes related to cross-border data transfers and new requirements related to data residency.
   - Establish regulatory change management metrics and KPIs, such as:
     - Time from regulatory change announcement to implementation completion
     - Number of regulatory changes identified and assessed per quarter
   - Update documentation and controls as regulations evolve.

This approach combines technology, expertise, and processes to improve regulatory adherence in a complex and dynamic regulatory environment.

Resources

Related best practices:

- OPS01-BP03 Evaluate governance requirements
- OPS01-BP04 Evaluate compliance requirements
- OPS07-BP02 Maintain a consistent review of operational readiness
- REL08-BP04 Deploy using immutable infrastructure
- REL08-BP05 Deploy changes with automation
- OPS07-BP05 Make informed decisions to deploy systems and changes

Related documents:

- Amazon Web Services: Risk and Compliance
- Operational Readiness Reviews (ORR)

Related videos:

- AWS re:Inforce 2025 - Best practices for managing governance, risk, and compliance globally (GRC301)
- AWS re:Inforce 2024 - Automation in action: Strategies for risk mitigation (GRC301)
- AWS Summit ANZ 2021 - Build an effective governance and compliance strategy with AWS Audit Manager
- AWS re:Inforce 2023 - Using AI/ML to scale governance, risk management, and audits (GRC222)

DSOPS06-BP02 Manage regulatory changes

Regulatory changes directly impact cloud operations, security controls, and compliance status. A structured change management process helps you identify, assess, and implement regulatory requirements efficiently. This practice assists in avoiding compliance gaps and reducing risk exposure while maintaining operational continuity in AWS environments.

Desired outcome: A systematic process that proactively identifies, evaluates, and implements regulatory changes while maintaining regulatory adherence across each AWS environment.

Common anti-patterns:

- Technical or operational changes are not traced back to regulatory changes.

- Implementing regulatory changes without an effective change management process.
- Lacking integration between compliance processes and cloud operations.
- Using manual compliance checks that don't scale with cloud adoption.

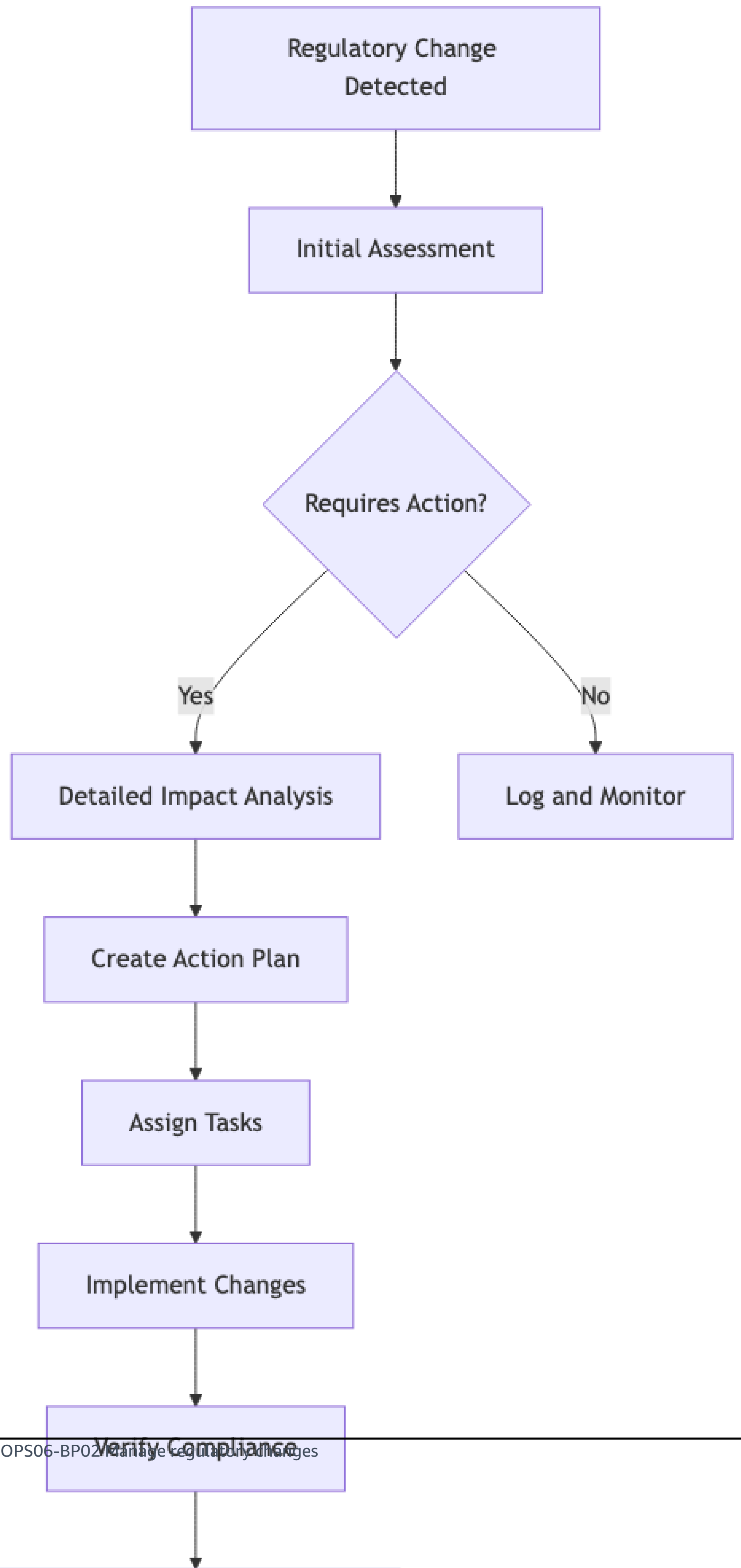
Benefits of establishing this best practice:

- Reduced risk through early identification of regulatory requirements.
- Lower costs by avoiding emergency remediation efforts.
- Increased business agility by incorporating regulatory adherence into development cycles.
- Improved audit readiness with documented evidence.
- Better decision-making for regional expansion and service adoption.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Create a regulatory change management function with defined roles, processes, and supporting technology. Establish regular review cycles and maintain comprehensive documentation of compliance activities. The following diagram shows an example change management process.


Regulatory Change Detected
Initial Assessment
Requires Action?
Yes
No
Detailed Impact Analysis
Log and Monitor
Create Action Plan
Assign Tasks
Implement Changes
Verify Compliance

Conducting a *detailed impact analysis* is an important part of this change management process. Consider asking the following questions in this step:

- What is the expected impact of implementing or not implementing the change?
- When should the change be implemented?
- Is this change applicable to each workload, or just a subset?
- How will the change be implemented?
- Is this an operational change or a technical change, or both?
- Does the change require updates to existing security and compliance policies?
- What compliance controls need to be developed or enabled? Are there existing controls that need to be modified or replaced?
- Will this require building new infrastructure, or new systems integrations?
- Are there third-party dependencies to consider?
- How will the change be verified?
- How will compliance be monitored?

Consider using AWS resource tagging to identify and track which resources are affected by specific regulatory requirements. Tags can help you scope the impact and automate compliance checks for affected resources.

Evaluate if the regulatory change mandates specific technical or operational measures for applications hosted on AWS infrastructure. For example, a healthcare regulator may mandate that the cloud service provider employs residents or citizens of a specific country or jurisdiction only for providing operational support.

Resources

Related best practices:

- OPS01-BP03 Evaluate governance requirements
- OPS01-BP04 Evaluate compliance requirements
- SEC02-BP04 Keep up to date with security recommendations
- SEC10-BP01 Identify compliance requirements

Related Services:

- AWS CloudFormation
- AWS Config

- AWS Control Tower
- AWS Security Hub
- AWS Well-Architected Tool

Security

The security pillar addresses how organizations establish and maintain secure, compliant workloads that meet digital sovereignty requirements. This includes building secure foundations, managing access controls, protecting sensitive data, detecting threats, securing infrastructure, and responding to incidents while adhering to jurisdictional regulations.

Topics

Definitions

The following are security-specific definitions.

- **Artificial intelligence/machine learning (AI/ML):** Technologies for automated analysis and decision-making. In sovereign security contexts, AI/ML can automate threat detection, and anomaly identification. However, organizations must consider data sovereignty implications when using AI services, including where training data is processed, where models are hosted, and whether inference operations remain within approved jurisdictions.
- **Application performance management (APM):** Tools and practices for monitoring application performance, availability, and user experience. For sovereign workloads, APM solutions must operate within approved regions, store telemetry data according to residency requirements, and provide visibility into cross-region data flows that might violate sovereignty constraints.
- **Automated reasoning:** Mathematical analysis of security configurations to prove properties about access policies and network reachability. Tools like AWS IAM Access Analyzer use automated reasoning to verify that IAM policies grant only intended access and to identify

potential unintended information exposure. Critical for sovereign workloads to mathematically prove that data access policies enforce jurisdictional boundaries.

- **Cyber incident response team (CIRT):** Dedicated team responsible for handling security incidents, including detection, analysis, containment, eradication, and recovery. For sovereign workloads, CIRT members must be located in approved jurisdictions, have appropriate security clearances, and follow incident response procedures that maintain data residency during forensic investigation and remediation.
- **Cyber threat intelligence (CTI):** Information about potential cybersecurity threats, including threat actor tactics, vulnerabilities, and indicators of compromise. Enables organizations to proactively defend against emerging threats. For sovereign environments, CTI feeds must be relevant to the region for them to be valuable.
- **Indicator of compromise (IoC):** Forensic evidence that a security breach has occurred or is occurring. Examples include suspicious IP addresses, malware signatures, unusual file hashes, or anomalous user behavior patterns. In sovereign contexts, IoC detection and analysis must occur within approved regions, and sharing IoCs with external threat intelligence solutions must comply with data export restrictions.
- **Just-in-time access:** Security practice of granting temporary, time-limited access only when needed, rather than permanent standing privileges. Reduces attack surface by minimizing the window of opportunity for credential compromise. For sovereign workloads, JIT access systems must verify that requesting users are in approved locations and maintain audit trails of all access grants and usage.
- **Large language model (LLM):** AI model trained on vast amounts of text data, capable of understanding and generating human-like text. Used in security for analyzing logs, generating security policies, and assisting with threat analysis. Organizations must consider where LLM processing occurs and whether prompts containing sensitive data remain within sovereign boundaries.
- **Mean Time to Detect (MTTD):** Average time required to identify a security incident from when it first occurs. Critical metric for security operations effectiveness. Sovereign constraints may impact MTTD if detection tools must operate within specific regions or if security analysts must be located in approved jurisdictions, potentially limiting 24/7 coverage.
- **Network reachability:** The ability to establish network connections between resources, determined by security groups, network ACLs, routing tables, and firewall rules. For sovereign architectures, network reachability analysis verifies that resources in approved regions cannot be accessed from unauthorized locations and that data cannot flow across jurisdictional boundaries.
- **Policy validation:** Process of verifying that access policies block unintended access and enforce intended security boundaries. Uses techniques like automated reasoning, policy simulation, and

access analysis. Essential for sovereign workloads to prove that IAM policies, resource policies, and SCPs enforce data residency and jurisdictional access requirements.

- **Root cause analysis (RCA):** Systematic investigation to identify the underlying causes of security incidents or operational failures. Goes beyond symptoms to find root issues. For sovereign workloads, RCA must be conducted by authorized personnel in approved locations, and forensic data must remain within jurisdictional boundaries throughout the investigation.
- **Session recording:** Capturing and storing all activities performed by operators during privileged access sessions, including commands run, files accessed, and configuration changes made. Provides audit trail for compliance and forensic investigation. For sovereign environments, session recordings must be stored in approved regions and protected with appropriate encryption and access controls.
- **Security information and event management (SIEM):** Centralized system for collecting, aggregating, analyzing, and correlating security logs and events from across the infrastructure. Provides real-time threat detection and compliance reporting. For sovereign workloads, SIEM infrastructure must operate within approved regions, and log data must not be transmitted outside jurisdictional boundaries for analysis.
- **Threat modeling:** Systematic approach to identifying security threats by analyzing system architecture, data flows, trust boundaries, and potential attack vectors. Assists in prioritizing security controls based on risk. For sovereign architectures, threat modeling must consider jurisdiction-specific threats such as foreign government access requests, cross-border data transfer risks, and geopolitical factors.
- **Tactics, techniques, and procedures (TTP):** Patterns of activities and methods used by threat actors to compromise systems and achieve their objectives. Understanding TTPs allows organizations to detect and defend against specific threat groups. Sovereign organizations must consider TTPs specific to nation-state actors and threats targeting regulated industries in their jurisdiction.
- **Tabletop exercise (TTX):** Simulated incident response scenario where team members discuss their roles and responses to a hypothetical security incident without actually performing the actions. Used to test incident response plans and identify gaps. For sovereign workloads, TTX scenarios should include jurisdiction-specific incidents like data sovereignty violations or unauthorized cross-border access.
- **User and entity behavior analytics (UEBA):** Technology that uses machine learning to establish baseline behavior patterns for users and entities (such as applications or devices), then detects anomalies that may indicate compromised accounts or insider threats. For sovereign environments, UEBA systems must process behavioral data within approved regions and alert on activities that violate jurisdictional access policies.

Design principles

- **Build sovereignty into architecture from day one:** Embed jurisdictional requirements, data residency controls, and regulatory compliance into your foundational architecture rather than adding them later. Use automated, technically enforced controls instead of relying solely on contractual agreements or manual processes.
- **Automate security validation and evidence collection:** Deploy automated mechanisms to continuously validate policy effectiveness, detect violations, and generate verifiable audit evidence. This reduces manual effort, improves consistent enforcement, and provides compliance-ready documentation for regulatory assessments.
- **Establish comprehensive visibility across data and access:** Implement automated discovery, classification, and monitoring of sensitive data throughout its lifecycle. Maintain detailed audit trails of access patterns, data movements, and security events to support governance and incident response.
- **Apply defense in depth with regional control:** Layer multiple security controls across identity, network, data, and application boundaries while verifying that operational teams within approved jurisdictions maintain authority over critical security functions. Combine preventive, detective, and responsive controls for comprehensive protection.
- **Protect data throughout its complete lifecycle:** Secure data at rest, in transit, and during compute using encryption, confidential computing, and privacy-enhancing technologies. Localize cryptographic operations within designated boundaries and implement specialized protections for highly sensitive workloads.

Security foundations

DSSEC01: How do you establish secure foundations aligned with sovereign security and privacy standards?

Establishing secure foundations aligned with security and privacy standards provides organizations with a proven, compliant architectural baseline that assist to verify that security and regulatory requirements are met from day one.

Best practices

- DSSEC01-BP01 Establish secure foundations aligned with regulatory requirements

DSSEC01-BP01 Establish secure foundations aligned with regulatory requirements

Establish secure foundations aligned with cybersecurity standards using proven, compliant architectural baselines that meet security and regulatory requirements from day one. A well-architected security foundation provides the necessary guardrails and automated controls to protect sensitive data and maintain regulatory adherence across multi-account environments.

Desired outcome: Security and compliance are built-in from day one, with automated guardrails, continuous monitoring, and comprehensive logging aligned with regulatory requirements.

Common anti-patterns:

- Implementing security controls as an afterthought rather than building them into the foundational architecture.
- Using ad-hoc security configurations without standardized baselines or compliance frameworks.
- Relying on manual security processes instead of automated, policy-driven controls.
- Implementing security controls that don't align with regulatory requirements.
- Implementing security architectures without considering the full lifecycle of compliance and audit requirements.

Benefits of establishing this best practice:

- Accelerates deployment timelines by providing pre-configured security controls and compliance frameworks.
- Maintains consistent security posture across accounts and workloads through automated governance.
- Simplifies audit and compliance reporting through built-in monitoring and logging capabilities.
- Provides data sovereignty controls that meet regulatory requirements for data residency and privacy.
- Enables scalable security operations through centralized management and automated remediation.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Organizations in highly regulated industries must establish security foundations that demonstrate compliance from day one while enabling business agility.

The AWS Security Reference Architecture (SRA) provides the architectural blueprint for implementing security controls across your infrastructure layers. By following proven patterns and using solution accelerators (pre-built, tested solutions that speed up implementation of common security and compliance requirements) organizations can implement security controls that align with frameworks such as NIST, ISO 27001, SOC 2, and industry-specific regulations while maintaining the flexibility to adapt to evolving requirements.

Consider using solution accelerators to set up foundational landing zone capabilities. The Landing Zone Accelerator (LZA) on AWS extends AWS Control Tower by automating the deployment of additional security controls, compliance frameworks, and governance policies. It provides infrastructure-as-code templates that implement security best practices and regulatory requirements across your multi-account environment.

You can find additional partner built Landing Zone Accelerator solutions from the AWS Digital Sovereignty Marketplace.

Implementation steps

1. **Deploy foundational landing zone with AWS Control Tower**: Enable Control Tower in your management account to establish organizational units (OUs), baseline security controls, and centralized logging. This creates the foundation for multi-account governance.
2. **Enhance security capabilities with Landing Zone Accelerators**:
   - Deploy Landing Zone Accelerator (LZA) on AWS or equivalent partner solutions to automate deployment of additional security controls beyond Control Tower's baseline.
   - Select and customize compliance frameworks specific to your regulatory requirements (NIST, ISO 27001, PCI DSS, HIPAA). For the AWS LZA, review the sample configurations on GitHub and adapt them to your organization's needs.
   - For region-specific compliance requirements, review these regional landing zone implementations:
     - Baseline Informatiebeveiliging Overheid (BIO) for the Dutch Public Sector.
     - Spain's National Security Framework (ENS).
     - Germany's Cloud Computing Compliance Criteria Catalogue (C5).

- Configure data sovereignty controls including encryption at rest and in transit, centralized key management with AWS KMS, and data residency policies. Enable controls from the "Digital Sovereignty group" in Control Tower to enforce data residency requirements.

3. **Implement security best practices and patterns**:
   - Deploy security controls across network, identity, data, and application layers. Implement network segmentation, VPC isolation, and centralized egress controls following the perimeter security guidance in the AWS SRA.
   - Establish data perimeters using identity-based, network-based, and resource-based controls to block unauthorized access and accidental data exposure. Follow Data Perimeter on AWS and implement data perimeter policy examples using service control policies and resource policies.
   - For workloads processing personal data, implement the AWS Privacy Reference Architecture (AWS PRA). Consider creating a dedicated personal data (PD) organizational unit with enhanced controls for collecting, storing, and processing personal data, as detailed in the PRA organization account structure.
4. **Apply industry-specific best practices**:
   - Review and implement industry-specific guidance from AWS Well-Architected Lenses that address unique regulatory and security requirements for your sector:
     - Healthcare Industry Lens for HIPAA and healthcare data protection
     - Financial Services Industry Lens for financial regulations and data security
     - Government Lens for public sector compliance requirements
   - For specialized use cases or additional implementation guidance, explore Prescriptive Guides, Reference Architectures, and Solution Accelerators on the AWS Architecture Center that provide detailed implementation patterns for specific scenarios.

Resources

Related best practices:

- SEC01-BP01 Separate workloads using accounts
- SEC01-BP03 Identify and validate control objectives
- SEC03-BP05 Define permission guardrails for your organization
- SEC01-BP06 Automate deployment of standard security controls

Related documents:

- AWS Security Reference Architecture

- Landing Zone Accelerator on AWS Implementation Guide
- AWS Control Tower User Guide
- AWS Well-Architected Security Pillar
- AWS Compliance Center

Related videos:

- Establishing a Data Perimeter on AWS, RSA Conference
- AWS re:Invent 2025 - Building Sovereign Cloud Environments (COP409)
- AWS re:Invent 2025 - Advanced AI Security: Architecting Defense-in-Depth for AI Workloads (SEC410)
- AWS re:Invent 2025 - AWS Security Hub: Unifying & simplifying security operations at scale (SEC228)
- AWS Security Reference Architecture: Visualize your security - NDC Security 2024

Related examples:

- Landing Zone Accelerator Sample Configurations
- AWS Security Reference Architecture GitHub Repository
- AWS Well-Architected Labs - Security

Related services:

- AWS Control Tower
- AWS Security Hub
- AWS Config
- Amazon GuardDuty
- AWS IAM Identity Center
- AWS KMS
- Amazon Macie
- AWS CloudTrail

Identity and access management

DSSEC02: How do you record and control access to data?

Recording and controlling access to data is essential for meeting key requirements of regulations like GDPR and HIPAA, which mandate strict oversight of who can access sensitive personal and healthcare information, and under what circumstances.

DSSEC03: How do you prove the effectiveness of your data access policies in blocking unintended information exposure?

Measuring the effectiveness of data access policies is critical to verify that they are actually blocking unauthorized data exposure rather than just appearing compliant on paper. Through continuous monitoring and regular assessment, organizations can validate their controls are working as intended and identify potential gaps.

Best practices

- DSSEC02-BP01 Establish comprehensive logging and monitoring of user actions
- DSSEC02-BP02 Control access to sensitive data
- DSSEC03-BP01 Validate policy effectiveness through automated analysis
- DSSEC03-BP02 Verify network security posture through automated analysis

DSSEC02-BP01 Establish comprehensive logging and monitoring of user actions

Comprehensive logging enables organizations to track user activities, detect unauthorized access, and provide evidence during security incidents or audits. This is a key capability required by sovereign workloads. The logging principles outlined here apply universally, assisting organizations to safeguard their operations and maintain data integrity across their digital infrastructure.

Desired outcome: Complete visibility into system activities and data access with tamper-proof logs stored in compliant regions, enabling rapid incident response and regulatory audits.

Common anti-patterns:

- Failing to enable comprehensive logging across AWS services, Regions, and accounts.
- Implementing inconsistent logging formats and standards across services.
- Not protecting log data from unauthorized access or modification.
- Not defining and enforcing consistent retention periods.
- Storing logs in regions that violate data residency requirements.
- Failing to regularly validate logging mechanisms for completeness, accuracy and integrity.

Benefits of establishing this best practice:

- Provides evidence for regulatory audits, demonstrates adherence to digital sovereignty requirements (such as GDPR in Europe, data localization laws in specific countries), and satisfies industry-specific compliance standards.
- Enables faster detection, thorough investigation, and more effective remediation of security incidents while maintaining sovereign control over security operations.
- Tracks data access patterns, supports data protection initiatives, and enables verification of compliance with data residency requirements.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Implementing effective logging for digital sovereignty requires a strategic approach addressing data residency constraints, retention requirements, and security controls. Your logging solution should be designed to be comprehensive, tamper-resistant, and aligned with local regulations. Verify logs are only accessible to authorized personnel and within authorized jurisdictions.

Key implementation considerations include defining logging scope based on regulatory requirements, implementing centralized log aggregation with data residency controls, establishing retention periods with immutability where required, and developing automated monitoring and alerting capabilities. These considerations are detailed in the implementation steps below.

Implementation steps

1. **Define logging requirements and retention periods:**

- Document specific logging requirements based on your industry and jurisdictional regulations.
- Identify mandatory retention periods for different types of logs.
- Determine data residency constraints for log storage.
- Define the scope of actions and access events that must be logged. For example, consider *who* accessed *what* and from *where*, as some of the minimum information required in log files. The granularity of logging needs to be carefully calibrated - while some systems may require detailed transaction logs, others might only need summary-level information.
- Identify critical systems and data that require enhanced logging.

2. **Configure AWS CloudTrail across accounts and Regions:**
   - Enable organization-wide AWS CloudTrail trails that capture read and write management events across AWS accounts.
   - Enable data events for sensitive S3 buckets, Lambda functions, DynamoDB tables, and other data services identified in the previous step.
   - Configure log file validation to detect unauthorized modifications. See this guide, Validating CloudTrail log file integrity.
3. **Based on the defined requirements, enable service-specific logging:**
   - Enable VPC Flow Logs for network traffic monitoring.
   - Configure AWS Config to record resource configuration changes.
   - Set up Amazon S3 server access logging for data access patterns.
   - Configure Amazon RDS and Amazon Aurora database audit logging.
   - Activate Amazon GuardDuty and Amazon Inspector as sources of additional security findings.
4. **Establish centralized log storage with sovereignty controls:** The implementation of a centralized logging system must prioritize data residency compliance while maintaining operational efficiency. This involves selecting and deploying log aggregation solutions that can enforce geographic data boundaries while providing comprehensive coverage.
   - Create dedicated log archive accounts within regions that meet your data residency requirements (for example, AWS Regions in specific countries or geographic areas that comply with local data sovereignty laws).
   - Implement IAM policies to restrict access, modification, or deletion of log files.
   - Implement log retention policies in line with jurisdictional data retention requirements.
   - Configure MFA for sensitive data access or tasks (for example, deletion of data)
   - Control replication of log files to other AWS Regions to maintain data sovereignty:
     - Using service control policies: Apply deny actions for PutReplicationConfiguration and DeleteReplicationConfiguration to specific S3 buckets containing sensitive log files, and add Region deny controls.

    - Using IAM policies: Blocks users from setting up replication using IAM Permission Boundaries.
    - Using S3 bucket policies: Temporarily block data transfers between AWS Regions in Amazon S3 to block replication of log files.
  - Apply encryption using AWS KMS with appropriate key controls. Understand security considerations before creating multi-Region keys.
  - Configure immutable S3 buckets using object locks with a write-once-read-many (WORM) configuration. Consider AWS CloudTrail Lake, which provides immutable, queryable storage of CloudTrail events for up to 7 years, simplifying compliance audits and forensic investigations without managing S3 buckets directly.

5. **Implement log analysis and monitoring:** An effective logging system must include robust analysis, monitoring and alerting capabilities that provide real-time visibility into system activities. This involves implementing continuous log analysis for critical events and developing anomaly detection algorithms that can identify unusual patterns or potential security incidents.
   - Configure Amazon CloudWatch Logs to monitor your trail logs. Detect and send notifications for unauthorized access or suspicious activities against security and compliance data, including:
     - Attempts to modify or delete security and compliance data
     - Attempts to modify security configurations protecting the data (such as unauthorized access to encryption keys)
   - Create dedicated IAM roles for log analysis and audit functions
   - Configure Amazon EventBridge rules for automated responses to critical events (such as IAM policy changes, encryption key deletion, or root account usage)
   - Consider additional AWS services such as Amazon OpenSearch Service, Quick Suite, or third-party solutions for enhanced analytics, reporting, and visualization
6. **Verify logging completeness and compliance:**
   - Regularly audit logging configurations against requirements:
     - Perform periodic testing to verify logs capture critical activities (such as IAM policy changes, S3 bucket deletions, or security group modifications)
     - Conduct simulated security incidents to test the effectiveness of your logging configurations. Consider using AWS Well-Architected Labs - Security for hands-on practice.
   - Validate log retention periods. Log retention policies must define clear retention periods based on both legal requirements and operational needs, implementing automated log rotation and archival strategies to manage data lifecycle. Use automated test cases (such as AWS Config Rules or custom Lambda functions) to continuously check configurations.

7. **Validate third-party controls:** Organizations often use third-party services that generate, process, or consume security and compliance data (such as application performance management (APM) SaaS providers, security information and event management (SIEM) systems, or compliance management solutions). To verify these services meet your sovereignty and compliance requirements:
   - Review the vendor's compliance:
     - Request relevant accreditations or certifications specific to the services they provide (such as SOC 2, ISO 27001, or Region-specific certifications)
     - Verify the data residency of the service and confirm it aligns with your sovereignty requirements. Where necessary, validate the security clearance and location of personnel who have access to your security and compliance data.
   - Assess technical integration points: Verify API security controls (such as authentication, encryption in transit, and rate limiting), review IAM federation configurations to maintain least privilege access, confirm data transfer mechanisms comply with your data residency requirements, and test integration points to verify logs are transmitted securely and completely.
8. **Implement continuous improvement:**
   - Regularly review and update your logging strategy based on regulatory changes and emerging threats
   - Assess new AWS services for logging requirements as you adopt them
   - Optimize log storage and analysis for cost and performance:
     - Use S3 Intelligent-Tiering to automatically move logs to cost-effective storage tiers
     - Implement lifecycle policies to transition older logs to Amazon Glacier for long-term retention
     - Review AWS Cost Optimization best practices for additional guidance
   - Incorporate feedback from security teams, auditors, and incident response exercises
   - Conduct regular training on log analysis techniques using resources like AWS Skill Builder and AWS Security workshops

Resources

Related best practices:

- SEC04-BP01 Configure service and application logging
- SEC04-BP02 Capture logs, findings, and metrics in standardized locations
- SEC04-BP03 Correlate and enrich security alerts

- SEC10-BP03 Prepare forensic capabilities
- SEC08-BP02 Enforce encryption at rest

Related services:

- AWS CloudTrail
- Amazon CloudWatch Logs
- Amazon S3
- AWS Key Management Service (KMS)
- Amazon OpenSearch Service
- AWS Config

Related documents:

- AWS Security Incident Response Technical Guide
- Centralized logging and monitoring
- Build your own centralized log analytics platform with Amazon OpenSearch Service
- The AWS Security Reference Architecture - Log Archive account

DSSEC02-BP02 Control access to sensitive data

Data access controls are fundamental to digital sovereignty, verifying that only authorized users and services can access data in adherence to regulatory requirements.

Desired outcome: Data remains accessible only to authorized users and services within designated sovereign boundaries, with unauthorized access attempts blocked before they occur.

Common anti-patterns:

- Zone of trust is not known or not clearly established.
- Relying on a single layer of defense. For example using only detective controls to detect violations, rather than applying preventative controls to stop violations in the first place.
- Not considering cross-service and intra-service data flows across AWS Regions.
- Overlooking encryption key management, including the residency of keys and the location of encryption and decryption operations.

Benefits of establishing this best practice:

- Maintain adherence to regional data sovereignty requirements.
- Enables developers to modify and extend application functionality without inadvertently exposing data.
- Improves transparency and visibility of data access controls leading to better auditability.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Implement granular data access controls for digital sovereignty through a layered approach:

Foundation: Establish a data perimeter using the three core principles - trusted identities accessing trusted resources from expected networks. This creates your primary zone of trust and blocks unauthorized access attempts.

Policy controls: Use IAM global condition keys to restrict access by region, IP address, VPC, and other request properties. Apply these controls to both identity-based and resource-based policies for comprehensive coverage.

Network security: Implement VPC architecture with security groups, leverage AWS PrivateLink for private connectivity, and use DNS controls to block resolution of non-compliant endpoints.

Data protection: Classify and tag resources for compliance, then enable encryption at rest across data storage services using appropriate KMS keys.

Monitoring: Prioritize preventive controls over detective controls, but implement comprehensive logging with CloudTrail, CloudWatch, and AWS Config rules to detect and remediate compliance gaps.

Consider the following implementation steps.

Implementation steps

1. **Build a data perimeter**: Begin by building a data perimeter that clearly establishes your zone of trust. Then apply principles of least privilege on AWS principals and resources. A data perimeter is a set of preventive controls that verifies that only *trusted identities* are accessing *trusted resources* from *expected networks*. This is a foundational step designed to block untrusted entities from accessing sensitive data held within your accounts. Use AWS IAM Access Analyzer to

   regularly validate your policies and maintain visible, transparent, and enforceable controls. The following 3 principles are central to this:

   1. **Only trusted identities**: Only *trusted identities* can access *my resources* and only *trusted identities* are allowed from *my networks*.
   2. **Only trusted resources**: *My principals* can only access *trusted resources* and that access from *my networks* only targets *trusted resources* (regardless of the principal involved).
   3. **Only expected networks**: Only *expected networks* can be the source of requests from *my principals* or to *my resources*.
   4. For more detail, see Blog Post Series: Establishing a Data Perimeter on AWS.

2. **Use global-condition keys to restrict access**: Create IAM policies that use global-condition keys. You can restrict access to AWS resources by requested region, IP Address, organization ID, source VPC, source VPC Endpoint and several other properties sent as part of API requests. For example, you can:
   - Restrict users to make changes to Amazon EC2 instances in the eu-central-1 Region only, by using the aws:RequestedRegion condition key.
   - Restrict users access to an Amazon S3 bucket unless their aws:SourceIP address (CIDR range) matches the condition specified.
   - Use the aws:SourceVpc condition to deny access to an S3 bucket unless the request originates from within a specific VPC.
3. **Apply resource-based policies:** Global condition keys can be combined with resource-based policies. For example, you can:
   - Deny access to an S3 bucket unless the request originates from a specific VPC endpoint.
   - Use a VPC Endpoint policy (a type of resource-based policy) to restrict which S3 Buckets can be accessed from a VPC Gateway type endpoint. Similar restrictions can also be applied to VPC Interface endpoints and Gateway Load Balancer endpoints. See Control access to VPC endpoints using endpoint policies - AWS PrivateLink.
   - For a complete list of global condition keys, see AWS global condition context keys.
4. **Implement network-level controls**: Apply an additional layer of enforcement using network-level controls. Use VPC Design and Security Groups to implement a strict VPC architecture with security groups that limit traffic flows:
   - Establish zonal separation based on data sensitivity, access patterns, privacy requirements and export controls. Separating Accounts, VPCs, applying Network ACLs and Security groups are a few strategies that you should consider.
   - Inspect and restrict traffic between zones. The Building a Scalable and Secure Multi-VPC AWS Network Infrastructure whitepaper provides several blueprints for you to consider.
   - Limit internet access to block unauthorized data transfers.

- Use AWS PrivateLink to establish private connectivity between VPCs and services:
  - Create interface endpoints for AWS services
  - Implement endpoint policies to restrict access
  - Avoid using public endpoints for sensitive services
  - Monitor endpoint usage for compliance
- Implement DNS controls to block resolution of non-compliant endpoints:
  - Use Amazon Route 53 Resolver rules to control DNS resolution
  - Block resolution of service endpoints in non-compliant Regions
  - Implement DNS query logging for auditing purposes

5. **Implement data protection controls**: Encrypt data at rest so that, if unauthorized parties were to somehow obtain the encrypted data, the unencrypted plaintext would not be available to them. Consider using AWS KMS keys with post-quantum cryptographic algorithms for enhanced security. For more information, see AWS KMS post-quantum TLS.
   - Start by implementing a comprehensive data classification and tagging strategy:
     - Tag resources with data classification tags (such as data-classification:sovereign, data-classification:public, or data-classification:confidential) to identify sovereignty requirements
     - Create automated processes that verify tag compliance
     - Implement tag-based access controls where possible
   - Enable encryption at rest for data storage services. The following are examples. Refer to the security documentation for each AWS service you use for complete encryption guidance:
     - Enable S3 bucket default encryption with customer-managed KMS keys
     - Enable Amazon EBS encryption by default in Regions where you operate
     - Enable Amazon RDS storage encryption for databases
     - Use encrypted AMIs for EC2 instances For detailed encryption guidance, see Setting default server-side encryption behavior for Amazon S3 buckets and the security documentation for each service.
6. **Implement logging and monitoring**: Prioritize preventive controls over detective controls, because it is more secure to block non-compliant access attempts before they occur rather than to detect such attempts after they occur. Apply detective controls to fill in coverage gaps and to provide an extra layer of defense. This can be considered as manual enforcement of data access controls.
   - Implement AWS CloudTrail and Amazon CloudWatch for comprehensive logging and monitoring:
     - Enable CloudTrail in Regions where you operate with a single trail
     - Create CloudWatch alarms for suspicious activities
     - Set up automated notifications for compliance issues

- Implement AWS Config rules to detect non-compliant resources
  - Create custom rules for region-specific compliance
  - Set up remediation actions for non-compliant resources
  - Aggregate config data to a central account
  - Generate compliance reports regularly

Resources

Related best practices:

- SEC08-BP04 Enforce access control
- SEC01-BP06 Automate testing and validation of security controls in pipelines

Related documents:

- Identity and Access Management
- Data Residency: AWS Policy Perspectives
- Data Residency with Hybrid Cloud Services Lens - AWS Well-Architected
- Data Classification
- Data security and risk management

Related videos:

- AWS re:Invent 2024 - Amazon S3 security and access control best practices
- AWS re:Invent 2019 - Provable access control: Know who can access your AWS resources
- AWS re:Inforce 2022 - AWS Identity and Access Management (IAM) deep dive
- AWS re:Invent 2018: The Theory and Math Behind Data Privacy and Security Assurance

Related services:

- AWS Identity and Access Management (IAM)
- AWS Key Management Service (KMS)
- AWS Organizations

DSSEC03-BP01 Validate policy effectiveness through automated analysis

Manual policy validation is error-prone and does not scale well with the complexity of modern cloud environments. Automated reasoning tools mathematically analyze access policies to identify potential security gaps, overprivileged access, and unintended permissions before they are exploited.

Desired outcome: Access policies grant only intended permissions, with no overprivileged access or unintended information exposure.

Common anti-patterns:

- Relying solely on manual code reviews to validate complex IAM policies and resource-based policies.
- Using generic policy templates without validating them against specific organizational requirements.
- Defaulting to providing over-permissive actions in policy documents.
- Not validating policies against regulatory compliance requirements before implementation.

Benefits of establishing this best practice:

- Can accelerate development cycles by catching policy issues early in the development process.
- Supports more confident policy changes through comprehensive impact analysis.
- Provides audit-ready documentation demonstrating due diligence in access control validation. You can track policy changes over time.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Implement automated policy reasoning using AWS IAM Access Analyzer and run automated policy checks with AWS CloudFormation Guard to validate access policies before deployment.

Common approaches include pre-deployment policy validation in CI/CD pipelines, automated policy drift detection, and integration with infrastructure as code (IaC) workflows.

Implementation steps

1. **Enable AWS IAM Access Analyzer**: IAM Access Analyzer provides comprehensive policy validation through mathematical reasoning, enabling you to: identify resources shared with external entities outside their zone of trust, identify internal access patterns, identify unused access, validate policies against policy grammar and AWS best practices, and validate policies using custom policy checks. The following steps outline how you can get started:
   - **Create an analyzer for your organization or account:** Enable IAM Access Analyzer in AWS console. If you are using AWS Organizations you can create a delegated administrator role in a member account. The delegated admin can then create and manage analyzers across other member accounts.
   - **Select supported resource types:** Select AWS resources to monitor. Resource types for external access detection are listed here. Resource types for internal access detection are listed here.
   - **Review initial findings:** Examine the findings dashboard to identify existing external access and internal patterns and prioritize remediation based on risk level and business requirements.
   - **Integrate with CI/CD pipelines:**
     - Use Access Analyzer APIs to validate policies during development and deployment processes.
       - **AWS CLI:** AWS accessanalyzer validate-policy
       - **AWS API:** ValidatePolicy
     - You can also use Access Analyzer APIs to:
       - Check whether the specified access isn't allowed by a policy (CheckAccessNotGranted)
       - Check whether a resource policy can grant public access to the specified resource type (CheckNoPublicAccess)
       - Preview Access Analyzer findings before deployment (Create, Get and List preview findings).
   - **Implement custom policy checks:** Create organization-specific validation rules to make sure new policies comply with your security standards and regulatory requirements. You can find examples of reference policies and learn how to set up and run policy checks for new access in the IAM Access Analyzer custom policy checks samples repository on GitHub.
2. **Integrate AWS CloudFormation Guard into CI/CD pipelines**: AWS CloudFormation Guard rules are another method of checking policy adherence. For example, you can write guard rules to:
   - Check if a policy grants wildcard access to specific services
   - Check if delete actions require multi-factor authentication (MFA) to be enabled.
   - Check if a resource-based policy (for example, an S3 bucket policy) grants public access.

- Check if a service control policy allows APIs calls beyond the allowed Regions.

3. **Validate policies locally (optional):** For development and testing purposes, you can validate and unit test CloudFormation guard rules locally before integrating them into your automated CI/CD pipeline.

   - Install CloudFormation Guard on your desktop.
   - Use the cfn-guard validate command to validate your CloudFormation templates.

   ```
   cfn-guard validate --rules rules.guard --data template.json
   ```

   - Develop and run unit test cases on your CloudFormation Guard rules. While you can validate actual CloudFormation templates using the cfn-guard validate command, unit tests go further. They assist in testing edge-case scenarios. Unit tests verify if the Guard rule is checking for the right set of property configurations. For example the below test case checks if the resource AWS::ApiGateway::RestApi has an endpoint configuration property of type PRIVATE.

   ```
   - name: MyTest4
     input:
       Resources:
         apiGw:
           Type: AWS::ApiGateway::RestApi
           Properties:
             EndpointConfiguration:
               Types: "PRIVATE"
     expectations:
       rules:
         check_rest_api_is_private: PASS
   ```

To summarize, this proactive approach reduces data breaches by identifying overprivileged access patterns, and validates policy effectiveness against AWS best practices. IAM Access Analyzer findings can also be forwarded to AWS Security Hub assisting you to build audit-ready evidence.

Resources

Related best practices:

- SEC01-BP06 Automate testing and validation of security controls in pipelines

Related documents:

- Using AWS Identity and Access Management Access Analyzer
- AWS IAM Access Analyzer pricing
- How to prioritize IAM Access Analyzer findings

Related examples:

- The AWS IAM Access Analyzer samples repository on GitHub provides examples showing how you can use AWS CLI and APIs to programmatically validate and preview policy documents.
- The AWS Guard Rules Registry is an open-source repository of AWS CloudFormation Guard rule files and managed rule sets and provides several guard rules you can use straight away.

Related videos:

- AWS re:Inforce 2024 - Refine unused access confidently with IAM Access Analyzer (IAM202-NEW)
- AWS re:Invent 2023 - Use new IAM Access Analyzer features on your journey to least privilege (SEC238)
- AWS re:Invent 2018: The Theory and Math Behind Data Privacy and Security Assurance (SEC301)
- AWS re:Invent 2025 - From Reactive to Proactive: Infrastructure governance by design (COP352)

Related services:

- AWS IAM Access Analyzer
- AWS CloudFormation Guard
- AWS Config
- AWS Security Hub

DSSEC03-BP02 Verify network security posture through automated analysis

Automated network analysis tools provide mathematical proof of network configurations. This enables you to verify adherence to data residency and privacy requirements.

Desired outcome: Maintain data sovereignty through continuous automated verification of network paths and boundaries.

Common anti-patterns:

- Relying solely on manual network configuration reviews without automated verification of actual connectivity paths.
- Assuming network boundaries are correctly configured without testing reachability scenarios.
- Not validating that network configurations block data exfiltration or unauthorized cross-border data flows.

Benefits of establishing this best practice:

- Provides mathematical proof of network security configurations and their effectiveness in blocking unauthorized access.
- Supports proactive identification of network misconfigurations before they are exploited.
- Automates complex network analysis that is typically time-intensive and error-prone if performed manually.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Implement network analysis using AWS VPC Network Access Analyzer and VPC Reachability Analyzer to mathematically verify (using automated reasoning to analyze possible network paths) network configurations and identify potential security gaps. Start by defining network access scopes that align with your data residency requirements, and data export controls. Then establish continuous monitoring for network configuration changes.

Key AWS services include:

- VPC Network Access Analyzer for identifying unintended network access patterns, and demonstrating regulatory adherence.
- VPC Reachability Analyzer for verifying connectivity between specific resources.
- Amazon Inspector for automated network reachability assessments of EC2 instances.

Common usage approaches include:

- Creating network access scopes to specify the desired connectivity between AWS resources.
- Establishing reachability tests.

- Integrating network analysis into CI/CD pipelines for infrastructure changes.

Implementation steps

1. Use VPC Network Access Analyzer to validate if traffic between a source and destination is blocked as you intended. To do this:
   - Create Network Access Scopes that specify prohibited network paths.
   - Define MatchPaths for network connections that violate compliance requirements (for example, cross-border data flows).
   - Configure ExcludePaths for legitimate exceptions to security policies.
2. Use VPC Reachability Analyzer to verify intended connectivity between resources. Reachability Analyzer analyzes the path between a source and destination by building a model of the network configuration. It does not send packets during analysis.
   - VPC Reachability Analyzer is especially useful when you are designing your network. For example, it assists you to understand the route that a connection would take if it were allowed to reach the destination. You can use this information to make sure data is encrypted in transit, or apply additional traffic filtering conditions.
   - You can also include specific intermediate components in the analysis. For example, you can analyze the path between a source and destination through a specific transit gateway. This makes it particularly valuable for security audits, policy enforcement, and compliance verification.

Automated network analysis provides mathematical verification of network security configurations, assisting you to maintain data sovereignty and block unauthorized access. VPC Network Access Analyzer and VPC Reachability Analyzer enable you to continuously verify that network paths align with your security requirements and compliance obligations.

Resources

Related best practices:

- SEC05-BP04 Automate network protection
- SEC05-BP02 Control traffic flow within your network layers

Related examples:

- Example network access scopes in Network Access Analyzer.

- Reachability analyzer source and destination resources list.

Related documents:

- VPC Network Access Analyzer User Guide
- VPC Reachability Analyzer User Guide
- Amazon Inspector Network Reachability Documentation
- AWS Security Reference Architecture
- Network Security on AWS Whitepaper
- Automated Reasoning for Network Security
- Network Monitoring and Analysis Best Practices
- An unexpected discovery

Related videos:

- AWS re:Inforce 2022 - Validate effective network access controls on AWS (NIS202)

Related services:

- VPC Network Access Analyzer
- VPC Reachability Analyzer
- Amazon Inspector
- AWS Security Hub

Detection

DSSEC04: How do you enhance threat detection capabilities for regulated workloads using intelligence and telemetry?

Organizations in highly regulated industries need strong threat detection capabilities to meet stringent monitoring and reporting requirements. Implementing relevant cyber threat intelligence and continuously validating detection coverage improves both regulatory adherence and efficient operational performance.

Best practices

- DSSEC04-BP01 Enhance threat detection through targeted intelligence
- DSSEC04-BP02 Detect threats through comprehensive telemetry and analysis

DSSEC04-BP01 Enhance threat detection through targeted intelligence

By using region and domain-specific threat intelligence, organizations can enhance their detection capabilities. This targeted approach allows security teams to contextualize threats within their specific regulatory and infrastructure environment.

Desired outcome: Achieve enhanced threat detection through contextually relevant threat intelligence that aligns with organizational risk profile and regulatory requirements.

Common anti-patterns:

- Relying solely on internally derived threat assessments.
- Lack of clear strategy for selecting, using, and governing cyber threat intelligence (CTI) information. This leads to under-performing threat detection capabilities.
- Failing to prioritize threat intelligence sources that focus on adversaries and attack patterns specific to a geographic region or regulatory environment.
- Implementing generic threat indicators without mapping them to your specific infrastructure, data classification levels, or compliance requirements.
- Failing to consider data residency and sovereignty requirements when sharing threat intelligence information. This leads to inadvertently transferring sensitive security data outside approved jurisdictions.

Benefits of establishing this best practice:

- Improved detection capabilities by shifting from reactive to proactive threat detection.
- Improved early warning capabilities by focusing on specific Indicator of Compromise (IoC) and configuration of defenses against prioritized tactics, techniques, and procedures (TTPs).
- Provides security analysts with contextual information about adversary behaviors and campaign patterns, enabling more targeted investigations.
- Facilitates information sharing with industry peers, government agencies, and security communities, creating collective defense benefits while maintaining regulatory adherence.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Review the guidelines around threat intelligence included in the applicable standards and compliance frameworks that are relevant to the organization. Evaluate available resources from cloud service providers, local cybersecurity authorities (such as NCSC, ACSC, ANSSI, and CISA), and third parties through the AWS Marketplace.

Develop supporting processes and capabilities to make sure CTI effectively improves threat detection and incident response performance. Implement monitoring and review of CTI impact on threat detection capabilities. Verify CTI provides relevant information with good accuracy and efficacy.

Implementation steps

1. **Select CTI sources and services**: Organizations have access to a diverse range of CTI services. You must carefully evaluate which services align with your requirements and risk profile. Threat intelligence provides information about potential cybersecurity threats from adversaries, enabling proactive defense strategies. Organizations typically share this intelligence in the following formats:
   - Indicator of compromise (IoCs)
   - Tactics, techniques, and procedures (TTPs)
   - High-level contextual information about potential threats. This includes targeted organizations, adversary motivations, geographic origins, and threat actor affiliations. Security leaders often share this type of information within trusted, sector-specific communities.

   CTI services are available through both commercial vendors offering premium features and open-source solutions providing free access to threat data. Assess your needs, budget constraints, and capabilities when selecting CTI services.

   Consider a cloud-based service such as Amazon GuardDuty. GuardDuty is a threat detection service that protects AWS workloads. It uses AWS's proprietary and external threat intelligence sources to safeguard your infrastructure. The service actively monitors your AWS account activities and alerts you to potential security threats. You can enable GuardDuty with no additional infrastructure or setup.

   AWS has extensive visibility into internet-wide threat patterns. This extensive reach enables AWS to detect, analyze, and block sophisticated attack techniques from various threat actors. For a detailed example of AWS's threat detection capabilities in action, see How AWS disrupts watering hole campaign by APT29.

2. **Integrate and operationalize**: To deploy your own CTI solution, refer to this AWS Prescriptive Guidance. The guide covers solution deployment, architecture patterns, and intelligence sharing best practices. Operationalize CTI capabilities across the organization, encompassing the following key components:
   - **Intelligence management:** Define standardized protocols for collecting, analyzing, and disseminating threat intelligence data. Establish clear roles, responsibilities, and authorization levels for security operations personnel.
   - **Technology integration:** Integrate automated CTI feeds into existing security infrastructure, including protective and detective controls. For example, integrate threat intelligence with SIEM solutions for real-time IoC detection.
   - **Quality assurance and governance**: Establish a governance framework to evaluate threat intelligence source accuracy, timeliness, and relevance. Implement regular reviews to assess CTI effectiveness and adjust sources accordingly.

Resources

Related best practices:

- SEC01-BP04 Stay up to date with security threats and recommendations
- SEC05-BP03 Implement inspection-based protection
- OPS01-BP04 Evaluate compliance requirements
- SEC04-BP03 Correlate and enrich security alerts

Related documents:

- AWS Prescriptive Guidance, Cyber Threat Intelligence Sharing on AWS
- Improve your security posture using Amazon threat intelligence on AWS Network Firewall
- How AWS tracks the cloud's biggest security threats and helps shut them down
- Meet MadPot, a threat intelligence tool Amazon uses to protect customers from cybercrime

Related videos:

- AWS re:Invent 2025 - Protecting Your Infrastructure with Amazon Threat Intelligence (SEC311)

DSSEC04-BP02 Detect threats through comprehensive telemetry and analysis

Effective threat detection capability needs comprehensive telemetry from multiple sources, the ability to correlate and contextualize data points, and centralized visibility of security events.

Desired outcome: Achieve effective threat detection that identifies emerging threats and supports adherence to regulatory standards through effective monitoring and analysis.

Common anti-patterns:

- Incomplete telemetry leading to significant blind spots around critical assets.
- Fragmented security data points spread across multiple tools and databases. Security analysts struggle to correlate signals from multiple sources, and may fail to detect emerging threats.
- Failing to layer preventative security controls with detective and proactive measures. Detective controls are still required for detecting potential bypass of preventative and proactive controls.

Benefits of establishing this best practice:

- Early detection of emerging threats allows you to isolate compromised systems and protect critical assets.
- Early detection supports regulatory adherence through systematic monitoring and documentation.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

1. Map compliance requirements to threat detection capabilities. In addition to mapping cybersecurity requirements to specific preventative or proactive controls, also detect threats that could compromise such controls.
2. Review the coverage, currency, and quality of telemetry involved in identifying malicious activities and emerging threats. Consider the following.
   - User and Entity Behavior Analytics (UEBA) for spotting anomalies.
   - Data loss prevention (DLP) to block unauthorized data transfers.
   - Security information and event management (SIEM) systems to correlate alerts and visualize attack paths.

3. Analyzing vast amounts of diverse telemetry data can be challenging. To identify emerging threats in a timely manner, consider the following:
   - Deploy automation to collect and store telemetry data.
   - Perform data analytics as part of your detection capability. Use extract, transform, and load (ETL) pipelines to standardize logs and trace formats. Use ETL queries to search across terabytes of data and use business intelligence or operational intelligence dashboards to visualize key metrics.
   - Use automation to handle the growing amount of data.
   - Integrate AI/ML and generative AI reasoning models to detect threats quickly from large datasets. AI/ML models and large language models (LLMs) are particularly useful for correlating logs and application traces. Evaluate these capabilities when choosing SIEM tools. However, always have a human in the loop to review findings, as the current generation of LLMs can only apply probabilistic methods to correlate information.
4. Perform regular reviews of the threat detection capabilities. The reviews should include:
   - Regulatory or compliance requirements
   - Extent of coverage
   - Detection metrics such as Mean Time to Detect (MTTD)
   - Accuracy of findings or alerts (for example, false positives versus actual findings) and the relevant configurations (for example, finding suppression rules, correlation rules)

Implementation steps

1. Conduct threat modeling to systematically identify attack paths and threats using frameworks such as STRIDE, PASTA, and LINDDUN. Evaluate frameworks such as MAESTRO (Multi-Agent Environment, Security, Threat Risk, and Outcome) to address threats emerging from the widespread adoption of generative AI models, tools, and agentic frameworks.
   - Use external data such as domain-specific threat intelligence or security community publications (for example, OWASP, AWS Security Bulletins) to improve precision.
   - Focus on actual adversary Tactics, Techniques, and Procedures (TTPs) rather than generic threats.
2. Map the output of threat modeling exercises to the detective controls and supporting telemetry data. Identify critical gaps and formulate remediation strategies. Consider other telemetry data produced by non-security sources such as system events or application logs. This data can provide insights or increase confidence when you correlate it with security logs.
3. Consider tools that support data analytic capabilities and add context to telemetry data, such as:

  - Security Analytics for Amazon OpenSearch Service that analyzes security event logs from different sources.
  - Amazon Security Lake that automatically centralizes security data from multiple sources.
  - AWS Security Hub that enables you to prioritize critical security issues by correlating different security telemetry.

4. Detect emerging threats from compliance drift. See the AWS Prescriptive Guidance for implementing detective controls on AWS.
5. Review exposures and attack paths. AWS Security Hub generates exposure findings from sources such as EC2 instances, DynamoDB tables, IAM users, S3 buckets, Lambda functions, RDS database instances, and EKS clusters. Security Hub provides a visual graph of potential attack paths, showing how attackers can take control of your resources.

Resources

Related best practices:

- SEC01-BP07 Identify threats and prioritize mitigations using a threat model
- SEC04-BP03 Correlate and enrich security alerts

Related documents:

- Detective controls
- Threat Technique Catalog for AWS
- Accelerate threat modeling with generative AI

Related videos:

- AWS re:Invent 2025 - Testing GuardDuty's Runtime Detections:Hands-on with real world attack scenarios
- AWS re:Invent 2025 - Threat-Modeling-As-Code - Transforming Your Threat Statements into Attack Trees
- AWS re:Invent 2025 - Privacy-preserving AI primitives: Building blocks for regulated industries-ARC328
- AWS re:Invent 2025 - Supercharge security investigations with custom detection & analytics (SEC350)

External publications:

- Threat Modeling: 12 Available Methods, N. Shevchenko, Carnegie Mellon University, Carnegie Mellon's Software Engineering Institute, December 3, 2018
- MITRE ATT&CK Cloud Matrix
- CISA - Insider Threat Mitigation Guide
- Insider Threat Detection Study - NATO Cooperative Cyber Defence Centre of Excellence

Infrastructure protection

DSSEC05: How do you secure infrastructure access while enabling regional operations?

Access to data and environments must be limited to locations that are trusted and meet local jurisdictional requirements. Organizations must implement comprehensive access controls, continuous monitoring, and role-based access aligned with jurisdictional boundaries to maintain sovereignty requirements.

Considerations:

Establish trust as a key component of your security posture and implement access control policies to govern infrastructure and environments. Whilst least privileged is an essential mechanism of control and only providing this when required, within a Sovereignty perspective, access to data and environments also need to be limited to locations that are trusted and meets local jurisdictional requirements.

First, identifying the authorized support staff is key, followed by the identifying or restricting locational accesses from which the staff are located. Once the support staff has been authenticated, it is then important to authorise that person for access to only the systems they need access to for the activity they are authorized for. Various systems and solutions can provide this authentication and authorisation.

Insider exploits are becoming more prevalent alongside cyber attacks through stolen credentials. To combat this, organizations should implement comprehensive access review processes and continuous monitoring of access patterns. This includes regular attestation of access rights, monitoring for unusual behavior patterns, and implementing break-glass procedures for emergency access that maintain sovereignty requirements.

Support staff access should be granted on a time-limited basis with clear documentation of the business justification. Organizations should maintain detailed records of access grants and revocations to demonstrate adherence to sovereignty requirements. Additionally, privileged access sessions should be monitored and logged, with automated alerts for suspicious activities or access attempts from unauthorized locations.

Consider implementing role-based access control (RBAC) aligned with support staff responsibilities and jurisdictional boundaries. This verifies that access permissions are consistently applied and can be quickly adjusted as staff roles or regulatory requirements change. Regular training on security awareness and sovereignty requirements should be mandatory for support staff to maintain access privileges.

Best practices

- DSSEC05-BP01 Control unauthorized remote access to infrastructure
- DSSEC05-BP02 Record operator sessions and retain logs
- DSSEC05-BP03 Empower regional teams

DSSEC05-BP01 Control unauthorized remote access to infrastructure

Implement access controls to make sure only authorized support staff from verified locations can access infrastructure resources. This includes identity verification, location validation, and continuous monitoring of access activities.

Securing access to infrastructure resources is critical for maintaining a robust digital sovereignty posture. Organizations must implement controls that verify user identity, validate location, and enforce appropriate access levels. This approach combines identity management, network controls, and automated monitoring to limit access to authorized personnel from trusted locations.

Desired outcome: Organizations maintain visibility and control over infrastructure access with detailed audit trails of support activities. Security risks are reduced through granular access controls with real-time detection and response to potential security threats. Support operations remain efficient while maintaining strict sovereignty boundaries.

Common anti-patterns:

- Granting overly broad permissions for support roles and using shared accounts instead of individual identities, violating least privilege principles.
- Relying solely on IP-based access without multi-factor authentication or just-in-time access for elevated privileges.
- Failing to regularly review and rotate access credentials while storing sensitive keys in plain text or version control systems.
- Using the same access controls across each environment without differentiating requirements and allowing direct SSH/RDP (Secure Socket Shell / Remote Desktop Protocol) access without proper monitoring.

Benefits of establishing this best practice:

- Reduced risk of unauthorized access and security breaches with faster incident detection and response capabilities.
- Automated controls and standardized processes improve operational efficiency.
- Enhanced audit trails demonstrate regulatory adherence with better visibility into infrastructure access patterns.
- Improved processes while maintaining strict security boundaries.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Organizations should assess their current infrastructure, support requirements, and compliance obligations before implementing access controls. Identify critical systems, map existing access patterns, and document regulatory requirements that influence your approach. The following practices establish secure access controls while maintaining operational efficiency.

Digital sovereignty considerations:

- When remote access is required, verify that access originates from within approved jurisdictions using secure connectivity mechanisms such as AWS Direct Connect, VPN connections, or AWS Verified Access that terminate within sovereign boundaries. This verifies that even remote infrastructure management maintains jurisdictional adherence.
- Make sure support staff operate from approved jurisdictions and use network access controls to enforce geographic restrictions.
- Combine location-based and identity-based access controls using Zero Trust Architecture principles aligned to data residency requirements.

- Maintain audit trails that demonstrate adherence to local regulations.
- Use encryption keys managed within sovereign boundaries.

Implementation steps

Identity and access management:

1. **Implement strong identity and access management:**
   - Use AWS Identity and Access Management (IAM) to create individual user accounts for support staff.
   - Assign the least privilege necessary for each role, following the principle of least privilege.
   - Regularly review and audit IAM permissions to verify they remain appropriate.
2. **Enforce multi-factor authentication (MFA):**
   - Require MFA for support staff accounts, especially those with elevated privileges.
   - Use hardware tokens or virtual MFA devices for added security.
3. **Implement just-in-time access:**
   - Use AWS IAM Access Analyzer to grant temporary, elevated permissions only when needed.
   - Implement automated processes to revoke access after a specified time or when no longer required.

Network and access controls:

1. **Implement network access controls:**
   - Use AWS Virtual Private Cloud (VPC) to create isolated network environments.
   - Implement Network Access Control Lists (NACLs) and Security Groups to restrict inbound and outbound traffic.
   - Use VPN or AWS Direct Connect for secure access from authorized locations.
2. **Use AWS Systems Manager for secure access:**
   - Use Session Manager for browser-based access to EC2 instances without the need for open inbound ports.
   - Implement fine-grained permissions for Session Manager access.
3. **Implement IP-based restrictions:**
   - Use IAM policy conditions to restrict access based on source IP addresses or ranges.
   - Regularly review and update allowed IP ranges to verify they remain current.

Governance and compliance:

1. **Use AWS Organizations:**
    - Implement a multi-account strategy to segregate environments and limit the blast radius of potential security incidents.
    - Use service control policies (SCPs) to enforce guardrails across your organization.
2. **Use AWS Config:**
    - Set up AWS Config rules to continuously monitor and assess the compliance of your AWS resources.
    - Create custom rules to enforce organization-specific security policies.
3. **Use AWS Control Tower:**
    - Implement AWS Control Tower to set up and govern a secure, multi-account AWS environment.
    - Consider using *Customizations for AWS Control Tower* (CfCT) to further customize your landing zone account structure and access controls.

Monitoring and logging:

1. **Enable and monitor AWS CloudTrail:**
    - Use CloudTrail to log API calls and management events across your AWS accounts.
    - Set up alerts for suspicious activities or unauthorized access attempts.
2. **Regular security assessments:**
    - Conduct periodic vulnerability assessments and penetration testing.
    - Use AWS Security Hub to get a centralized view of your security posture.

Data protection and key management:

1. **Implement secure key management:**
    - Use AWS Key Management Service (KMS) for centralized management of encryption keys.
    - Implement key rotation policies and access controls for sensitive data.
2. **Implement data protection measures:**
    - Use AWS Macie to discover, classify, and protect sensitive data stored in S3 buckets.
    - Implement encryption at rest and in transit for sensitive data.

Automation and DevOps security:

1. **Centralized package and dependency management:**

   - Use AWS CodeArtifact to implement centralized artifact repositories for secure package management.
   - Use Amazon Inspector for automated dependency scanning and vulnerability detection.
   - Maintain versioned and validated packages with approval workflows.
2. **Automated software deployment:**
   - Use infrastructure as code (IaC) with AWS CloudFormation, AWS CDK, or Terraform for each deployment.
   - Implement AWS CodePipeline and AWS CodeBuild for automated testing and security validation.
   - Create standardized deployment pipelines with AWS CodeDeploy for consistent releases.
3. **Reduced interactive access:**
   - Use AWS Systems Manager Automation to automate routine administrative tasks.
   - Implement AWS IAM Identity Center with temporary elevated access for just-in-time access controls.
   - Create automated approval workflows using AWS Step Functions and Amazon SNS.
4. **Automated compute protection:**
   - Deploy Amazon GuardDuty for automated threat detection and continuous monitoring.
   - Implement AWS Systems Manager Patch Manager for automated patch management across your fleet.
   - Enable AWS Security Hub for centralized security findings and automated remediation.

Training and continual improvement:

1. Provide regular security awareness training to support staff.
2. Keep the team updated on the latest AWS security features and best practices.

Resources

Related best practices:

- SEC02-BP02 Use temporary credentials
- SEC02-BP03 Store and use secrets securely
- SEC02-BP04 Rely on a centralized identity provider
- SEC03-BP01 Define access requirements
- SEC03-BP02 Grant least privilege access
- SEC03-BP07 Analyze public and cross-account access

- SEC05-BP01 Create network layers
- SEC05-BP02 Control traffic flow within your network layers
- SEC06-BP01 Perform vulnerability management
- SEC06-BP02 Provision compute from hardened images
- SEC04-BP01 Configure service and application logging
- SEC04-BP02 Capture logs, findings, and metrics in standardized locations
- SEC04-BP04 Initiate remediation for non-compliant resources

Related documents:

- AWS Security Best Practices
- AWS Identity and Access Management Best Practices
- AWS Systems Manager Session Manager

Related videos:

- AWS re:Inforce 2025 - Integrate Zero Trust into your cloud network (NIS304)
- AWS re:Invent 2025 - Innovations in Infrastructure Protection to strengthen your network (SEC310)
- AWS re:Invent 2025 - Advanced VPC design and new capabilities (NET340)
- AWS re:Inforce 2022 - Security best practices with AWS IAM (IAM201)
- Intro to Network Security - best practices for securing your network

Related services:

- AWS Identity and Access Management (IAM)
- AWS IAM Identity Center
- AWS Organizations
- AWS Control Tower
- AWS Config
- AWS CloudTrail
- AWS Security Hub
- Amazon VPC
- AWS Systems Manager
- AWS Key Management Service (KMS)

- Amazon Macie
- AWS Direct Connect
- AWS WAF

DSSEC05-BP02 Record operator sessions and retain logs

Implement session recording and log retention to maintain accountability, support security investigations, and meet regulatory requirements. This practice provides visibility into infrastructure changes and audit trails for operational activities.

Desired outcome: Organizations maintain visibility into operator actions and system changes with detailed audit trails for compliance and security investigations. Events can be reconstructed during incident response with demonstrable evidence of security controls effectiveness. A historical record of infrastructure modifications is preserved for audit and troubleshooting purposes.

Common anti-patterns:

- Failing to enable logging across regions and accounts resulting in visibility gaps. Retaining inconsistent configurations across environments.
- Storing sensitive log data without proper encryption, access controls, or immutable storage mechanisms.
- Implementing inadequate log retention periods and failing to establish automated alerting for critical security events.
- Lacking proper access controls for log storage and failing to implement centralized log aggregation strategies.
- Inadequate monitoring of operator sessions, missing suspicious or unauthorized activities.

Benefits of establishing this best practice:

- Visibility into operator activities with early detection of unauthorized or suspicious activities.
- Detailed audit trails and forensic evidence enable faster investigation and event reconstruction.
- Automated log collection and demonstrable evidence of security controls effectiveness for auditors and stakeholders.
- Detailed session logs and activity tracking improve troubleshooting capabilities.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Establish a logging strategy that addresses operational visibility and compliance requirements. Consider log retention periods, encryption requirements, and access controls to protect sensitive operational data. When implementing logging for digital sovereignty, verify logs are stored within approved jurisdictions and comply with data residency requirements, implement encryption using cryptographic keys managed within sovereign boundaries, maintain audit trails that demonstrate adherence to local data protection and privacy laws, and configure cross-border log transfer restrictions where required by regulatory frameworks.

Implementation steps

1. **Enable AWS CloudTrail across regions and accounts:**
   - Configure multi-Region logging
   - Enable log file validation
   - Implement encryption for log files
   - Set up immutable log storage using S3 Object Lock
2. **Implement session logging:**
   - Enable session logging
   - Configure session logging to Amazon S3 and CloudWatch Logs
   - Enable encrypted session data
   - Implement session log retention policies
3. **Set up Amazon CloudWatch Logs:**
   - Create log groups for different types of operator activities
   - Configure retention periods based on compliance requirements
   - Implement metric filters for important events
   - Set up cross-account log aggregation
4. **Establish a log storage strategy:**
   - Define log retention periods based on compliance requirements
   - Implement lifecycle policies for log data
   - Configure secure archive storage using S3 storage classes
   - Implement access controls for log data
5. **Configure monitoring and alerting:**
   - Set up CloudWatch alarms for suspicious activities
   - Create automated notifications for security events using Amazon SNS
   - Implement real-time log analysis with CloudWatch Logs Insights
   - Configure dashboards for log monitoring

Resources

Related best practices:

- SEC04-BP01 Configure service and application logging
- SEC04-BP02 Analyze logs, findings, and metrics centrally
- SEC04-BP04 Implement actionable security events
- SEC08-BP01 Implement secure key management
- SEC08-BP02 Enforce encryption at rest
- SEC08-BP03 Automate data at rest protection
- SEC03-BP01 Define access requirements
- SEC03-BP02 Grant least privilege access
- SEC03-BP03 Establish emergency access process
- OPS08-BP02 Analyze workload logs
- OPS08-BP04 Create actionable alerts

Related videos:

- AWS re:Invent 2022 - Cloud compliance, assurance, and auditing (COP304)
- AWS re:Inforce 2025 - Operationalizing Amazon Security Lake with analytics and generative AI (TDR342)

Related documents:

- AWS CloudTrail Best Practices
- Centralized Logging with Amazon CloudWatch

Related services:

- AWS CloudTrail
- Amazon CloudWatch
- Amazon CloudWatch Logs
- AWS Systems Manager
- Amazon S3
- AWS Key Management Service (KMS)
- Amazon SNS

- AWS Config
- AWS Security Hub
- Amazon Security Lake
- Amazon EventBridge
- AWS Lambda
- Amazon Athena

DSSEC05-BP03 Empower regional teams

Empowering regional teams while maintaining organizational consistency requires a balanced approach that combines clear governance frameworks with operational autonomy. This best practice outlines how organizations can effectively delegate authority to regional teams while maintaining alignment with global standards and security requirements.

Desired outcome: Regional teams make decisions quickly and drive innovation within their markets. Regional teams maintain clear accountability structures for compliance and regulatory adherence. Organizations maintain efficient knowledge sharing mechanisms that enable cross-region collaboration. Operational bottlenecks are reduced through distributed decision-making authority.

Common anti-patterns:

- Maintaining excessive centralized control and creating delayed decision-making processes that block regional teams from responding to regional requirements.
- Lacking clear regional authority boundaries and proper governance frameworks, creating confusion about decision-making responsibilities.
- Implementing inconsistent compliance standards across regions with insufficient cross-region communication channels, resulting in knowledge silos and duplicated efforts.
- Failing to develop adequate local expertise and effective knowledge transfer mechanisms, creating dependencies on central resources.

Benefits of establishing this best practice:

- Regional teams make decisions quickly and drive innovation through regional autonomy resulting in enhanced market responsiveness.
- Localized expertise and accountability structures support better regulatory adherence across jurisdictions.

- Reduced operational bottlenecks through distributed authority with better resource utilization and regional optimization.
- Efficient knowledge sharing and best practice distribution across regions enable cross-regional learning.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Establish a balanced governance model that delegates appropriate authority to regional teams while maintaining organizational consistency. Assess current centralization levels, define regional autonomy boundaries, and implement frameworks that enable local decision-making within global standards.

Consider digital sovereignty requirements when establishing regional governance models. Verify regional teams operate within approved jurisdictional boundaries, implement location-specific compliance and regulatory frameworks, maintain data residency requirements through regional controls, and enable regional teams to respond to jurisdiction-specific regulatory changes.

Implementation steps

1. Establish a regional authority structure:
   - Define clear scope of regional team autonomy using AWS Organizations
   - Document decision-making boundaries with AWS IAM policies
   - Create regional escalation paths
   - Implement local approval workflows using Service Catalog
   - Set up regional resource ownership with resource tagging
2. Configure regional access controls:
   - Implement region-specific IAM roles
   - Set up local administrative permissions with IAM policies
   - Configure regional service control policies
   - Enable cross-region resource sharing with AWS RAM
   - Establish regional security boundaries using AWS Organizations OUs
3. Deploy regional operations framework:
   - Create regional operations playbooks using AWS Systems Manager Documents
   - Set up local monitoring and alerting with Amazon CloudWatch
   - Implement regional incident response using AWS Systems Manager Incident Manager
   - Enable local resource management with AWS Systems Manager

  - Establish regional support structures
4. Enable knowledge sharing:
  - Create central documentation repository using Amazon S3 or AWS Wickr
  - Implement cross-region communication channels
  - Set up regular knowledge sharing sessions
  - Build regional expertise networks
  - Maintain shared best practices library with AWS Systems Manager Documents
5. Implement regional compliance controls:
  - Set up local compliance monitoring with AWS Config
  - Enable regional audit capabilities using AWS CloudTrail
  - Create local compliance reporting with AWS Audit Manager
  - Implement regional data controls using AWS KMS with regional keys
  - Establish local regulatory alignment with AWS Artifact

Resources

Related best practices:

- SEC02-BP01 Use strong sign-in mechanisms
- SEC02-BP04 Rely on a centralized identity provider
- SEC03-BP01 Define access requirements
- SEC03-BP02 Grant least privilege access
- SEC03-BP08 Share resources securely within your organization
- SEC04-BP01 Configure service and application logging
- SEC04-BP02 Capture logs, findings, and metrics in standardized locations
- OPS01-BP03 Evaluate governance requirements
- OPS02-BP01 Resources have identified owners
- OPS02-BP02 Processes and procedures have identified owners
- OPS11-BP04 Perform knowledge management

Related documents:

- AWS Organizations User Guide
- AWS Identity and Access Management Best Practices
- Organizing Your AWS Environment Using Multiple Accounts

Related videos:

- AWS Summit DC 2022 - Building and governing multi-accounts using AWS Control Tower
- AWS Summit ANZ 2022 - Security best practices the well-architected way (SEC3)

Related services:

- AWS Organizations
- AWS Identity and Access Management (IAM)
- Service Catalog
- AWS Resource Access Manager (RAM)
- AWS Control Tower
- AWS Config
- AWS CloudTrail
- AWS Audit Manager
- AWS Key Management Service (KMS)
- Amazon CloudWatch
- AWS Systems Manager
- Amazon S3
- AWS Artifact

Data protection

DSSEC06: How do you apply data sovereignty requirements?

Data sovereignty requirements are spread over multiple frameworks, industry-specific regulations, sovereign legislations, and local government policies. These requirements are jurisdiction-specific and range from residency, and handling restrictions to using specialized protection measures.

DSSEC07: How do you apply verifiable controls over sensitive data?

Having verifiable controls and comprehensive visibility over sensitive data provides tangible evidence that security measures are in place. This includes implementing data sovereignty controls, privacy-enhancing technologies, and data lineage tracking to demonstrate compliance.

DSSEC08: How do you fulfil specialised data protection requirements?

Customers operating in highly-regulated industries are often subject to specialised data protection requirements. These range from using approved encryption algorithms and devices to pre-certified confidential compute environments that protect data during processing.

Best practices

- DSSEC06-BP01 Classify data with sovereignty attributes
- DSSEC07-BP01 Establish data sovereignty controls aligning with regional data residency requirements
- DSSEC07-BP02 Provide technical options to enhance privacy
- DSSEC07-BP03 Track data lineage
- DSSEC08-BP01 Protect sensitive data during compute
- DSSEC08-BP02 Localize cryptographic operations

DSSEC06-BP01 Classify data with sovereignty attributes

Data sovereignty attributes are typically derived from storage, processing, retention, handling, geo-location, transmission and access control requirements. Attach these attributes as metadata to your datasets. When combined with data classification tags, these attributes can be utilized to enforce policy as code (PaC), build verifiable IAM policies and implement fine grained access controls. Metadata is typically expressed as labels, tags or model parameters and is attached to your resources.

Desired outcome: Data is classified and tagged with jurisdiction-specific sovereignty attributes that govern storage, processing, retention, geo-location, transmission, and access.

Common anti-patterns:

- Relying on manual processes or periodic batch jobs to discover, classify and analyse data. This leads to gaps in coverage and delayed detection of compliance violations.
- Classification and analysis rules are not reviewed and updated periodically to support new data types, regulatory changes, or evolving business requirements.
- Even when data is classified, additional data sovereignty attributes are not baselined, leading to privacy violations and financial penalties.

Benefits of establishing this best practice:

- Security controls can be calibrated to sensitivity levels, data sovereignty requirements arrived through rules-based evaluations rather than subjective opinions.
- Automated classification and analysis supports consistent application of security and privacy controls across your environment.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Data sovereignty related requirements are not typically confined to within a single compliance framework. Instead, they are spread over multiple frameworks, industry-specific regulations (such as HIPAA and PCI DSS), sovereign legislations, and local government policies. They can also materialize through amendments to existing regulations.

Consider the following while automating the discovery and qualification of data sovereignty requirements:

- Apply rules-based evaluation methods. Rules evaluate and set metadata related to data sovereignty and not just attach data sensitivity labels.
- Include a human in the loop (HTIL) step when rules are probabilistic. For example, in the case of AI/ML or large language model (LLM) based evaluations.
- Implement modular, event-driven data pipelines that automatically trigger discovery, analysis and classification workflows when new data is created or ingested. A modular design allows you to customize classification and analysis logic as regulations evolve.

Implementation steps

The implementation steps below use AWS services including Amazon Macie for automated sensitive data discovery, AWS Glue for data cataloging and extract, transform, load (ETL) operations, Amazon EventBridge for event routing, and AWS Step Functions for orchestrating data pipelines. These services integrate to create resilient, scalable pipelines and can handle diverse data sources and formats. You can also use alternative ISV tools to achieve similar outcomes.

1. **Set up a data classification model:**
   - Establish a consistent data classification model in line with the cybersecurity and privacy requirements of the jurisdiction where the data will be stored. For more detail, see data classification models and schemes.
   - Review classification models with your designated data protection officer (DPO) and data privacy experts. Having designated data steward and data owner roles may assist in streamlining this process.
2. **Map data classification levels to data sovereignty attributes:** Establish clear mapping between your classification levels and jurisdiction-sensitive data sovereignty attributes. For example, data classified as unrestricted (lowest level) would have different data sovereignty attributes than data classified as sensitive (highest level).
3. **Automatically trigger classification**:
   - Configure EventBridge to capture data lifecycle events from sources like Amazon S3, Amazon RDS, and Amazon DynamoDB.
   - Set up EventBridge rules to trigger classification and analytics workflows when new data is generated or ingested.
4. **Deploy modular classification components**: Depending on the type of data (structured, unstructured) consider using one or more of the following options.
   - Use Amazon Macie to automatically discover and classify sensitive data types including Personally Identifiable Information (PII), Protected Health Information (PHI), and other confidential information. The Macie documentation provides a detailed step-by-step walkthrough on how to configure automated sensitive data discovery. Going beyond default settings and managed data identifiers, you can also set up custom identifiers to detect sensitive data patterns unique to your workloads.
   - Implement AWS Glue crawlers to discover data schemas and populate the AWS AWS Glue Data Catalog. Then use the Glue PII detect functionality to detect PII, PHI, and other confidential data within Glue-registered tables.
   - Configure Amazon Comprehend, or Amazon Comprehend Medical for advanced text analysis and entity recognition over unstructured data.
   - Deploy AWS Lambda functions for custom classification logic specific to your industry or regulatory requirements.
   - Regardless of which method or AWS services you use, the evaluation logic must utilize the jurisdiction-aware data classification model described in the first step.

5. **Trigger post-classification enrichment jobs**:
    - Run metadata enrichment jobs to apply additional data sovereignty tags.
    - Establish a consistent tagging taxonomy in line with your data classification model and data sovereignty attributes.
    - Set up EventBridge to trigger enrichment jobs, post classification. You can then apply downstream processing rules to attach one or more data sovereignty related tags. Lambda functions, step function workflows, ECS tasks are some of the options you can use to apply these tags.
    - Apply tag keys and values consistently. Consider adopting enforceable tag policies using AWS Organizations.
    - Add as many data sovereignty tags as required.
6. **Enable monitoring and alerting**:
    - Configure AWS CloudTrail to maintain audit logs of data discovery and classification activities.
    - Use AWS Security Hub to aggregate security findings and compliance status across your data pipeline. When Amazon Macie publishes findings to AWS Security Hub, it uses the AWS Security Finding Format (ASFF). For examples, see Examples of Macie findings in AWS Security Hub.

Resources

Related best practices:

- SUS04-BP01 Implement a data classification policy
- SEC07-BP01 Understand your data classification scheme
- SEC07-BP02 Apply data protection controls based on data sensitivity
- SEC07-BP03 Automate identification and classification
- SEC07-BP04 Define scalable data lifecycle management

Related documents:

- Amazon Macie User Guide - Discovering sensitive data
- AWS Glue Developer Guide - Using crawlers to populate the Data Catalog
- Amazon EventBridge User Guide - Creating rules that react to events
- AWS Security Best Practices for Data Classification
- Detect PII data in Amazon Aurora with Amazon Comprehend
- Creating a notification workflow from sensitive data discovery with Amazon Macie, Amazon EventBridge, AWS Lambda, and Slack

Related tools:

- Apache Atlas

Related videos:

- Best practices for protecting sensitive data in AWS with Amazon Macie
- AWS Summit SF 2022 - Securing sensitive information with AWS Glue Sensitive Data Detection (ANA314)
- AWS re:Inforce 2022 - Amazon Macie for data protection and governance (TDR206)

Related services:

- Amazon Macie
- AWS Glue
- Amazon EventBridge
- AWS Step Functions
- Amazon Comprehend
- AWS Security Hub

DSSEC07-BP01 Establish data sovereignty controls aligning with regional data residency requirements

In highly regulated industries, you need to implement data sovereignty controls. These are controls designed to meet requirements related to data residency, access restrictions, and regulatory adherence. They assist in securing sensitive data while maintaining operational efficiency.

Effective data sovereignty controls provide the foundation for regulatory adherence, customer trust, and operational resilience in multi-jurisdictional environments.

Desired outcome: Sensitive data remains within designated geographic boundaries through automated, verifiable controls. You maintain adherence to regional data protection regulations and possess audit-ready evidence of data sovereignty measures. Regulatory requirements are met while operational efficiency is preserved across jurisdictions.

Common anti-patterns:

- Relying solely on contractual agreements without technical enforcement mechanisms to enforce data residency.
- Implementing data residency controls as an afterthought rather than embedding them into the architecture from the beginning.
- Applying uniform data sovereignty controls across data types without considering sensitivity levels and regulatory requirements.
- Failing to implement monitoring and alerting for data sovereignty violations.

Benefits of establishing this best practice:

- Enhanced adherence assurance through automated enforcement of data residency requirements and detailed audit trails.
- Reduced compliance costs through automated controls that reduce manual oversight and reduce audit preparation time.
- Enhanced customer trust by demonstrating verifiable commitment to data sovereignty and privacy protection.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Begin by conducting a data classification and analysis exercise. Identify sensitive data types and potential data sovereignty attributes. Implement a layered approach using AWS services. Use services that provide built-in data residency controls, automated monitoring, and audit capabilities. Deploy preventive controls to block unauthorized data movement, detective controls to monitor compliance, and responsive controls to remediate violations automatically. Choose appropriate AWS infrastructure options (for example, AWS Region, AWS Outposts, Dedicated Local Zones) to further restrict data residency.

Key AWS services for data sovereignty implementation include AWS Control Tower for organizational guardrails and AWS Config for compliance monitoring. Also use AWS CloudTrail for audit logging and AWS Organizations for policy enforcement. Use AWS KMS with customer-managed keys in specific Regions, Amazon S3 with bucket policies restricting cross-Region replication, and AWS IAM with location-based conditions. These services establish effective data sovereignty controls.

Implementation steps

1. **Understand your data residency needs**: Data privacy legislations such as European Union General Data Protection Regulation (EU GDPR) don't explicitly state requirements related to data residency or data export controls. However, many privacy legislations including EU GDPR, UK GDPR, and India's Digital Personal Data Protection Act, 2023 (DPDP), introduce a concept of adequacy. For example, under EU GDPR we have Art 45 Transfers on the basis of an adequacy decision. It states that

   A transfer of personal data to a third country or an international organisation may take place where the Commission has decided that the third country, a territory or one or more specified sectors within that third country, or the international organisation in question ensures an adequate level of protection.

   Based on these principles, the EU has established adequacy arrangements with the UK, with the U.S (under the EU-US Data Privacy Framework) and with Japan. In practical terms it means personal data can flow between these jurisdictions. But you must consult privacy and legal experts before baselining your requirements. Primarily because the type of data you may want to transfer could be subject to additional sectoral controls.

   For instance the Reserve Bank of India (RBI) has additional guidance requiring payment-related data to be stored locally. As does the Telecom Regulatory Authority of India (TRAI) for telecom related data. Furthermore, national competent authorities often provide new clarifications and exemptions. For example, the UAE's Ministry of Health and Prevention (MoHAP) issued this clarification regarding the transfer of healthcare data.
2. **Establish organizational level data residency controls**: Deploy AWS Control Tower with digital sovereignty controls enabled to create secure, compliant multi-account environments. Enable the following key controls:
   - CT.MULTISERVICE.PV.1 Region deny control to restrict operations to approved regions
   - Data residency detective controls to monitor public access and cross-region data movement
   - Encryption controls to provide data protection at rest and in transit
   - Here's an example of enabling CT.MULTISERVICE.PV.1:

```
# Enable region deny control for an OU. Allow only us-east-1 and us-west-2. But also
 add exempted actions and principals.
AWS controltower enable-control \
    --target-identifier arn:aws:organizations::01234567890:ou/o-EXAMPLE/ou-zzxx-
zzx0zzz2 \
    --control-identifier arn:aws:controltower:us-east-1::control/EXAMPLE_NAME \
    --parameters '[{"key":"AllowedRegions","value":["us-
east-1","us-west-2"]},{"key":"ExemptedPrincipalArns","value":
["arn:aws:iam::*:role/ReadOnly","arn:aws:sts::*:assumed-
role/ReadOnly/*"]},{"key":"ExemptedActions","value":
["logs:DescribeLogGroups","logs:StartQuery","logs:GetQueryResults"]}]'
```

See Region deny control applied to the OU in the AWS Control Tower documentation for more options.

3. **Deploy preventive controls aligned to individual workloads**: In addition to the region deny control which acts at an OU (Organizational Unit) level, Control Tower provides several digital sovereignty related preventative controls. These controls can be applied to protect individual workloads. Consider enabling these controls to meet workload specific compliance requirements. For example, CT.KMS.PV.6 requires that the AWS KMS customer-managed key (CMK) is configured with a key material originating from an external key store (XKS) only.
4. **Deploy additional data residency controls**: Beyond data residency controls shown in Step 1, consider applying additional controls to block cross-region networking, VPC peering, Transit Gateway peering or VPN Connections. For more options, see Data residency controls with preventive behavior.
5. **Deploy data residency detective controls**: Deploy controls to continuously monitor data sovereignty compliance. See Data residency controls with detective behavior for more options. See Detect whether Amazon S3 settings to block public access are set as true for the account to understand how such controls work.
6. **Implement key management restrictions**: AWS creates and stores KMS keys in specific AWS Regions. You cannot use a KMS key from one Region to encrypt or decrypt data in another Region. You can further restrict key usage to specific accounts and principals, thus enabling workload level isolation. Using a dedicated AWS Cloud HSM infrastructure or by using a dedicated external key store (AWS KMS XKS), you can further:
   - Restrict where keys are stored
   - Where encryption/decryption operations occur

   Additionally, with XKS, you even control the physical location and the provider of the HSM devices enabling you to perform encryption/decryption operations outside of AWS Cloud.
7. **Choose appropriate AWS Infrastructure**: The AWS Global Infrastructure provides you several infrastructure solutions and options. Those include:
   - AWS Regions: Standard multi-tenant cloud infrastructure.
   - AWS GovCloud: Dedicated infrastructure for US government workloads.

- AWS European Sovereign Cloud: Infrastructure wholly located within the European Union (EU) and supported by EU residents only.
- AWS Dedicated Local Zones: Configurable infrastructure that aligns with your data isolation, in-country data residency, and digital sovereignty needs. Can be deployed to a location you choose.
- AWS Local Zones: Run applications on AWS infrastructure closer to your end users and workloads.
- AWS Outposts: Run AWS infrastructure and services on-premises.

Develop selection criteria to choose one or a continuum of infrastructure options. Consider the following factors:

- Sensitivity and the type of the data you will be processing. How does the sensitivity or data type affect your residency requirements?
- Sovereign or sectoral mandates. As discussed in Step 1, sovereign data privacy legislations or industry regulations may demand data to be located within specific jurisdictions.
- The operational responsibilities you are willing to accept. For example, when you deploy AWS Outpost on your own managed infrastructure, you are responsible for aspects such as physical security, power supply among other things.
- Citizenship, residency, and security vetting requirements for operators supporting your workloads.
- Your technology requirements. AWS Regions, AWS GovCloud, and AWS European Sovereign Cloud offer the broadest range of AWS services. Local Zones, Dedicated Local Zone and AWS Outposts offer a different mix of AWS Services. See the resources section.
- Security services and tooling required. Here's an overview of security services and tooling available with Dedicated Local Zones.
- Interoperability and portability requirements. Several AWS Services align with their open-source counterparts. Examples include Amazon OpenSearch Service, Amazon Managed Workflows for Apache Airflow (MWAA), Amazon Keyspaces (for Apache Cassandra) (for Apache Cassandra), and Amazon ElastiCache (Redis OSS). In addition, Dedicated Local Zones, Local Zones and Outposts offer Amazon Elastic Kubernetes Service. This allows you to deploy Kubernetes workloads across the continuum from AWS Regions to AWS Outposts.
- Requirements related to survivability. AWS provides different isolation boundaries, such as Availability Zones, Regions, control planes, and data planes. Choose infrastructure aligning with your business continuity goals.
- Performance requirements. Consider your compute, storage, and networking requirements.
- Costs. Pricing for AWS Outposts racks, is different from pricing for AWS Local Zones. Use AWS Pricing Calculator to understand what works best for you.

- Flexibility. Factor in business growth and the infrastructure required to support this growth. How often will you need additional capacity? How long are you willing to wait for this capacity to come on-line?

Resources

Related best practices:

- SEC02-BP01 Use strong sign-in mechanisms
- SEC07-BP01 Understand your data classification scheme
- SEC08-BP01 Implement secure key management
- SEC08-BP02 Enforce encryption at rest
- OPS01-BP04 Evaluate compliance requirements

Related documents:

- AWS Digital Sovereignty Pledge: Control without compromise
- AWS Control Tower Digital Sovereignty Controls
- Data residency controls in AWS Control Tower
- AWS GDPR Center
- AWS Compliance Programs
- AWS Local Zones features and Services by Region
- AWS Dedicated Local Zones features - See "Your choice of services"

Related videos:

- AWS re:Invent 2025 - AWS European Sovereign Cloud: From concept to reality (SEC201)

Related services:

- AWS Control Tower
- Amazon Macie
- AWS Config
- AWS KMS
- AWS Organizations
- AWS CloudTrail

- AWS Security Hub

DSSEC07-BP02 Provide technical options to enhance privacy

Digital sovereignty requirements demand that organizations secure data and provide verifiable evidence of privacy protection measures. Technical privacy enhancement options enable organizations to process sensitive data while minimizing exposure risks. These options support business objectives and regulatory mandates through cryptographic techniques, data minimization, and privacy-preserving technologies.

Desired outcome: Sensitive data remains protected throughout its lifecycle with technical privacy controls that minimize exposure risks. Organizations maintain auditable evidence of privacy protection measures for regulatory adherence. Data utility is preserved for legitimate business operations while privacy guarantees are maintained.

Common anti-patterns:

- Relying solely on access controls without implementing data-level privacy protections. This leaves sensitive information vulnerable when permission boundaries are compromised.
- Using static masking or tokenization techniques that cannot be customized per use case, user role or data type.
- Implementing privacy controls as an afterthought rather than baking in privacy by design principles into the software development lifecycle (SDLC) activities.

Benefits of establishing this best practice:

- Advanced privacy techniques demonstrate due diligence in protecting citizen data. They assist in meeting digital sovereignty requirements across multiple jurisdictions.
- Privacy-preserving technologies enable legitimate data use cases while maintaining strong privacy protections. This supports business innovation within regulatory constraints.
- Logging and monitoring of privacy controls provide verifiable evidence for regulatory audits and compliance assessments.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Organizations should implement a layered approach to data privacy. This approach combines multiple technical controls based on data sensitivity levels and use case requirements, including encryption, tokenization, differential privacy, and secure multi-party computation techniques. Select and configure privacy technologies that align with your specific regulatory requirements. Maintain data accessibility for business operations.

Implementation steps

1. **Assess data sensitivity and privacy requirements**:
   - Conduct data mapping to identify sensitive data types including PII, PHI, financial information, and proprietary business data.
   - Define privacy requirements based on applicable regulations such as GDPR, HIPAA, CCPA, and industry-specific standards.
   - Establish data classification schemas that align with privacy protection levels and regulatory requirements.
   - Use Amazon Macie, AWS Glue PII Detect, Amazon Comprehend PII Detect to automatically discover and classify sensitive data across your AWS environment. Use Amazon CodeGuru to find potential privacy related leakages in your application code.
2. **Implement encryption and key management**:
   - Deploy AWS Key Management Service (KMS) with customer-managed keys for granular control over encryption operations.
   - Implement AWS CloudHSM for hardware-based key protection when regulatory requirements mandate usage of a single-tenant hardware security module (HSM).
   - Use AWS Certificate Manager for managing SSL/TLS certificates to encrypt data in transit.
   - Consider using External Key Stores, where at least one part of the encryption and decryption process needs to be carried outside of a cloud service provider's infrastructure. AWS provides an external key store implementation with the KMS External Key Store (XKS) feature and partners offer prebuilt external key store solutions on top.
   - Use AWS PrivateLink enabled endpoints to have resources in your VPC securely communicate with AWS Services. PrivateLink manifests as Interface and Gateway Endpoints at the edge of your VPC.
3. **Deploy tokenization, data masking, and privacy enhancements**:
   - Implement format-preserving encryption (FPE) to maintain data utility while protecting sensitive values.
   - Deploy tokenization systems that replace sensitive data with non-sensitive tokens.
   - Configure data masking based on user roles and access patterns.

- Use AWS Secrets Manager to securely store and rotate tokenization keys and masking rules.
- Deploy secure aggregation methods for statistical analysis. These methods don't expose individual data points. For example, provide only summary tables.
- Go beyond aggregation methods and implement differential privacy techniques. Differential privacy embeds random noise into the query engine. This reduces the chances of identifying a single record having sensitive information while still retaining utility of the data.

4. **Establish confidential computing environments and privacy vaults**:
   - Deploy AWS Nitro Enclaves (part of EC2) for processing highly sensitive data in isolated compute environments.
   - Implement secure multi-party computation (SMPC) protocols for collaborative data analysis without data sharing. Consider AWS Clean Rooms to collaborate with your partners without sharing raw data.
   - Set up privacy vaults. Privacy vaults store PII and PHI at a single immutable place and provide APIs for querying. Privacy vaults also play a role towards reducing data sprawl and implementing data minimization. For an example of this, see How to Scale for Global SaaS Growth with a Skyflow Data Privacy Vault on AWS.
5. **Implement privacy monitoring and compliance**:
   - Set up AWS CloudTrail to log privacy-related operations including encryption, decryption, and data access events.
   - Configure Amazon CloudWatch to monitor privacy control effectiveness and detect anomalous access patterns with intelligence threat detection capabilities offered by Amazon GuardDuty.
   - Use AWS Config to make sure privacy controls remain properly configured and compliant with organizational policies.
   - Deploy AWS Security Hub to aggregate privacy and security findings across your environment.
6. **Enable data subject rights and consent management**:
   - Implement automated data subject access request (DSAR) processing using AWS Lambda and Amazon API Gateway.
   - Deploy consent management systems and track user preferences and consent status.
7. **Implement data minimization**:
   - Archive data not in use.
     - Use Amazon S3 lifecycle policies to archive data to Glacier.
     - Use Amazon Data Lifecycle Manager to delete Amazon Elastic Block Store (Amazon EBS) snapshots no longer in use.
   - Use Amazon DynamoDB TTL to delete items that are no longer relevant. Use cases include in-session game data or similar event data that have already been materialized to another long-term storage solution (for example to S3).

- Do not copy datasets to enable individual use cases. Instead build data adapters with built-in authorization policies, and audit trails (for example HTTP APIs) to improve the accessibility of your datasets.
- Minimize proliferation of database views constructed over tables. Instead use data access policies with column, row and cell level filters to enable predefined use cases.

Resources

Related best practices:

- Data Analytics Lens - Best practice 3.1 – Privacy by Design
- SEC08-BP01 Implement secure key management
- SEC08-BP02 Enforce encryption at rest
- SEC08-BP03 Automate data at rest protection
- SEC03-BP01 Define access requirements
- SEC07-BP01 Understand your data classification scheme

Related documents:

- AWS Key Management Service Developer Guide
- Amazon Macie User Guide
- AWS Nitro Enclaves User Guide
- AWS CloudHSM User Guide
- AWS Encryption SDK Developer Guide
- AWS Secrets Manager User Guide

Related videos:

- AWS re:Inforce 2022 - Building privacy compliance on AWS (DPP101)
- AWS re:Invent 2022 - Protecting secrets, keys, and data: Cryptography for the long term (SEC403)
- AWS re:Invent 2025 - State of the Art: AWS data protection in 2025 (ft. Vanguard) (SEC203)
- AWS re:Invent 2025 - Privacy-preserving AI primitives: Building blocks for regulated industries (ARC328)
- Cryptographic Computing: Protecting Data in Use - AWS Online Tech Talks

Related services:

- AWS Key Management Service (KMS)
- AWS CloudHSM
- AWS Nitro Enclaves
- Amazon Macie
- AWS Secrets Manager
- AWS Certificate Manager

DSSEC07-BP03 Track data lineage

Organizations in highly regulated industries must implement data lineage tracking. Data lineage provides critical transparency for regulatory adherence, impact analysis, and data governance. It documents the complete journey of data from source to consumption. This capability becomes essential for demonstrating audit readiness and maintaining trust in data-driven decision making.

Desired outcome: Data movement, transformations, and dependencies are tracked automatically across data pipelines through end-to-end lineage capabilities. Organizations maintain visibility into data flows and possess audit-ready documentation of data handling practices.

Common anti-patterns:

- Manual lineage documentation that becomes outdated quickly and fails to capture real-time data transformations and dependencies.
- Siloed lineage tracking that only covers specific tools or solutions without providing end-to-end visibility across the data landscape.
- Reactive lineage capture that attempts to reconstruct data flows after issues occur rather than proactively tracking lineage during data processing.

Benefits of establishing this best practice:

- Enhanced regulatory adherence through detailed audit trails that demonstrate data handling practices and support regulatory reporting requirements.
- Improved data governance by providing visibility into data quality, transformation logic, and impact analysis for schema changes and system modifications.
- Increased data trust by enabling data consumers to understand data origins, transformation history, and quality measures before making business decisions.

- Reduced compliance costs through automated documentation and audit trail generation that removes manual effort and reduces audit preparation time.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Establish a data lineage tracking strategy. Implement automated lineage capture using AWS services. Use services that integrate with your existing data processing tools and solutions. Deploy a centralized lineage repository that can store, query, and visualize complex data relationships and integrate with governance tools.

Key AWS services for data lineage implementation include Amazon DataZone for centralized data governance and lineage visualization, and AWS Glue for automated lineage capture from extract, transform, load (ETL) processes.

Implementation steps

1. **Assess current data landscape and lineage requirements**: Conduct an inventory of your data sources, processing systems, and consumption patterns. Identify critical data flows that require lineage tracking based on regulatory requirements, business criticality, and data sensitivity levels. Document data classification levels and associated lineage requirements:
   - **Highly sensitive data**: Full column-level lineage with transformation details
   - **Regulated data**: Table-level lineage with processing metadata
   - **Internal data**: Basic flow tracking with key transformation points
2. **Deploy Amazon DataZone for centralized data governance**: Amazon DataZone provides OpenLineage-compatible features that enables you to capture and visualize data lineage. Follow the steps described in the DataZone documentation. See the resources (Related documents) section for additional technical measures you can apply.

Resources

Related best practices:

- SEC07-BP01 Understand your data classification scheme
- SEC07-BP02 Apply data protection controls based on data sensitivity
- SEC07-BP03 Automate identification and classification
- SEC07-BP04 Define scalable data lifecycle management

Related documents:

- Introducing end-to-end data lineage preview visualization in Amazon DataZone
- Building end-to-end data lineage for one-time and complex queries using Amazon Athena, Amazon Redshift, Amazon Neptune, and dbt
- Capture data lineage from dbt, Apache Airflow, and Apache Spark with Amazon SageMaker AI
- Announcing the general availability of data lineage in the next generation of Amazon SageMaker AI and Amazon DataZone
- Automate data lineage in Amazon SageMaker AI using AWS Glue crawlers supported data sources
- Build data lineage for data lakes using AWS Glue, Amazon Neptune, and Spline
- Enhance data governance through column-level lineage in Quick Suite

Related videos:

- AWS re:Invent 2024 - Empower your data journey with Amazon DataZone's data lineage (ANT207-NEW)
- AWS Summit Tel Aviv 2024 - Tracing Data from Streaming to Iceberg Lakes with OpenLineage (DEM301) - In Hebrew

Related services:

- Amazon DataZone
- Amazon SageMaker AI
- AWS Glue

DSSEC08-BP01 Protect sensitive data during compute

If you hold sensitive datasets, you might require air-gapped confidential compute environments for processing. This is in addition to at-rest and in-transit encryption.

Desired outcome: Compute environments remain isolated and hardened, blocking threat actors from gaining access to data or the compute environment. Sensitive data is protected during processing through confidential computing capabilities.

Common anti-patterns:

- Assuming it is sufficient to protect data in storage and traversing networks, and overlooking the compute environment.
- Failure to isolate ongoing compute activity from neighboring activity in shared services or on shared hardware.
- Failure to initialize compute environments to a known-clean state, compromising compute activity.
- Failure to scrub residual traces of compute activity on completion, allowing the next user of shared hardware to retrieve previous state.
- Reinventing or reimplementing security primitives for each workload, not reusing trusted implementations.

Benefits of establishing this best practice:

- Access is denied to threat actors, other AWS customers, AWS operators and personnel.
- Data is protected through its full lifecycle, removing opportunities for threat actor access during compute activity.
- Well-known and trusted encryption is applied to data in compute, blocking external or side-channel attacks from accessing plaintext data.

Level of risk exposed if this best practice is not established: Low

Implementation guidance

When you process data marked as most sensitive, you often require air-gapped environments, which are also referred to as confidential compute environments. This may be driven by motivations of protecting code and data from the operators supporting the underlying cloud infrastructure or from the customers' own operators.

Implementation steps

1. **Use confidential compute**: The AWS Nitro System forms the foundation of AWS's confidential computing capabilities. It is designed with no operator access, meaning there is no mechanism for a system or person to log in to Amazon Elastic Compute Cloud (Amazon EC2) servers, read the memory of EC2 instances, or access data stored on instance storage and encrypted Amazon Elastic Block Store (Amazon EBS) volumes. The blog Confidential computing: an AWS perspective is an introductory guide you can refer to. You can extend functionality offered by Nitro Systems with the following:

- **Isolate and offload compute to Nitro enclaves**: Nitro Enclaves extends Nitro protection by creating isolated compute environments within EC2 instances. These enclaves are separate, hardened, and highly constrained virtual machines with *no persistent storage, no interactive access, and no external networking*. The isolation is achieved through the Nitro Hypervisor, which partitions physical resources and does not implement general-purpose administrative capabilities.

  A key aspect of Nitro Enclaves is that they inherit the same isolation and side-channel mitigations as other EC2 instances running on the same server processor. When creating an enclave, you allocate a fixed number of virtual CPUs (vCPUs) and memory pages. These resources are subtracted from the parent instance and utilized by the Nitro Hypervisor to create another fully protected independent virtual machine (VM) environment.
- **Isolate compute activity in dedicated hardware**: While Nitro Enclaves provide strong isolation for compute activity within an EC2 instance, the underlying hardware is not dedicated entirely to that workload. Some data sovereignty requirements may specify that workloads run on dedicated hardware, rather than the usual EC2 multi-tenant physical server model.

  An EC2 Dedicated Host is a physical server fully dedicated for your use. Dedicated Hosts provide visibility and the option to control how you place your EC2 instances on a specific, physical server. Amazon EC2 instances placed on EC2 Dedicated Hosts use the same AWS Nitro System as described above, with the same benefits available.
- **Enable memory encryption**: For additional defense in depth against physical attacks at the memory interface level, AWS offers memory encryption on various EC2 instance types. To use this, you must select an EC2 instance type that supports memory encryption.
  - You can enable hardware-enforced memory encryption on EC2 instances that support AMD SEV-SNP. Refer to this AMD whitepaper to learn more about Secure Encrypted Virtualization (SEV) and Secure Nested Paging (SNP). In the context of EC2 instances, this has the effect of protecting the VM's memory contents from being deciphired by the hypervisor. Using mechanisms such as Reverse Map Table (RMP) protects VMs from data corruption, memory aliasing, memory re-mapping and replay related threats.
  - EC2 instances with AWS Graviton2 or later AWS Graviton processors support always-on memory encryption. The encryption keys are securely generated within the host system, do not leave the host system, and are destroyed when the host is rebooted or powered down.
  - EC2 instances with third generation Intel Xeon Scalable processors (Ice Lake), such as M6i instances, and fourth generation Intel Xeon Scalable processors (Sapphire Rapids), such as

M7i instances, support always-on memory encryption using Intel Total Memory Encryption (TME).

- EC2 instances with third generation AMD EPYC processors (Milan), such as M6a instances, and fourth generation AMD EPYC processors (Genoa), such as M7a instances, support always-on memory encryption using AMD Secure Memory Encryption (SME).

- **Use Marketplace offerings**: Evaluate your Confidentiality, Integrity and Attestation (confidentiality, integrity, and availability triad) requirements. The AWS Marketplace offers several confidential compute solutions that complement the AWS Nitro System and assists in meeting confidentiality, integrity, and availability requirements unique to your workloads.

2. **Apply secure key management practices**: Proper key management is crucial to achieve digital sovereignty goals. It is necessary to correctly apply keys in encryption activity, and to correctly manage these keys. AWS KMS provides durable, secure, and redundant storage for AWS KMS keys, and many AWS services integrate with AWS KMS to support encryption of your data.

   AWS Key Management Service (AWS KMS) uses FIPS 140-2 Level 3 validated hardware security modules to protect your encryption keys. Because KMS keys remain within AWS KMS, you must call AWS KMS to use a KMS key in a cryptographic operation. There is no mechanism to export AWS KMS keys in plain text, which keeps your sensitive cryptographic material secure. When deploying workloads using a multi-account strategy, we recommend keeping AWS KMS keys in the same account as the workload that uses them.

   Consider the full end to end journey of data and encryption keys when workloads transport data into secure enclaves for processing and then transport results back out. Neither the data nor keys should be exposed during this process. If your workload design requires that application data decryption occurs only within a secure enclave boundary, for example using keys stored outside AWS or in an AWS CloudHSM instance, apply appropriate in-transit encryption of these keys as they traverse the network.

Resources

Related best practices:

- SEC08-BP01 Implement secure key management

Related documents:

- Confidential computing: an AWS perspective

- Building zero trust generative AI applications in healthcare with AWS Nitro Enclaves
- AWS introduces Graviton5: the company's most powerful and efficient CPU
- AWS Security Reference Architecture
- AWS KMS cryptography essentials
- Data protection in Amazon EC2
- Host and Instance Features
- What is Nitro Enclaves?
- Amazon EC2 Dedicated Hosts

Related videos:

- AWS re:Invent 2025 - Deep Dive into the AWS Nitro System (CMP316)
- AWS re:Invent 2025 - Introducing Nitro Isolation Engine: Transparency through Mathematics (CMP359)
- AWS re:Invent 2020 - Deep dive on AWS Nitro Enclaves for applications running on Amazon EC2

Related services:

- Amazon Elastic Compute Cloud (EC2)
- Amazon Nitro System
- AWS Key Management Service (KMS)
- AWS CloudHSM

DSSEC08-BP02 Localize cryptographic operations

You may have specific requirements to ring-fence cryptographic operations to within specific infrastructure locations. By constructing boundaries around cryptographic operations, you can protect both your data and cryptographic keys to meet such requirements.

Desired outcome: Cryptographic operations remain within defined geographical and jurisdictional boundaries under your control. Keys, keystores, and cryptographic infrastructure are protected from access by Cloud Service Provider operators. Cryptographic functions run only in trusted infrastructure locations. You possess verifiable attestations and certifications for cryptographic operations, implementations, and the infrastructure where operations are performed. Cryptographic activities are tracked and audited within the controlled environment.

Common anti-patterns:

- Sharing cryptographic keys across multiple accounts, workloads, or geographies.
- Using cryptographic keys to which a third-party has access, or which are shared with other organizations.
- Using weak or outdated cryptographic algorithms, or using self-designed cryptographic algorithms and implementations.

Benefits of establishing this best practice:

- Underlying plaintext data remains secure and unknown to threat actors, even if the encrypted data is somehow obtained.
- Well-known and well-tested cryptographic algorithms and implementations are unlikely to have readily exploitable weaknesses.
- Trust boundary remains fully within the control of the workload owner.

Level of risk exposed if this best practice is not established: Low

Implementation guidance

Localize cryptographic operations by installing and configuring your own key management infrastructure and hardware security modules (HSMs) hosted on infrastructure that is managed by you. Build authenticated and authorized APIs for external applications (such as workloads hosted on AWS) to invoke your cryptographic infrastructure programmatically, and apply additional in-transit encryption to protect plain text data.

Data sovereignty requirements do not always require you to own and manage cryptographic infrastructure and operations. The default HSMs provided by the AWS KMS may be sufficient for your workloads. Additionally, with *customer managed keys* customers create and control the lifecycle of keys and key policies they own. For use cases with more stringent requirements, AWS KMS External Key Store (XKS) or AWS CloudHSM integration should be considered.

Implementation steps

Consider the following aspects while choosing your data-at-rest encryption strategy.

1. **Use AWS KMS Regional keys and multi-Region Keys appropriately:** AWS KMS Regional keys are regional resources that never leave their AWS Region. Cryptographic operations using these keys are performed within the Region where the key is created. AWS KMS uses FIPS 140-2 Level 3 validated hardware security modules to protect your encryption keys. Given the close

integration of AWS KMS and other AWS services, if the workload is deployed in a single AWS Region and uses KMS Regional keys, this may be sufficient to achieve sovereignty goals.

There is no mechanism to export AWS KMS keys in plain text, which keeps your sensitive cryptographic material secure. AWS KMS Multi-Region keys maintain Regional isolation for cryptographic operations, but allow the same key material to exist in multiple Regions for data replication scenarios. Consider whether this will support your data availability goals while remaining consistent with your sovereignty goals.

2. **Consider AWS KMS External Key Store (XKS) or AWS CloudHSM integration for strict sovereignty requirements:** For organizations with strict sovereignty requirements, AWS CloudHSM can be integrated with KMS through the Custom Key Store feature, providing dedicated hardware security modules (HSMs) within the chosen Region. Alternatively, the AWS KMS External Key Store (XKS) feature enables customers to use their own key management infrastructure while still using KMS APIs, maintaining control over the root of trust.
3. **Restrict access to keys and actions that use keys:** When deploying workloads using a multi-account strategy, we recommend keeping AWS KMS keys in the same account as the workload that uses them. The AWS IAM service can be used to control access to KMS keys and restrict cryptographic operations using those keys. Identity-based policies can be attached to IAM users, groups, or roles, to control their permissions to use KMS keys. Resource-based policies can be attached to KMS keys to control how the keys are used. Both identity-based and resource-based policies can be applied simultaneously. For sovereignty requirements it may be useful to add a condition to restrict access by AWS Region.
4. **Record and control encryption context:** Each AWS KMS cryptographic operation with symmetric encryption KMS keys accept an encryption context. This is an optional set of non-secret key-value pairs that act as additional authenticated data (AAD). The encryption context is not secret and not encrypted, and appears in plaintext in AWS CloudTrail Logs for auditing purposes. For example key-value pairs like "department": "10103.0" and "classification-level": "sensitive" can be used in IAM policies to refine or limit access to KMS keys in your account.
5. **Consider cost and operational burden:** Operationalizing and maintaining your own cryptographic infrastructure in the form of external key stores or self-managed HSM instances brings additional costs and skills into play. You are also required to fulfill operational requirements such as applying security updates, plus sourcing and replacing failed hardware (in the case of AWS KMS XKS) modules. A pragmatic approach could be to use such infrastructure to only protect data with higher sensitivity classification levels, or where national regulators demand such infrastructure.

Resources

Related best practices:

- SEC08-BP01 Implement secure key management

Related documents:

- Using IAM policies with AWS KMS
- Key policies in KMS
- AWS post-quantum cryptography migration plan
- Establishing a European trust service provider for the AWS European Sovereign Cloud
- How to Protect the Integrity of Your Encrypted Data by Using AWS Key Management Service and EncryptionContext

Related videos:

- AWS re:Invent 2020: Do you need an AWS KMS custom key store?
- AWS re:Invent 2022 - Protecting secrets, keys, and data: Cryptography for the long term

Related services:

- AWS Key Management Service (KMS)
- AWS KMS External Key Store (XKS)
- AWS CloudHSM

Incident Response

DSSEC09: How do you develop a cyber security incident management capability that meets regulatory and compliance standards?

Organizations in sovereign and highly regulated industries require specialized incident response capabilities that balance effectiveness with strict compliance requirements. This includes addressing data sovereignty constraints, mandatory control standards, and coordination with national cyber security authorities.

Best practices

- DSSEC09-BP01 Integrate compliance requirements into incident response

DSSEC09-BP01 Integrate compliance requirements into incident response

Organizations must embed regulatory and compliance requirements into both incident response planning and running to verify timely incident reporting, evidence preservation, and regulatory authority notification. This best practice provides guidance on how organizations meet compliance obligations while containing and remediating incidents. By aligning incident response capabilities with specific regulatory requirements, organizations demonstrate adherence during incident investigations and audits.

Desired outcome: Organizations respond to security incidents in accordance with regulatory requirements and procedures. Incidents are contained and remediated while meeting mandatory reporting obligations, reducing compliance violations.

Common anti-patterns:

- Misalignment between incident response plans and regulatory requirements, resulting in compliance violations, regulatory fines, and legal liability.
- Incident response plans become outdated due to infrequent reviews, failing to reflect changes in organizational structure, regulatory requirements, or technical architecture.
- Failure to run incident response plans during actual incidents, including delayed regulatory authority notification and inadequate incident classification.

Benefits of establishing this best practice:

- Organizations achieve compliance-ready incident response capabilities, meeting regulatory obligations and procedures, and mitigating compliance violations.
- Regular incident response plan validation verifies that incident response capabilities remain aligned with evolving regulatory requirements, reducing regulatory risks and fines.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Compliance requirements related to incident reporting include the following:

- **Reporting incidents to National Competent Authorities**: Report incidents to a designated authority (such as the Information Commissioner's Office (ICO) or National Cyber Security Centre (NCSC) or Data Protection Authority (DPA) within specific timelines. Consider the following aspects.
  - *Reporting timeframes*: Timeframe within which information must be shared with authorities and affected stakeholders.
  - *Impact to stakeholders*: Share information regarding the incident, its impact, and steps taken to minimize the impact.
  - *Obligations related to locating root cause(s)*: Conduct and report results of a root cause analysis (RCA) within specific timeframes.
  - *Communication protocols*: The contact persons or authorities involved in the incident reporting and the method to share the information.
- **Validation of incident handling procedures**: Perform periodic reviews (attestation, certification) of your organization's incident handling procedures. Validations may include checking for specific capabilities such as digital forensics, or usage of specific tools and services.

To meet compliance standards around incident reporting, consider the following.

- **Include regulatory adherence requirements in your incident response plan**: This includes:
  - Clear roles and responsibilities for cybersecurity incident management.
  - Cybersecurity incident classification aligned with regulatory and compliance standards. The classification determines relevant processes including reporting requirements, communication protocols, and information sharing timelines.
  - Internal communication and escalation protocols. Including incident triage, internal reporting and case handling procedures.
  - External communication plan. Including authority notification processes, regulatory reporting timelines, and information sharing procedures.
  - Cyber resilience capabilities aligned with business continuity (BC) and disaster recovery (DR) plans to maintain critical services during cybersecurity incidents.
  - Digital forensics capabilities that meet regulatory reporting needs. These capabilities perform thorough assessment, threat neutralization, and evidence extraction from compromised systems.
- **Develop and maintain the incident response capabilities across**:

  - **People**: Executive sponsors, security analysts, cyber incident response team (CIRT), security engineers, legal, Human Resources (HR). Use cross-functional teams to assess incident impact and provide response plan inputs.
  - **Process**: Incident response plan, playbook, runbooks, DR or BC plan
  - **Technology or tool**: Security information and event management (SIEM) tools, security controls, cloud operation tools
- **Regularly review incident response plans**: Update plans to reflect organizational changes in people, processes, and technology. Conduct validation through tabletop Exercise (TTX), purple or red teaming. Post-incident reviews can also provide inputs for incident response plan improvements.

Implementation steps

1. Identify and clarify applicable compliance standards and requirements. Determine which regulatory frameworks apply to your organization (such as GDPR, HIPAA, or PCI-DSS). Document the specific incident response capabilities requirements for each applicable standard, including notification timelines, reporting procedures, evidence preservation requirements, and communication protocols.
2. Develop and validate the incident response plan aligned with compliance requirements. Refer to AWS Prescriptive Guidance on Security recommendations for responding to incidents. The document emphasizes that successful incident response requires three key foundations - preparation, operations, and post-incident activity - and recommends establishing a well-defined incident response plan, creating runbooks and playbooks, implementing event-driven security automation, documenting support engagement processes, and configuring alerts for security events to verify that organizations can effectively detect, respond to, and remediate security incidents in the cloud.
3. Implement incident response capabilities. Refer to the AWS Security Incident Response Technical Guide for an overview of responding to incidents within AWS environment. This guide provides step-by-step procedures for the three key phases: preparation (establishing incident response plans and automated responses), operations (detection, analysis, containment, eradication, and recovery), and post-incident activity (lessons learned and process improvements).
4. Validate compliance alignment. Conduct tabletop exercises or simulations to validate that your incident response procedures meet the specific requirements of each applicable compliance standard.

Resources

Related best practices:

- SEC10 How do you anticipate, respond to, and recover from incidents?
- SEC01-BP03 Identify and validate control objectives
- OPS01-BP04 Evaluate compliance requirements
- SEC01-BP08 Evaluate and implement new security services and features regularly

Related documents:

- AWS Security recommendations for responding to incidents
- AWS Security Incident Response Technical Guide
- Threat Technique Catalog for AWS

Related videos:

- AWS re:Invent 2025 - AWS detection and response innovations that drive security outcomes (SEC323)
- AWS re:Invent 2025 - Accelerating incident response through AIOps (COP334)
- AWS re:Invent 2025 - The incident is over: Now what? (COP216)

Reliability

The reliability pillar focuses on verifying that workloads perform their intended functions correctly and consistently within sovereign boundaries. In highly regulated industries, reliability extends beyond traditional availability and recovery to include maintaining regulatory adherence during failures, respecting jurisdictional constraints during disaster recovery, and keeping audit trails intact throughout each operational state.

Topics

Definitions

The following are reliability-specific definitions:

- **Cross-region replication:** An automatic, asynchronous copying of data from a source location in one geographic region to a destination in a different geographic region.
- **Regulatory boundaries:** Legal, geographic, or jurisdictional limits imposed by laws, regulations, and compliance requirements that dictate where data can be stored, how it must be processed, and who can access it.
- **Jurisdictional boundaries:** Legal and geographic limits that define where specific laws, regulations, and governmental authority apply. They determine which legal system has the power to enforce rules, adjudicate disputes, and govern activities within a particular territory.
- **Sovereign boundaries:** The geographic borders that define the territorial limits of a nation-state's supreme authority and legal control, within which that government has exclusive power to create and enforce laws without external interference. Delineate the physical territory over which a nation exercises complete and independent governmental authority, including control over people, resources, data, and legal matters within those borders.

- **Regional isolation:** the practice of keeping data, systems, and resources physically and logically separated within specific geographic regions to verify that they operate independently without cross-region dependencies. This approach blocks failures, security breaches, or regulatory violations in one region from affecting operations in other regions.
- **ICT Business Continuity Management (BCM):** Specific compliance area under sovereignty regulations (such as EU DORA Chapter IV) requiring documented resilience and recovery capabilities that respect jurisdictional boundaries.
- **Data replication:** Copying data between sites while respecting jurisdictional boundaries to keep replicas within approved regions. Must be carefully designed to avoid inadvertent cross-border transfers.
- **Data perimeters:** The technical implementation of sovereignty requirements that block unauthorized cross-border data flows that define and enforce where data can reside, how it can be accessed, and who can interact with it based on attributes like location, identity, network, and resource type. They establish explicit controls to block unauthorized data access, movement, or processing outside defined geographic, organizational, or regulatory boundaries.
- **Sovereign-compliant failure prevention:** Resilience strategies designed to maintain compliance during incidents by keeping data and operations within approved boundaries during failures. Integrates sovereignty requirements into resilience planning from the design phase.
- **Hub-and-spoke (in sovereignty context):** Architectural pattern with centralized core services (hub) and regionally-isolated implementations (spokes) that balances global consistency with local sovereignty requirements while maintaining compliance boundaries. Network architecture where a central hub location manages and controls resources, with multiple spoke regions connecting to it for services, creating potential sovereignty issues when data must transit through the hub's jurisdiction.
- **Fault isolation:** The practice of containing failures within defined boundaries to block them from cascading across a system. It uses techniques like bulkheads, cell-based architectures, and failure domains to verify that when one component fails, the impact is limited and other parts of the system continue operating normally. This approach minimizes blast radius and improves overall system resilience by treating failures as inevitable and designing explicit containment strategies.
- **Foreign:** Belonging to, located in, or coming from another country or external source.

Design principles

- **Embed sovereignty into recovery architecture:** Design automated recovery procedures that respect jurisdictional boundaries by default. Verify that data and operations remain within approved regions during both normal operations and failure scenarios. This approach removes the need for manual compliance verification.

- **Separate regional concerns from core solutions:** Establish clear architectural boundaries between region-agnostic core services and region-specific implementations. This enables independent regional operations while maintaining consistent security and governance standards across jurisdictions.
- **Build for portability and interoperability:** Design workloads using infrastructure as code, containerization, and standardized APIs. This enables seamless deployment and failover across compliant regions while avoiding vendor lock-in and maintaining data sovereignty.
- **Maintain continuous visibility and auditability:** Implement comprehensive monitoring, logging, and audit trails that operate within sovereign boundaries. This provides real-time visibility into system health and compliance status during both normal operations and failure events.
- **Plan for independence and resilience:** Document critical dependencies and establish manual contingency procedures. Design systems that can operate independently when technology systems or third-party services become unavailable. This verifies business continuity under each failure scenario.

Foundations

DSREL01: How do you plan for survivability?

Comprehensive survivability planning addresses cyber attacks, natural disasters, and infrastructure failures while maintaining regulatory adherence. This planning is fundamental to customer trust, meeting service level agreements, and verifying long-term organizational viability in complex, regulated environments.

DSREL02: How do you reduce vendor risks and costs?

Vendors can inadvertently introduce vulnerabilities into an organization's environment, potentially leading to data breaches, compliance violations, and significant financial and reputational damage.

By having an oversight of and managing vendor risks, companies can strengthen their overall security posture, adhere to regulatory requirements, and optimize costs associated with incident response and remediation.

Best practices

- DSREL01-BP01 Document and rank critical business continuity risks
- DSREL01-BP02 Develop mitigation plans for critical risks
- DSREL01-BP03 Document recovery procedures
- DSREL01-BP04 Select and operationalize recovery sites
- DSREL01-BP05 Establish independent localized operations
- DSREL01-BP06 Train team members on recovery procedures
- DSREL01-BP07 Create manual backup plans for system outages
- DSREL02-BP01 Implement continuous third-party risk management (TPRM) processes

DSREL01-BP01 Document and rank critical business continuity risks

Documenting and ranking critical business continuity risks enables you to proactively identify potential service disruptions and allocate resources effectively. You should also consider compliance obligations while ranking risks.

Without a structured approach, you risk regulatory penalties and compliance violations. You may also face operational disruptions and damage to customer trust.

Desired outcome: A prioritized risk register documents critical business continuity risks, aligned mitigation strategies, and recovery procedures. Organizations maintain effective compliance management and operational resilience through systematic risk tracking and mitigation.

Common anti-patterns:

- Conducting risk assessments without cross-functional input, missing critical dependencies and creating incomplete risk profiles.
- Treating risk documentation as one-time deliverables instead of living artifacts that evolve with changing business conditions.
- Lacking data-driven quantitative risk metrics and clear recovery objectives, making it difficult to prioritize investments.
- Limiting risk assessment to technical or compliance aspects only, ignoring business impact and operational dependencies.

- Conducting infrequent testing and validation of recovery procedures, leading to outdated plans and unverified assumptions.

Benefits of establishing this best practice:

- Systematic risk management documentation supports regulatory adherence. Stakeholder confidence is built by demonstrating preparedness and providing transparency to customers, partners, and regulators.
- Uses data-driven decision making to focus investments on highest-impact risks and avoid over-engineering low-impact areas.
- Accelerates incident response and reduces downtime through proactive identification of vulnerabilities.
- Provides systematic documentation that demonstrates due diligence to auditors and enables effective compliance management across the organization.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Start by assembling a cross-functional team, including members from IT, security, compliance, legal, and business units. Together, establish a standardized risk assessment framework that aligns with regulatory requirements. Create a centralized risk register. Use the register to document critical workloads, track dependencies, record compliance obligations, and prioritize risks based on likelihood and impact. Include regular updates and validation through testing to maintain effectiveness and regulatory adherence.

Many sovereign nations are having to deal with an ever-increasing number of cyber-attacks and acts of sabotage. As a mitigation, nations are putting forward new regulatory requirements related to cyber-resiliency with the objective of protecting critical national infrastructure, and strategically important public services.

New regulations such as the EU Digital Operational Resilience Act (EU DORA) applies to a broad set of financial entities and requires them to bolster their cybersecurity and ICT operational resiliency capabilities. It is implemented on three levels. On Level 1 - Regulation and amending Directive, Section II, Article 6 explicitly calls out for a sound, comprehensive, well-documented ICT risk management framework.

Implementation steps

1. **Identify and document risks**: Conduct business impact analysis (BIA) to identify critical workloads and their dependencies. BIA should uncover the following:
   - **Technical risks:** Infrastructure failures, security breaches, data loss
   - **Operational risks:** Process failures, human error, third-party dependencies
   - **Compliance risks:** Regulatory changes, audit findings, policy violations
2. **Prioritize risks**:
   - Score each risk by likelihood and impact (typically 1-5 scale)
   - Calculate risk priority: Likelihood × impact = risk score
   - Consider:
     - Financial impact (revenue loss, fines, remediation costs)
     - Operational impact (downtime, service degradation)
     - Reputational impact (customer trust, brand damage)
     - Compliance impact (regulatory penalties, legal exposure)
3. **Build mitigation strategies**: For each risk, document the following:
   - **Preventive controls:** What stops the risk from occurring (automation, redundancy, monitoring)
   - **Detective controls:** How you'll detect if it happens (alarms, audits, logs)
   - **Remediation measures:** How you'll respond (runbooks, failover procedures)
   - **Owner:** Who's accountable for managing this risk
   - **Timeline:** By when will the risk be fully mitigated
   - **Impact:** What impact will it have on systems and operations if the strategy were to be implemented
4. **Track risks over time**:
   - Use a centralized tool (for example, collaborative trackers)
   - Schedule regular reviews (monthly for high risks, quarterly for medium or low)
   - Track metrics: open risks, overdue mitigations, risk trend over time
   - Integrate with change management (assess new risks when deploying changes)
5. **Create recovery procedures**:
   - Document step-by-step recovery runbooks
   - Establish teams and roles related to incident management
   - Establish escalation paths and communication plans
   - Map system dependencies that are part of your recovery path
   - Define recovery metrics for each critical workload
6. **Maintain operational resilience**:

- Test recovery procedures regularly (tabletop exercises, DR drills using AWS GameDays, AWS chaos engineering tools, AWS Skills Builder). This facilitates discovery of previously unknown risks.
- Update risk register after incidents (lessons learned)
- Conduct periodic risk assessments (at least annually)

The following is a example of what a risk register could look like:

Risk ID	Description	Category	Likelihood	Impact	Priority	Mitigation strategy	Owner	Status	Review date
R-001	Primary Region failure	Technica	2	5	10	Multi-Region DR	Ops Lead	Active	Monthly

Additional fields may include:

1. Expected date on which the mitigation strategy will be reviewed.
2. Estimated cost of maintaining the as-is state without mitigating the risk.
3. The impact of activating a given mitigation strategy.

The key is making risk registers a continual process rather than a one-off document. Integrate risk review into your regular operational cadence, and verify that high-priority risks have an owner.

Resources

Related best practices:

- OPS01-BP06 Evaluate tradoffs while managing benefits and risks
- OPS10-BP03 Prioritize operational events based on business impact
- OPS01-BP05 Evaluate threat landscape

Related documents:

- Risk Management

Related videos:

- [AWS re:Invent 2023: Backup and Disaster Recovery Strategies for Increased Resilience: Leveraging AWS Services for Cost-Effective Business Continuity (ARC208)]

DSREL01-BP02 Develop mitigation plans for critical risks

Proactive risk management is essential for maintaining business continuity, regulatory adherence, and stakeholder trust. You need a structured approach to manage critical risks. This assists in reducing potential financial penalties, operational disruptions, and reputational damage.

Your strategy should include robust mitigation plans. These plans should demonstrate due diligence, protect sensitive data, and enable effective incident response in cloud environments. With this systematic approach to risk management, you can better safeguard operations, adhere to regulatory requirements, and improve trust with customers and stakeholders.

Desired outcome: Organizations maintain prioritized, tested mitigation plans that enable rapid response to critical risks while maintaining regulatory adherence. Potential disruptions are reduced to acceptable levels through actionable mitigation strategies. Business continuity and regulatory adherence are sustained even during adverse events.

Common anti-patterns:

- Adopting a reactive approach that addresses risks only after incidents occur, missing opportunities for prevention.
- Operating in silos without cross-functional input, leading to fragmented risk ownership and uncoordinated mitigation efforts.
- Maintaining static, generic documentation that fails to reflect current AWS services or specific organizational risks.
- Conducting incomplete risk assessments that overlook critical areas such as compliance, operational risks, and third-party dependencies.
- Neglecting to validate mitigation strategies through regular testing, simulations, or tabletop exercises.
- Relying on manual processes and unclear communication protocols, hindering effective incident response and stakeholder notification.

Benefits of establishing this best practice:

- Supports faster incident response and system recovery through pre-planned, documented mitigation strategies.
- Shows proactive risk management to auditors, regulators, and stakeholders. Supports adherence to frameworks like GDPR, HIPAA, and PCI-DSS.
- Enhances operational resilience and business continuity through improved system reliability and proactive risk management.
- Preserves institutional knowledge and ensures consistent response procedures regardless of personnel changes.
- Creates a culture of operational excellence through regular testing, updates, and continuous improvement.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Consider building mitigation plans by prioritizing high-risk items in your risk register. Create a cross-functional team that includes security, compliance, operations, and business stakeholders. Use AWS tools and industry frameworks to identify, prioritize, and design mitigations that align with AWS best practices and regulatory requirements.

Digital Sovereignty considerations: Include aspects related to data residency, data privacy, and operator access restrictions when designing your mitigation plans. Your recovery path must not result in compliance violations.

Implementation steps

Building a technical mitigation plan involves: understanding your application components, their runtime characteristics, the underlying fault isolation boundaries and their inter-dependencies. Several AWS services support this process:

1. Use AWS Audit Manager (Audit Manager) to discover potential reliability risks and mitigations. Audit Manager common controls lists clear mitigation steps for commonly encountered high availability (HA) scenarios. A common control is a guideline that's not specific to one framework or AWS resource. Instead, it maps to domains such as HA and data protection.
2. Use AWS Systems Manager Application Manager and resource groups to build a unified view of your applications and their underlying AWS resources.
3. Use AWS Resilience Hub to define your resilience goals, assess your resilience posture against those goals, and implement recommendations for improvement based on the AWS Well-Architected Framework.

4. Map service dependencies using AWS X-Ray and trace requests through distributed applications. Use X-Ray to discover how microservices and components interact at runtime, especially for serverless and containerized workloads.
5. Implement compliance and preventive controls. Use AWS Control Tower to set preventive, proactive and detective guardrails. When you enable a control in Control Tower, it also shows you the compliance framework that the control maps to.
6. Automate your recovery using Systems Manager Automation runbooks, Amazon Q Developer for monitoring and troubleshooting AWS resources in communication applications, and AWS Lambda to run one-off recovery tasks.
7. Implement continuous improvement and effective reporting with Quick Suite.

Resources

Related best practices:

- OPS01-BP04 Evaluate compliance requirements
- SEC01-BP07 Identify threats and prioritize mitigations using a threat model
- SEC10-BP02 Develop incident management plans
- OPS07-BP05 Make informed decisions to deploy systems and changes

Related documents:

- Compliance validation for AWS Cloud Map
- Use AWS Chatbot in Slack to remediate security findings from AWS Security Hub

Related videos:

- AWS re:Inforce 2025 - Best practices for managing governance, risk and compliance globally (GRC301)

Related services:

- Quick Suite
- AWS Audit Manager
- AWS Chatbot
- AWS Compliance Programs
- AWS Config

- AWS Control Tower
- AWS IAM
- AWS Organizations
- AWS Resource Groups
- AWS Security Hub
- AWS Systems Manager

DSREL01-BP03 Document recovery procedures

Document your disaster recovery procedures to enable operations to recover quickly, while maintaining your regulatory posture. Include steps to recover critical systems and data, meet recovery timelines, and establish clear lines of accountability. Keep your documentation up to date, simple to follow, and ready for audits.

Desired outcome: Organizations maintain current, tested recovery documentation that enables rapid system restoration. Recovery procedures consistently meet regulatory requirements and achieve defined RTOs and RPOs. Teams run efficient recovery operations during disruptions without confusion or delays.

Common anti-patterns:

- Maintaining outdated or untested recovery procedures stored in single locations without version control or regular updates after infrastructure changes.
- Failing to define clear roles, responsibilities, and system interdependencies in recovery plans, leading to confusion during actual recovery operations.
- Using generic, manual recovery procedures instead of automated workflows tailored to specific system requirements and compliance needs.
- Missing essential compliance checks and audit requirements in recovery documentation, creating regulatory gaps during recovery operations.

Benefits of establishing this best practice:

- Accelerates recovery operations through clear, step-by-step automated procedures that anyone can follow consistently regardless of who performs them.
- Supports regulatory adherence and audit readiness with well-documented procedures that include essential compliance checks and reporting capabilities.
- Reduces data loss and recovery time through automated processes that achieve defined RTOs and RPOs consistently.

- Maintains current, effective procedures through version control and regular updates, allowing teams to identify and address weaknesses proactively.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

To document your recovery procedures:

- Start with your most critical systems.
- Create clear steps that anyone can follow.
- Define who does what.
- Add automated steps where possible.
- Include compliance requirements.

Key implementation elements:

- Define RTOs and RPOs.
- Map dependencies between applications, data stores, and services.
- Use infrastructure as code (IaC) to automate environment rebuilds.
- Regularly review and update documentation after infrastructure changes.

Implementation steps

1. Define and document recovery objectives for each system using AWS Resilience Hub (Resilience Hub). Use this service to set recovery targets, track compliance requirements, and monitor resilience metrics.
2. Create a system inventory and dependency maps using AWS Systems Manager and resource groups.
3. Establish a recovery team structure with defined roles and communication procedures, including clear contact information and escalation paths for effective incident response.
4. Implement infrastructure as code (IaC) using AWS CloudFormation (CloudFormation) and AWS Cloud Development Kit (AWS CDK) (CDK) to create templates for automated recovery.
5. Document recovery procedures in Systems Manager documents. Configure backup and recovery workflows using AWS Backup for system backups and enable Amazon S3 versioning for data protection.

6. Set up automated recovery processes using AWS Auto Scaling for capacity management and Amazon EventBridge with AWS Lambda for automation. Configure monitoring with Amazon CloudWatch and audit logging with AWS CloudTrail.

Resources

Related best practices:

- REL13-BP05 Automate recovery

Related documents:

- AWS Disaster Recovery Documentation
- AWS Resilience Hub User Guide

Related videos:

- Backup and Disaster Recovery Strategies for Increased Resilience: Leveraging AWS Services for Cost-Effective Business Continuity

Related services:

- Amazon CloudWatch
- Amazon EventBridge
- Amazon Simple Storage Service (S3)
- AWS Auto Scaling
- AWS Backup
- AWS CDK
- AWS CloudFormation
- AWS CloudTrail
- AWS Resilience Hub
- AWS Resource Groups
- AWS Systems Manager

DSREL01-BP04 Select and operationalize recovery sites

Select and set up recovery sites that meet your data and operational sovereignty requirements, especially if you operate in regulated industries. Your recovery plans should maintain data and workloads in approved locations, meet industry regulations, and follow your organizational security policies.

Desired outcome: Organizations maintain disaster recovery sites that preserve data sovereignty and regulatory adherence. Recovery sites enable rapid restoration when needed and function reliably during normal operations and failover scenarios. Data and workloads remain within approved jurisdictions throughout the recovery processes.

Common anti-patterns:

- Automatically replicating data to nearest Regions without considering data sovereignty implications, treating data uniformly regardless of sensitivity levels, and failing to document data flows during normal and disaster scenarios.
- Moving data across jurisdictions without proper compliance checks or verification that recovery sites meet applicable regulations.
- Depending on manual, error-prone recovery procedures instead of automated, tested processes that maintain consistent compliance.

Benefits of establishing this best practice:

- Supports quick and reliable disaster recovery through clear, consistent, and automated recovery steps that work across each scenario.
- Builds customer trust and competitive advantage by demonstrating commitment to data sovereignty, compliance requirements, and operational excellence.
- Maintains data and workloads within approved jurisdictions throughout the recovery processes, fostering regulatory adherence even during disruptions.
- Reduces risk of regulatory penalties and improves audit readiness.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Select AWS Regions that meet your needs for disaster recovery and automate your recovery steps. Agree with your business stakeholders as to what constitutes minimum service levels for a given workload. Perform validations (for example, data integrity checks, connectivity checks, data flows, and latency) after completion of every stage to check if minimum service levels are achieved.

Your recovery procedure should also document full recovery timelines, and how stakeholders are to be kept informed during every stage until full recovery is achieved.

Digital sovereignty considerations: To set up compliant recovery sites, start by classifying data, including data type and sensitivity levels. Understand which data privacy legislations and cybersecurity standards are applicable to you. Check where data can be stored. Check where your operational recovery teams are located and what access they have. Work with legal and compliance teams to establish clear policies for cross-border data movement.

Key implementation elements:

- Assess regulatory requirements (for example, GDPR, country-specific data privacy legislations and directives)
- Select AWS Regions that meet residency and compliance needs
- Apply data residency controls (for example AWS Control Tower Service Control Policies, AWS Network Firewall rules, AWS Transit Gateway routes) to manage traffic during disaster recovery
- Automate deployment of workloads to recovery sites using infrastructure as code (IaC)
- Encrypt data using keys managed in compliant Regions (for example, AWS KMS with restricted key policies)
- Regularly test failover to validate adherence and functionality

Implementation steps

1. Assess regulatory requirements:
   1. List data protection and data privacy requirements that apply to your business.
   2. Document industry-specific regulations you must follow.
   3. Define recovery timelines for each system.
   4. Map which data types can be stored in which locations.
   5. Get sign-off from legal and compliance teams.
2. Select and validate AWS Regions for recovery:
   1. Review AWS Regional Services to verify service availability in your chosen AWS Regions.
   2. Use AWS Artifact to check compliance certifications and attestations for each AWS Region.
   3. Test the network latency between primary and recovery sites.
3. Configure encryption and key management using AWS Key Management Service (AWS KMS) and AWS CloudHSM:

    1. Create Region-specific encryption keys.
    2. Set up key policies that enforce data sovereignty.
    3. Document key management procedures.
4. Design and configure the recovery network using Amazon Virtual Private Cloud (Amazon VPC), AWS Transit Gateway, and AWS Direct Connect:
    1. Create isolated network segments with VPC.
    2. Set up security controls and define routing between Regions.
    3. Configure Transit Gateway to connect VPCs and on-premises networks.
    4. Control cross-Region traffic and enforce network security policies.
    5. Set up Direct Connect for dedicated connection to AWS with consistent network performance and secure data transfer.
5. Automate your recovery infrastructure using AWS CloudFormation (CloudFormation) or AWS Cloud Development Kit (AWS CDK) (CDK):
    1. Create infrastructure templates.
    2. Validate your infrastructure templates with CloudFormation cfn-validate and cfn-test.
    3. Automate recovery steps. Consider using AWS Elastic Disaster Recovery, AWS Step Functions, Amazon EventBridge, AWS Lambda functions to orchestrate recovery flows.
    4. Run regular recovery drills to test automation.
    5. Validate compliance requirements during tests.
    6. Document test results and update procedures based on findings.

Resources

Related best practices:

- Encryption best practices for AWS Key Management Service
- REL13-BP02 Use defined recovery strategies to meet the recovery objectives
- ADVREL03-BP02 Choose AWS Regions that meet your legal and disaster recovery requirements

Related documents:

- AWS Regions
- Encryption best practices for AWS Key Management Service

Related videos:

- AWS re:Inforce 2025-Navigating sovereignty requirements: Architectures and solutions on AWS (DAP202)

Related services:

- Amazon VPC
- AWS CDK
- AWS CloudFormation
- AWS CloudHSM
- AWS Direct Connect
- AWS KMS
- AWS Regional Services
- AWS Resource Groups
- AWS Systems Manager
- AWS Transit Gateway
- AWS Elastic Disaster Recovery

DSREL01-BP05 Establish independent localized operations

Organizations in highly regulated industries must establish independent localized operations across different geographical regions. This maintains adherence to data sovereignty requirements and regulatory mandates specific to each jurisdiction.

Localized operations enable autonomous workload management within specific regions. You can enforce local data residency requirements and comply with regional regulations such as GDPR, HIPAA, or CCPA. This approach assists in managing cross-border risks while maintaining operational agility and meeting the needs of local users and regulatory authorities.

Desired outcome: Organizations maintain autonomous Regional operations with isolated data processing, storage, and governance capabilities. Local regulations and data residency requirements are met through Region-specific compliance controls. Security standards and operational efficiency remain consistent across each jurisdiction.

Common anti-patterns:

- Operating from a single central AWS Region without considering data sovereignty requirements, using shared resources across jurisdictions, and failing to properly segment networks and isolate regional operations.

- Implementing centralized, uniform compliance policies and IAM controls without accounting for local regulatory variations or region-specific requirements.
- Not implementing proper data classification and handling procedures based on local sensitivity requirements, storing regulated and non-regulated data in shared systems without proper separation.
- Using global edge services and CDNs without appropriate geo-restrictions, regional controls, and region-specific encryption controls.

Benefits of establishing this best practice:

- Regulatory requirements are met through adherence to local data protection laws and industry-specific regulations while enabling Region-specific audit trails.
- Strengthens data sovereignty by maintaining complete control over data location and processing boundaries, minimizing exposure to cross-region breaches and legal penalties.
- Enhances operational resilience through independent regional operations that can continue functioning during disruptions without impacting other Regions.
- Improves performance and user experience by reducing latency through geographically distributed operations that keep data closer to users.
- Simplifies compliance reporting and auditing through clear regional separation and jurisdiction-specific documentation.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Establish independent localized operations through a strategic, multi-phase approach. Start with a comprehensive regulatory assessment to understand specific requirements for each localized Region and operation. Document data residency mandates, processing restrictions, and compliance frameworks specific to each local jurisdiction.

Design your architecture with clear Regional boundaries. Use separate AWS accounts and dedicated resources for each jurisdiction. This provides complete operational, security, and billing isolation.

Implement Region-specific governance policies and operational procedures that align with local requirements. Maintain your overall security posture and operational excellence across each Region. Support your implementation with local teams who understand both the technical and regulatory landscape of their Region.

Consider data classification, data protection requirements, and access controls carefully. These elements are crucial for maintaining regulatory compliance while operating efficiently across multiple jurisdictions.

Key implementation elements:

- Deploy workloads in AWS Regions aligned with regulatory requirements
- Use AWS Organizations to enforce Region-specific guardrails
- Implement data classification and encryption at rest and in transit

Implementation steps

1. Develop a regulatory assessment framework for each target Region. Document data protection laws, processing requirements. Create a compliance matrix and engage legal and compliance teams for validation.
2. Configure AWS Organizations with Region-specific and compliance-based OUs. Implement service control policies and AWS Control Tower to set up guardrails.
3. Implement a multi-account structure for each Region using AWS Organizations. Create production, development, shared services, and security accounts per Region. Document the account hierarchy and relationships.
4. Define an encryption strategy within a Region. The strategy should include the type of encryption keys, secure key storage, key policies, access controls and key rotation. Use relevant AWS services such as AWS KMS and AWS CloudHSM.
5. Design and configure network architecture for Regional segmentation. Use relevant AWS services such as Amazon VPC, AWS Transit Gateway or AWS PrivateLink to provide network segmentation, security controls and connectivity in each Region.

Resources

Related best practices:

- SEC09-BP02 Enforce encryption in transit
- SEC08-BP02 Enforce encryption at rest
- DRHCSEC03-BP01 Implement controls that enhance your digital sovereignty governance posture

Related documents:

- AWS Organizations User Guide

- Data Residency with Hybrid Cloud Services Lens

Related videos:

- AWS re:Inforce 2025-Navigating sovereignty requirements: Architectures and solutions on AWS (DAP202)

Related services:

- Amazon VPC
- AWS CloudHSM
- AWS Control Tower
- AWS KMS
- AWS Organizations
- AWS PrivateLink
- AWS Resource Groups
- AWS Systems Manager
- AWS Transit Gateway

DSREL01-BP06 Train team members on recovery procedures

In regulated industries, training on recovery procedures is essential for maintaining regulatory adherence, minimizing incident impact, and protecting against financial and reputational damage. Well-trained teams can run recovery plans effectively under time-pressure, maintaining both regulatory adherence and swift system restoration while reducing human error.

Desired outcome: Team members can run recovery procedures efficiently, meeting regulatory requirements and minimizing business disruption.

Common anti-patterns:

- Training only specific individuals creates single points of failure and knowledge bottlenecks.
- Using content that doesn't reflect current infrastructure, procedures, or regulatory requirements.
- Relying on theoretical training without regular drills in realistic environments.
- Conducting training too infrequently and failing to measure effectiveness or competency.
- Not incorporating industry-specific regulatory requirements into training programs and recovery steps.
- Failing to properly document procedures, leading to inconsistency and knowledge gaps.

Benefits of establishing this best practice:

- Well-trained teams run procedures quickly and decisively, minimizing business impact through practiced workflows.
- Structured training programs demonstrate due diligence and maintain regulatory alignment.
- Regular practice builds confidence, reduces stress, and enhances cross-functional collaboration during incidents.
- Systematic training minimizes human error and knowledge loss while lowering incident-related costs.
- Training sessions identify optimization opportunities while building confidence among customers, partners, and regulators.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Establish a task-based training program that combines theoretical knowledge with hands-on practice through simulation exercises and game days.

Key implementation elements:

- Develop role-specific recovery runbooks
- Use AWS services to simulate failures (for example, chaos engineering)
- Schedule quarterly recovery drills
- Automate compliance validation during recovery

Implementation steps

1. Define a training program structure with role and product domain based curricula and learning objectives using AWS Skill Builder. Establish assessment criteria and certification paths to validate team member competency. Create a training schedule and document training requirements.
2. Define what each role needs to do during recovery, including operations procedures and step-by-step actions. Add validation checks so teams can verify that they're on track at each stage. Include compliance controls so that regulatory requirements are met during recovery. Set up version control to track changes and schedule quarterly reviews to keep runbooks current.
3. Set up a simulation environment using AWS Resilience Hub and AWS Fault Injection Service.

Resources

Related best practices:

- OPS07-BP01 Ensure personnel capability
- OPS03-BP07 Resource teams appropriately
- ADVREL05-BP01 Perform routine evaluation of your workload's fault tolerance capabilities
- OPS07-BP03 Use runbooks to perform procedures
- SEC10-BP04 Develop and test security incident response playbooks

Related documents:

- Creating your own runbooks

Related videos:

- AWS GameDay - Learn by doing

Related services:

- Amazon CloudWatch
- Amazon EventBridge
- AWS Audit Manager
- AWS CloudFormation
- AWS CloudTrail
- AWS Config
- AWS Fault Injection Service
- AWS GameDay
- AWS Systems Manager
- AWS Well-Architected

DSREL01-BP07 Create manual backup plans for system outages

In highly regulated industries, manual contingency plans are essential for maintaining business continuity and regulatory compliance when primary systems fail. These plans serve as critical fallback procedures that enable organizations to continue operations and protect customer data while addressing technical failures. This practice assists in avoiding compliance violations, financial penalties, and reputational damage.

Desired outcome: Critical business operations continue with minimal disruption during system outages, maintaining service levels and regulatory compliance.

Common anti-patterns:

- Assuming automated systems and recovery mechanisms work in every scenario without manual backup procedures.
- Maintaining outdated or incomplete procedures that haven't been validated through regular drills.
- Creating single points of failure by not clearly defining roles or training multiple staff on manual processes.
- Failing to align manual procedures with industry-specific compliance requirements and audit obligations.
- Lacking clear escalation paths and failing to prepare necessary tools for manual operations.

Benefits of establishing this best practice:

- Maintains operational continuity and minimizes downtime during system outages.
- Demonstrates proactive risk management and meets regulatory requirements for business continuity.
- Enables immediate action and streamlined decision-making rather than waiting for system restoration.
- Maintains service levels and communication during incidents, protecting brand reputation.
- Creates a trained workforce capable of running critical functions under challenging conditions.

Level of risk exposed if this best practice is not established: Low

Implementation guidance

Begin by identifying mission-critical systems and mapping dependencies. Develop clear, role-based procedures for manual operations, and integrate these plans into broader disaster recovery strategies. Regularly test and refine processes to address gaps.

Implementation steps

1. Identify and document critical systems using AWS Systems Manager and AWS Resource Groups, defining priorities for core services, business applications, data stores, and infrastructure components.
2. Define recovery objectives per system, including RTO, RPO, maximum tolerable downtime, and data loss tolerance, using AWS Resilience Hub and documenting requirements.
3. Map dependencies using Service Catalog and AWS Config, documenting service dependencies, data flows, network connections, and external services.
4. Develop role-based procedures for operations, security, application owners, and support staff, documenting using AWS Systems Manager Documents and including validation steps.
5. Configure monitoring with Amazon CloudWatch for system metrics, service status, recovery progress, and compliance checks, setting up alerts and creating dashboards.
6. Implement a comprehensive documentation management system (DMS), creating operation manuals, quick reference guides, troubleshooting guides, and contact lists, with version control and scheduled reviews. Services that can support DMS include AWS Amazon Kendra, AWS DataSync alongside AWS Cloud Storage and AWS Databases.

Resources

Related best practices:

- OPS10-BP01 Use a process for event, incident, and problem management
- SEC06-BP03 Reduce manual management and interactive access
- REL13-BP01 Define recovery objectives for downtime and data loss

Related documents:

- AWS Resilience Hub User Guide
- AWS Systems Manager Documents
- Business Resilience
- What is DMS

Related videos:

- Backup and Disaster Recovery Strategies for Increased Resilience: Leveraging AWS Services for Cost-Effective Business Continuity

Related services:

- Amazon CloudWatch
- AWS Config
- AWS Resilience Hub
- AWS Resource Groups
- Service Catalog
- AWS Systems Manager

DSREL02-BP01 Implement continuous third-party risk management (TPRM) processes

In highly regulated industries, a robust third-party risk management (TPRM) process is essential to mitigate risks associated with vendor relationships.

A TPRM framework maintains alignment between third-party vendors and an organization's security standards, regulatory requirements, and business continuity objectives. This protects against vulnerabilities, regulatory violations, and operational disruptions that could lead to penalties, data breaches, or reputational damage.

Desired outcome: Third-party vendor risks are identified, assessed, and mitigated throughout the vendor engagement lifecycle, maintaining alignment with security standards and compliance requirements.

Common anti-patterns:

- Conducting one-time or as-needed vendor assessments without standardized criteria or ongoing monitoring.
- Relying on manual tracking processes and lacking clear understanding of data flows and vendor dependencies.
- Using varying security criteria across vendors and failing to implement standardized security requirements in contracts.
- Over-relying on vendor self-attestations.
- Allowing unapproved technology usage and lacking proper incident response coordination with vendors.
- Failing to adapt evaluations to evolving threats and regulations while neglecting continuous compliance monitoring.

Benefits of establishing this best practice:

- Real-time monitoring of vendor risk posture with automated alerting and proactive supply chain risk identification.
- Systematic evidence collection and centralized documentation demonstrating adherence to industry regulations.
- Streamlined vendor onboarding/offboarding processes while maintaining security standards and cost optimization.
- Pre-established communication channels and procedures for coordinating security incidents and protecting operations.
- Enhanced confidence from auditors, regulators, and customers through demonstrated vendor risk management.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Implement a continuous TPRM lifecycle process integrated with existing risk management, procurement, and compliance functions, using AWS services for automation and monitoring.

Key implementation elements:

- Establish centralized TPRM governance with clear roles and responsibilities
- Develop standardized vendor assessment questionnaires aligned with regulatory requirements
- Create vendor risk scoring models based on data sensitivity and service criticality
- Implement automated monitoring and alerting for vendor risk changes
- Maintain contractual safeguards and incident response alignment
- Maintain continuous compliance checks and detailed audit trails

This approach creates a scalable process that maintains consistent security and regulatory standards across third-party relationships.

Implementation steps

1. Establish a governance structure with a TPRM steering committee, risk assessment team, compliance officers, and business stakeholders.
2. Implement an assessment framework using AWS Audit Manager for vendor questionnaires, risk assessment templates, compliance checklists, and evidence collection.

3. Configure a vendor management database to track vendor profiles, risk scores, compliance status, and contract details.
4. Set up automated monitoring using Amazon EventBridge, Amazon CloudWatch, AWS Security Hub, and AWS Config. Configure risk alerts, compliance checks, and performance monitoring.

Resources

Related best practices:

- SEC03-BP09 Share resources securely with a third party
- MASEC 2: What security tools (AWS or third-party) do you use?

Related documents:

- Plan your AWS account governance structure

Related videos:

- AWS re:Inforce 2024 - Automation in action: Strategies for risk mitigation (GRC301)

Related tools:

- Amazon CloudWatch
- Amazon DynamoDB
- Amazon EventBridge
- Quick Suite
- AWS Audit Manager
- AWS CloudTrail
- AWS Config
- AWS Organizations
- AWS Security Hub
- AWS Systems Manager

Workload architecture

DSREL03: How do you design workloads that retain sovereignty while providing for recoverability?

Organizations must maintain full control over their data and workloads for regulatory adherence, security, and strategic autonomy, while also implementing robust recovery mechanisms that may be dependent on external services or resources. Seek a balance between protecting sensitive information and maintaining operational independence, while still using the benefits of distributed computing and cloud technologies for business continuity.

Best practices

- DSREL03-BP01 Design automated jurisdiction-aware, fully-observable recovery procedures
- DSREL03-BP02 Design workloads to be interoperable across primary and disaster recovery (DR) sites
- DSREL03-BP03 Design for code, config, and data portability across primary and disaster recovery (DR) sites

DSREL03-BP01 Design automated jurisdiction-aware, fully-observable recovery procedures

Highly regulated industries need automated recovery procedures that take into account the applicable jurisdiction. These procedures must comply with data sovereignty laws and industry-specific regulations. Compliance-aligned recovery automation assists in reducing inadvertent violations during incidents while maintaining rapid response capabilities. This approach provides complete audit trails and enables organizations to reduce potential legal penalties and business disruption.

Desired outcome: Disaster recovery procedures maintain jurisdictional adherence and data sovereignty requirements while providing rapid recovery capabilities and complete audit trails.

Common anti-patterns:

- Automatically replicating data across Regions without understanding or respecting local data protection laws and regulatory requirements.
- Implementing generic or black-box recovery procedures that lack transparency and jurisdiction-specific considerations.
- Failing to regularly test recovery procedures in compliance-constrained scenarios and lacking automated compliance checks.
- Missing comprehensive audit trails and automated notification workflows for regulators and compliance teams.
- Not monitoring for regulatory changes and failing to maintain dynamic region-specific configurations.

Benefits of establishing this best practice:

- Pre-validated, automated procedures reduce downtime and minimize manual compliance verification steps.
- Automated enforcement of jurisdictional requirements and data residency rules reduce the chances of potential compliance violations during recovery.
- Comprehensive audit trails and transparent recovery procedures satisfy regulatory requirements.
- Jurisdiction-aware automation and predefined procedures assists with reducing human error and inadvertent violations.
- Improved confidence from regulators and auditors while optimizing costs through reduced manual oversight and avoided penalties.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Design a compliance-first recovery architecture that embeds regulatory requirements directly into automation logic using AWS services.

Implementation Steps

1. Map regulatory requirements to technical controls using AWS Control Tower, AWS Security Hub Cloud Security Posture Management (CSPM), AWS Config Rules and AWS Systems Manager runbooks for automated compliance validation and remediation.
2. Implement infrastructure as code templates for jurisdiction-aware recovery workflows with appropriate regional controls using AWS CloudFormation.

3. Use resource tagging and AWS Organizations service control policies (SCPs) to enforce data sovereignty requirements and block unauthorized cross-Region transfers.
4. Deploy comprehensive observability using AWS CloudTrail, Amazon CloudWatch, and centralized logging for immutable audit trails.
5. Establish policy-driven recovery frameworks that evaluate compliance constraints before running recovery actions using AWS Backup, AWS Elastic Disaster Recovery (DRS), AWS Resilience Hub.

This approach facilitates automated, compliance-aligned recovery processes while maintaining complete visibility for regulatory reporting.

Resources

Related best practices:

- OPS01-BP03 Evaluate governance requirements
- OPS01-BP04 Evaluate compliance requirements
- REL13-BP05 Automate recovery

Related videos:

AWS re:Inforce 2023 - Best practices for cloud governance at scale (GRC305)

Related services:

- AWS Audit Manager
- Amazon CloudWatch
- AWS CDK
- AWS CloudFormation
- AWS CloudHSM
- AWS CloudTrail
- AWS Config
- AWS Control Tower
- AWS KMS
- Amazon OpenSearch Service
- AWS Organizations

- AWS Systems Manager

DSREL03-BP02 Design workloads to be interoperable across primary and disaster recovery (DR) sites

In highly regulated industries, workloads must be designed for seamless interoperability across primary and disaster recovery sites to maintain both operational continuity and regulatory requirements. Customers must aim for reduction in disruption during failovers while meeting regulatory demands for data integrity, maintaining audit trails, and meeting recovery objectives.

Desired outcome: Workloads operate consistently across primary and disaster recovery sites, maintaining performance, security controls, and regulatory requirements during normal operations and failover scenarios.

Common anti-patterns:

- Hard-coding Region-specific configurations and creating dependencies that block applications from running identically across sites.
- Using different security policies, Identity and Access Management (IAM) configurations, and compliance controls between primary and DR environments.
- Relying on human intervention for failover and deployments, increasing risk of errors and recovery time.
- Creating tightly coupled architectures and data silos that don't account for cross-region requirements.
- Inadequately testing DR functionality and treating it as a cold standby rather than a fully validated environment.

Benefits of establishing this best practice:

- Reduced Recovery Time Objective (RTO) through consistent configurations and automated failover, minimizing downtime during outages.
- Uniform controls, audit capabilities, and configurations across environments meeting data redundancy requirements.
- Improved confidence through regularly validated DR capabilities, automated processes, and simplified maintenance.
- Efficient resource utilization through automated scaling and ability to use DR sites for testing and development.

- Demonstrated interoperability across regions with validated performance under various failure scenarios.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Design containerized or serverless workloads using environment agnostic architectures and infrastructure as code (IaC) to maintain consistency across primary and DR sites.

Key implementation elements:

- Deploy containerized or serverless architectures that abstract infrastructure dependencies
- Use parameter stores and configuration management for environment-specific, jurisdiction-aware settings
- Implement automated CI/CD pipelines for identical deployments across regions
- Configure automated data replication and synchronization between sites
- Establish comprehensive health checks, monitoring, and logging across regions
- Regularly test DR plans and automated failover mechanisms

This approach improves workload portability while maintaining operational consistency and reliability across environments.

Implementation steps

1. Implement a containerization strategy depending on knowledge using Amazon ECS and if kubernetes knowledge exists using Amazon EKS to configure container registries, orchestration, scaling policies, and health checks.
2. Deploy serverless applications using AWS Lambda, Amazon API Gateway, and AWS Step Functions, configuring regional endpoints.
3. Set up configuration management with AWS Systems Manager Parameter Store for environment variables, application configs, secrets management, and regional settings.
4. Develop infrastructure as code using AWS CloudFormation and AWS CDK, including environment parameters, regional configurations, resource definitions, and dependencies.
5. Configure a CI/CD pipeline with AWS CodePipeline for source control integration, build processes, testing frameworks, and deployment stages, and implement AWS CodeBuild.
6. Implement data replication using Amazon S3 replication, Amazon RDS read replicas, and AWS DMS, configuring synchronization, monitoring, and failover.

Resources

Related best practices:

- DRHCOPS03-BP02 Understand factors that determine your data replication strategy

Related videos:

- Mastering Observability: Building Resilient Systems on AWS with CloudWatch and X-Ray
- AWS re:Invent 2025 - Architecting resilient multicloud operations, feat. Monzo Bank (HMC201)
- AWS re:Invent 2025 - Build and optimize edge architecture for resiliency with AI (HMC403)
- AWS re:Invent 2025 - Digital sovereignty and data residency w/ AWS Hybrid and Edge services (HMC310)

Related services:

- Amazon API Gateway
- Amazon ECS
- Amazon EKS
- Amazon RDS
- Amazon S3
- AWS CDK
- AWS CloudFormation
- AWS CodeBuild
- AWS CodePipeline
- AWS DMS
- AWS Lambda
- AWS Step Functions
- AWS Systems Manager Parameter Store

DSREL03-BP03 Design for code, config, and data portability across primary and disaster recovery (DR) sites

In highly regulated industries, designing for workload portability across primary and disaster recovery sites is both a regulatory mandate and business necessity. This approach provides seamless failover capabilities while maintaining regulatory requirements, minimizing recovery times, and reducing dependencies on specific services or regions, ultimately protecting against both operational disruptions and vendor lock-in risks.

Desired outcome: Code, configurations, and data can be deployed and recovered consistently across primary and disaster recovery sites, maintaining compliance requirements and operational continuity during failover scenarios.

Common anti-patterns:

- Hard-coding region-specific resources (for example, Amazon Machine Image (AMI) IDs, Amazon Resource Names (ARNs), endpoints) in application code or infrastructure templates.
- Relying on manual processes for configuration management and deployments without version control or automated pipelines.
- Creating Region-dependent storage patterns and failing to account for cross-region replication requirements.
- Storing environment-specific secrets and parameters within application code rather than using centralized parameter stores.
- Implementing DR as an afterthought and neglecting regular testing of failover procedures.
- Using single-region Domain Name System (DNS) and networking configurations that create bottlenecks during failover scenarios.
- Ignoring encryption and compliance requirements during cross-region data transfers.

Benefits of establishing this best practice:

- Reduced Recovery Time Objective (RTO) and Recovery Point Objective (RPO) through automated failover, replication, and consistent infrastructure provisioning across Regions.
- Demonstrated robust DR capabilities meeting regulatory requirements while enabling quick response to regional outages.
- Reduced human error and configuration drift through automated processes and infrastructure as code practices.
- Faster development cycles with portable code and simplified DR validation through consistent environments.
- Efficient resource utilization through automated scaling and ability to use multiple regions for active-active scenarios.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Design a portable architecture that separates application logic, infrastructure, and data management using AWS services.

Implementation steps

1. Establish an infrastructure as code foundation using AWS CloudFormation and AWS CDK for consistent, Region-agnostic deployments, creating base templates, Regional variations, parameter files, and resource mappings.
2. Set up configuration management with AWS Systems Manager Parameter Store and AWS Secrets Manager for parameter hierarchies, environment configs, and Regional settings.
3. Deploy containerized applications using Amazon ECS and Amazon EKS for consistent runtime environments, configuring task definitions, service discovery, auto scaling, and load balancing.
4. Configure automated cross-Region data replication with Amazon RDS read replicas, Amazon S3 cross-Region replication, Amazon DynamoDB global tables, and AWS DMS.
5. Establish consistent governance framework using AWS Control Tower, AWS Organizations for service control policies, tag policies, and backup policies, and configure AWS Config for compliance monitoring.
6. Create CI/CD pipelines using AWS CodePipeline, AWS CodeBuild, and AWS CodeDeploy for multi-Region deployment with environment-specific parameters.
7. Implement comprehensive cross-Region health checks and monitoring using Amazon Route 53, AWS Global Accelerator, and AWS Health.

Resources

Related best practices:

- GENREL05-BP02 Replicate embedding data across all regions of availability
- DRHCOPS03-BP02 Understand factors that determine your data replication strategy

Related services:

- AWS CloudFormation
- AWS CDK
- AWS CodeBuild
- AWS CodeDeploy
- AWS CodePipeline

- AWS Config
- AWS DMS
- Amazon DynamoDB
- Amazon ECS
- Amazon EKS
- AWS Organizations
- Amazon RDS
- Amazon S3
- AWS Secrets Manager
- AWS Systems Manager

Change Management

DSREL04: How do you implement both region-agnostic and region-aware change?

Region-agnostic components can be managed uniformly across your onboarded regions, simplifying updates and reducing operational complexity, while region-aware components can be tailored to meet specific local requirements, regulations, and performance needs. This separation allows for more efficient change management processes, better resource utilization, and improved system reliability while also improving adherence to regional data sovereignty and regulatory requirements.

Best practices

- DSREL04-BP01 Design systems with clear boundaries between the core solution and its regional implementations
- DSREL04-BP02 Design adaptive capacity management within sovereign boundaries

DSREL04-BP01 Design systems with clear boundaries between the core solution and its regional implementations

Establish clear architectural boundaries between core solution and regional implementations to effectively manage regulatory requirements across multiple jurisdictions while maintaining operational efficiency. This separation enables organizations to adapt to legal requirements and local regulatory requirements, particularly data sovereignty laws and privacy regulations. This approach reduces the risk of compromising foundational system integrity or risking cross-region data leakage.

This structured approach also enables faster regional expansion and reduced operational complexity. It maintains centralized governance and security standards while supporting compliance with diverse regulatory frameworks.

Desired outcome: Regional data remains within designated geographic boundaries to satisfy data sovereignty requirements, while core services apply consistent security policies across regions. Regional teams independently deploy features and scale workloads without impacting other regions, and regional failures remain isolated. New regions are onboarded using standardized templates, and compliance audits can be scoped to individual region.

Common anti-patterns:

- Deploying monolithic applications across regions without separating core services from region-specific components, leading to unnecessary complexity and compliance risks.
- Embedding region-specific configurations, business rules, and compliance requirements directly into core application code rather than maintaining modular separation.
- Creating direct dependencies between regions through shared databases, overlapping IAM roles, or tight coupling of resources that prevent independent operation.
- Implementing inconsistent API contracts and interfaces across regions, breaking interoperability and complicating maintenance.
- Relying on manual deployment processes without automated pipelines, resulting in configuration drift and inconsistent implementations.
- Failing to properly segment networks and isolate regional workloads using appropriate AWS account structures, VPCs, and subnets.
- Centralizing sensitive data without appropriate filtering or controls, such as aggregating regional logs into a single global bucket.

Benefits of establishing this best practice:

- Supports adherence to region-specific regulations while maintaining consistent security controls across the core solution and regions.
- Reduces time to onboard new regions by reusing core service components, standardizing deployment processes, and allowing region-specific feature deployment without impacting the solution.

- Isolates regional failures to prevent cascading effects on other regions or core services, limiting the blast radius during outages.
- Enables independent solution updates, right-sizing of resources per region based on local demand, and transparent allocation of expenses to regional cost centers.
- Maintains uniform functionality across regions while accommodating local requirements, preferences, and customizations.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Architecting applications for multi-regional deployment in regulated industries requires a thoughtful hub-and-spoke approach that balances global consistency with local compliance requirements. Organizations should establish a central hub for core services (authentication, business logic, and configuration management) while implementing spoke regions that handle local data processing and compliance-specific workflows. This separation is best achieved through AWS organizational units (OUs) through AWS service organizations, dedicated accounts, and proper networking constructs that enforce both logical and physical boundaries. By using well-defined APIs and event-driven patterns, organizations can establish loose coupling between components while maintaining consistent deployment pipelines that adapt core services to regional requirements without modifying the underlying code base.

Key implementation elements:

- Use separate AWS accounts for core and regional workloads.
- Implement infrastructure as code (IaC) with region-specific parameters.
- Enforce network traffic controls (for example, VPC peering or AWS Transit Gateway).
- Establish a central hub region for core services.
- Design standardized APIs for communication between core and regional components.
- Implement region-specific configuration management. Use event-driven architecture to achieve loose coupling.

Implementation steps

1. Establish an AWS Organization Structure with AWS Organizations, creating OUs for core and regional workloads. Implement SCPs for compliance, and configure tag policies for resource management, while also implementing AWS Control Tower for governance.

2. Separate accounts by creating separate AWS accounts for core services, regional workloads, and shared services. Implement AWS IAM Identity Center for centralized access management.
3. Develop infrastructure as code (IaC) using AWS CloudFormation or AWS CDK. Create core and regional infrastructure templates with parameterized configurations for region-specific settings.
4. Design network architecture using Amazon VPC with a hub VPC for core services and spoke VPCs for regional workloads. Use AWS Transit Gateway for traffic control and network segmentation. Alternatively, set up VPC Peering where appropriate.
5. Deploy core services in the central hub region. Use AWS Resource Access Manager (RAM) to share resources within an AWS Region. Consider Transit Gateway inter-Region peering to share access if required.
6. Implement an API layer using Amazon API Gateway, designing standardized APIs for core-regional communication. Implement API versioning, configure regional endpoints, and use AWS PrivateLink to secure API access.

Resources

Related best practices:

- REL08-BP04 Deploy using immutable infrastructure
- DRHCOPS07-BP01 Use AWS services and tools for automation and infrastructure as code (IaC) across hybrid and edge environments

Related documents:

- Organizing Your AWS Environment Using Multiple Accounts
- Reliability Pillar - AWS Well-Architected Framework
-
- AWS Control Tower User Guide

Related videos:

- AWS re:Invent 2024 - Anatomy of an AWS Region (ARC204)
- AWS re:Invent 2024 - Best practices for creating multi-Region architectures on AWS (ARC323)

Related tools:

- Amazon API Gateway
- Amazon Cognito
- Amazon VPC
- VPC Peering
- AWS CloudFormation
- AWS Control Tower
- AWS IAM Identity Center
- AWS Lambda
- AWS Organizations
- AWS PrivateLink
- AWS Systems Manager Parameter Store
- AWS Transit Gateway
- AWS Organization Structure
- AWS CDK

DSREL04-BP02 Design adaptive capacity management within sovereign boundaries

Implement adaptive capacity management strategies that balance operational resilience with data sovereignty requirements in highly regulated environments. This approach enables organizations to automatically scale resources within approved jurisdictional boundaries while maintaining strict compliance with data residency regulations.

Desired outcome: Workloads scale automatically to meet demand while maintaining adherence to data sovereignty requirements. Resources adjust dynamically within approved geographic boundaries without manual intervention, optimizing both performance and cost. Automated policies confine data and compute resources to designated jurisdictions during scaling events. Each capacity change is logged with compliance metadata, providing complete audit trails for regulatory review.

Common anti-patterns:

- Implementing disaster recovery or auto scaling mechanisms that inadvertently move data or compute resources across sovereign boundaries without proper governance.
- Over-provisioning resources to handle peak loads rather than implementing dynamic scaling within approved regions, leading to unnecessary costs.

- Failing to implement comprehensive resource tagging and automated compliance checks for sovereignty tracking and policy enforcement.
- Making scaling decisions without considering data residency requirements or validating AWS service compliance with local regulations.
- Relying on manual capacity management instead of automated, policy-driven scaling processes that respect compliance boundaries.
- Neglecting real-time monitoring and alerting systems for tracking sovereignty compliance during scaling events.

Benefits of establishing this best practice:

- Maintains continuous regulatory adherence through automated enforcement of data residency and sovereignty requirements during scaling operations.
- Optimizes costs by dynamically right-sizing resources based on actual demand while maintaining compliance within sovereign boundaries.
- Enhances operational resilience through automated scaling and distributed capacity across multiple availability zones within approved regions.
- Improves incident response through automated scaling mechanisms that respect compliance boundaries without manual intervention.
- Provides comprehensive audit readiness through detailed logging and monitoring of capacity decisions and their compliance impact.
- Reduces compliance risk by systematically blocking unauthorized data or resource movement outside approved jurisdictions.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Implementing effective capacity management in regulated environments requires a balanced approach that combines automated scaling capabilities with strict data sovereignty controls. Organizations must establish a comprehensive framework that uses AWS services while enforcing policy-driven governance, so that capacity adjustments maintain adherence to regulatory requirements and data residency obligations.

Success depends on creating a robust architecture that integrates resource tagging, service control policies, and automated scaling mechanisms with continuous compliance monitoring, enabling organizations to dynamically adjust capacity while maintaining strict control over data location and resource boundaries.

This approach not only optimizes operational efficiency but also demonstrates due diligence in maintaining regulatory adherence, particularly crucial for organizations operating in highly regulated industries where data sovereignty violations can result in significant penalties and loss of trust.

Key implementation elements:

- Identify geographic restrictions for data.
- Select AWS services and Regions certified for sovereignty.
- Implement auto scaling with guardrails for resource limits.
- Track data location and resource configurations.

Implementation steps

1. Implement a resource tagging strategy to identify data classification and sovereignty requirements. Use AWS Resource Groups to organize resources by compliance scope. Enforce tagging using AWS Organizations Tag Policies and Service Control Policies. Use AWS CloudFormation Guard rules to block incorrectly tagged resources from being deployed.
2. Create capacity management policies, configure AWS Service Quotas for default limits and quota monitoring, and set up resource scheduling for effective resource management.
3. Configure AWS Auto Scaling and Application Auto Scaling with scaling policies, resource limits, and target tracking. Set up scaling notifications to track scale-out and scale-in events.
4. Deploy Amazon CloudWatch for custom metrics, dashboards, and alarms. Implement AWS CloudTrail for logging, and AWS Config rules for continuous compliance assessment.
5. Configure AWS KMS for Region-specific keys, key policies, and key rotation. Set up data movement controls for data sovereignty by constructing data perimeters.

Resources

Related best practices:

- REL07-BP01 Use automation when obtaining or scaling resources
- SUS02-BP01 Scale workload infrastructure dynamically
- PERF02-BP05 Scale your compute resources dynamically

Related documents:

- Data Residency with Hybrid Cloud Services Lens
- Performance and capacity management
- Management and Governance Cloud Environment Guide

Related examples:

- Capacity management
- Implementing and enforcing tagging

Related services:

- Amazon CloudWatch
- AWS Application Auto Scaling
- AWS Auto Scaling
- AWS CloudTrail
- AWS Config
- AWS Control Tower
- AWS KMS
- AWS Organizations
- AWS Resource Groups
- AWS Service Quotas
- AWS Organizations Tag Policies
- Service Control Policies

Failure management

DSREL05: How do you manage failures within sovereign boundaries?

Implementing predictable failure prevention while maintaining sovereign adherence is critical. It improves business continuity without compromising regulatory requirements or data sovereignty obligations. Organizations must balance the need for robust failure prevention with strict compliance requirements and carefully consider where and how data can be stored or processed.

DSREL06: How do you improve the survivability of sovereign workloads during failures?

Improving the survivability of sovereign workloads during failures is crucial. It involves maintaining both operational continuity and regulatory adherence. This can be challenging when compliance restrictions limit your failover options. It requires careful architectural design to overcome unique challenges related to data replication, failover mechanisms, and recovery strategies.

DSREL07: How do you maintain transparency and auditability during failure management?

Clear documentation and traceability of actions taken during a failure event provide essential evidence for regulatory adherence. They support post-incident analysis and enable organizations to identify areas for improvement. This transparency enables organizations to demonstrate their commitment to accountability and proper incident handling.

Best practices

- DSREL05-BP01 Establish sovereignty-aware failure detection mechanisms
- DSREL05-BP02 Design sovereign-compliant failure prevention
- DSREL06-BP01 Design Regional service continuity controls during system degradation
- DSREL06-BP02 Maintain sovereign regulatory adherence during failure states
- DSREL07-BP01 Implement transparent failure management for regulated industries
- DSREL07-BP02 Continually monitor compliance during failures and failovers

DSREL05-BP01 Establish sovereignty-aware failure detection mechanisms

Establishing sovereignty-aware failure detection mechanisms is crucial for maintaining compliance with data residency requirements in highly regulated industries. These mechanisms verify that system monitoring and recovery processes operate strictly within designated jurisdictions. Failure to implement proper sovereignty controls can result in serious compliance violations, operational disruptions, and compromised customer trust.

Desired outcome: Failures are automatically detected and responded to within sovereign boundaries. Monitoring data, alerts, and recovery processes remain within designated jurisdictions. Teams have comprehensive visibility into system health without compliance violations. Recovery procedures run without delays from compliance verification.

Common anti-patterns:

- Consolidating monitoring data from multiple regions into a single global dashboard without considering data residency requirements.
- Using notification services that route alerts through regions outside of regulatory boundaries.
- Aggregating application and system logs in regions that don't align with data sovereignty requirements.
- Implementing external monitoring solutions without verifying their data handling and storage practices.
- Focusing only on technical failures while ignoring compliance and sovereignty-related failure scenarios.
- Having disaster recovery processes that move workloads across jurisdictional boundaries without proper controls.
- Relying on reactive, human-driven checks instead of automated systems.

Benefits of establishing this best practice:

- Maintains adherence to data sovereignty laws during both normal operations and failure scenarios.
- Enables rapid response to failures without concern for inadvertent compliance violations.
- Verifies that monitoring and alerting data remains within appropriate jurisdictional boundaries for regulatory adherence.
- Reduces exposure to regulatory penalties by maintaining sovereignty controls during critical failure scenarios.
- Provides clear visibility into system health while respecting geographical and regulatory constraints.
- Enables immediate response to failures without delays caused by compliance verification processes.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Implement sovereignty-aware failure detection by designing monitoring architectures that respect jurisdictional boundaries while maintaining comprehensive system visibility through Region-specific infrastructure and compliant in-boundary and Region alerting mechanisms.

Use AWS services that provide built-in Regional isolation and sovereignty controls, verifying that monitoring data flows and storage locations align with regulatory requirements.

Establish clear escalation procedures that operate within regulatory constraints and enforce encryption and least privilege access principles.

- Deploy monitoring tools in the same AWS Region as regulated workloads.
- Encrypt logs/metrics at rest and in transit using AWS Key Management Service (KMS).
- Automate alerts and remediation with AWS serverless services like AWS Lambda.

Implementation steps

1. Deploy Regional monitoring infrastructure within the same AWS Regions as regulated workloads. Create Amazon CloudWatch log groups, dashboards, and alarms in each AWS Region where regulated workloads operate to improve data residency adherence.
2. Configure AWS KMS with Region-specific encryption keys for observability data at rest and in transit. Verify that sensitive monitoring data remains within appropriate jurisdictions to maintain data sovereignty.
3. Configure Regional alerting and automation systems within appropriate Regions to enable automated incident response and alert processing within the Region.
4. Establish jurisdiction-specific access controls and compliance monitoring roles with Region-specific permissions. Maintain regulatory adherence across different jurisdictions, and set up continuous compliance validation through AWS IAM and AWS Config.

Resources

Related best practices:

- PERF05-BP02 Use monitoring solutions to understand the areas where performance is most critical
- PERF05-BP05 Use automation to proactively remediate performance-related issues

- OPS04-BP02 Implement application telemetry
- REL13-BP02 Use defined recovery strategies to meet the recovery objectives
- DRHCOPS03-BP04 Implement failover automation, and test your disaster recovery strategies
- OPS10-BP07 Automate responses to events
- SEC04-BP03 Automate response to events
- REL06-BP04 Automate responses (Real-time processing and alarming)

Related documents:

- How AWS is helping customers achieve their digital sovereignty and resilience goals
- How AWS can help you navigate the complexity of digital sovereignty

Related videos:

- AWS re:Invent 2024 - Digital sovereignty: Overcome complexity and enable future-readiness (SEC229)
- AWS re:Invent 2023 - Meet digital sovereignty needs with AWS Dedicated Local Zones (WPS214)

Related services:

- Amazon CloudWatch
- Amazon SNS
- Amazon SQS
- AWS Config
- AWS IAM
- AWS KMS
- AWS Lambda
- AWS Systems Manager

DSREL05-BP02 Design sovereign-compliant failure prevention

Sovereign-compliant failure prevention is essential for highly regulated industries to maintain data residency requirements and regulatory adherence during system failures or disasters. This approach involves designing resilient architectures that keep data and operations within sovereign boundaries. It reduces disruptions, protects against penalties, and supports continuous adherence to local laws and governance requirements.

Desired outcome: Systems recover from failures while maintaining data sovereignty and regulatory adherence. Automated failover and recovery processes keep data and operations within approved geographic boundaries. Business continuity is maintained during incidents without compliance violations. Recovery procedures run with full audit trails, and compliance monitoring remains active throughout failure scenarios.

Common anti-patterns:

- Implementing disaster recovery that moves data or workloads outside sovereign boundaries without proper regulatory approval and sovereignty checks.
- Relying on a single compliance mechanism or control point that could compromise entire regulatory posture if it fails.
- Inadequate encryption controls and centralized key management systems that don't account for sovereign requirements.
- Lacking real-time monitoring, alerting, and automated processes for compliance validation during failures.
- Implementing generic DR approaches without considering jurisdiction-specific requirements, increasing risks through manual processes.
- Inadequate network segmentation and hardcoded non-compliant dependencies that create risks during failures.

Benefits of establishing this best practice:

- Maintains adherence to data sovereignty laws regulating that data is subject to the laws and legal jurisdiction of the nation where it is collected, stored, or processed, reducing exposure to penalties, legal challenges, and reputational damage.
- Supports business continuity within sovereign boundaries while building customer trust through transparent compliance.
- Provides continuous monitoring, automated remediation, and jurisdiction-aware failover to maintain adherence during failures.
- Enables faster recovery times while maintaining compliance posture throughout incidents, with immutable logs for tracking.
- Reduces potential fines, legal costs, and business disruption through proactive compliance-aware failure prevention.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Design a sovereign-compliant failure prevention strategy by implementing defense-in-depth principles across infrastructure, application, data, and compliance layers. Build redundant systems and set up automated compliance monitoring within approved geographical boundaries. Use AWS services that enforce data residency, encryption, and auditable recovery procedures throughout failure scenarios.

- Establish redundant infrastructure across multiple availability zones within geographic-approved Regions, restricting data and backups accordingly
- Implement automated compliance validation with continuous monitoring for both technical and compliance metrics, including drift detection
- Design and automate backup and recovery procedures using infrastructure-as-code (IaC) that respects data residency requirements
- Provide comprehensive data protection through encryption at rest and in transit using customer-managed keys
- Create and validate incident response playbooks through regular audits and testing to maintain regulatory adherence

Implementation steps

1. Conduct initial assessment and planning by documenting regulatory requirements and mapping data flows. Define compliance controls and create an architectural design that meets sovereignty requirements. Verify alignment with regulatory and geographical boundaries.
2. Implement infrastructure layer to create secure and compliant infrastructure within sovereign boundaries. Set up Amazon VPC in approved Regions and configure multi-AZ deployments. Implement network segmentation, deploy AWS Control Tower, and set up AWS Organizations.
3. Implement application layer controls with application-level encryption using AWS Encryption SDK or AWS KMS. Configure service endpoints and set up automated health checks with Amazon Route 53 and AWS Lambda. Deploy AWS WAF (Web Application Firewall) and use AWS App Mesh for compliance-aware routing. This assists with data protection and maintaining application integrity.
4. Implement data layer protection and compliance monitoring by configuring AWS KMS for key management. Implement backup strategies with AWS Backup and enable encryption. Use Amazon Macie for sensitive data monitoring and deploy AWS Config. Set up Amazon

CloudWatch alerts and implement AWS CloudTrail. Create automated compliance dashboards and configure AWS Security Hub for continuous compliance monitoring.

Resources

Related best practices:

- REL11-BP01 Monitor all components of the workload to detect failures
- Plan for Disaster Recovery (DR)
- REL10-BP01 Deploy the workload to multiple locations

Related documents:

- Announcing the Well-Architected Data Residency with Hybrid Cloud Services Lens
- A multi-dimensional approach helps you proactively prepare for failures, Part 2: Infrastructure layer

Related videos:

- AWS re:Invent 2023: Mastering Cloud Governance - Best Practices for Secure and Scalable AWS Environments
- Navigating the Complex World of Data Regulation and Compliance
- AWS re:Invent 2024 - Well-architected for data residency with hybrid cloud services (HYB309)
- AWS re:Invent 2023 - Navigating data residency and protecting sensitive data (HYB309)

Related services:

- Amazon CloudWatch
- Amazon Macie
- Amazon Route 53
- Amazon VPC
- AWS App Mesh
- AWS Backup
- AWS CloudTrail
- AWS Config
- AWS Control Tower

- AWS Encryption SDK
- AWS KMS
- AWS Lambda
- AWS Organizations
- AWS Security Hub
- AWS WAF

DSREL06-BP01 Design Regional service continuity controls during system degradation

Regional service continuity controls are essential for highly regulated industries to maintain uninterrupted operations and regulatory adherence during system degradation or partial failures. These controls verify that workloads remain operational during outages while preserving data integrity and audit trails, enabling organizations to meet compliance mandates and maintain customer trust. Proactive implementation of these controls protects against regulatory violations, reduces risks, and supports business-critical operations to meet service level agreements during Regional disruptions.

Desired outcome: Organizations maintain automated, policy-driven Regional failover and recovery procedures across multiple AWS Regions. Service continuity, regulatory adherence, and data consistency are preserved during system performance degradation or outages. Business operations continue without manual intervention during Regional disruptions.

Common anti-patterns:

- Relying on human intervention for degradation detection and Regional failover, leading to extended downtime and compliance violations.
- Insufficient cross-Region health checks and performance metrics to identify and respond to degradation early.
- Poor data synchronization, conflict resolution, and session management across Regions during failover scenarios.
- Inadequate provisioning in secondary Regions and lack of proper dependency management for third-party services.
- Implementing Regional continuity without verifying that secondary Regions meet compliance requirements and data sovereignty standards.
- Insufficient testing of failover procedures and lack of graceful degradation strategies for maintaining core functionality.

Benefits of establishing this best practice:

- Maintains business operations through automated cross-Region failover capabilities, reducing downtime and data loss during Regional outages.
- Supports ongoing adherence to regulatory requirements with audit trail preservation and automated logging across Regions, meeting uptime and data durability standards.
- Reduces revenue loss and enhances customer confidence through seamless failover and consistent service delivery.
- Provides immediate, policy-driven responses to degradation events, reducing RTO and RPO without manual intervention.
- Optimizes infrastructure costs while maintaining high availability and compliance standards through efficient use of Regional resources.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Design Regional service continuity through multi-Region architectures (active-active or active-passive). Set up automated monitoring and policy-driven failover mechanisms, verifying consistent regulatory posture and maintaining core services during degradation. Focus on automated remediation procedures and regular testing of failure scenarios.

- Deploy across multiple Regions (minimum two when possible) and Availability Zones (minimum three when possible) with consistent security and compliance controls, if allowed within the soverign jurisdictional requirements. When multi-Region deployments are not possible because of data residency constraints, consider other options. For example:
  - Deploy active or passive instances on your own managed infrastructure to support minimum viable service levels. For some types of applications AWS Outposts or AWS Local Zones can also be considered.
  - Deploy active or passive instances with another Cloud Service provider on a temporary basis.
- Implement automated health monitoring, degradation detection, and intelligent DNS routing for seamless failover.
- Design stateless workloads with cross-Region replication capabilities for improved resilience. Configure robust data synchronization and conflict resolution mechanisms where state replication is necessary. For example, to support transactions or maintain event ordering.
- Create automated degradation response playbooks and test regularly using tools like AWS Fault Injection Service.

- Configure robust data synchronization and conflict resolution mechanisms across Regions.

Implementation steps

1. Deploy infrastructure across at least two Regions and three Availability Zones. Configure inter-Region connectivity with AWS Transit Gateway. Use Amazon Route 53 for global DNS management to provide high availability and resilience during system degradation.
2. Configure cross-Region data replication and establish conflict resolution mechanisms to maintain data consistency and availability across Regions using services like Amazon DynamoDB Global Tables and Amazon S3 cross-Region replication.
3. Set up health checks and implement custom metrics with Amazon CloudWatch Synthetics canaries, and use AWS X-Ray for distributed tracing to detect service degradation promptly and trigger appropriate responses.
4. To support seamless service continuity during degradation events, implement DNS health checks, and DNS failovers using Amazon Route 53. Configure Application Load Balancer routing policies, and set up automated failover triggers using AWS Auto Scaling.

Resources

Related best practices:

- REL 10. How do you use fault isolation to protect your workload?
- REL 11. How do you design your workload to withstand component failures?

Related documents:

- Guidance for Cross Region Failover & Graceful Failback on AWS
- Creating a Multi-Region Application with AWS Services series
- Creating an organizational multi-Region failover strategy

Related videos:

- AWS re:Invent 2024 - Best practices for creating multi-Region architectures on AWS (ARC323)
- Back to Basics: Implementing Multi-Region Disaster Recovery with AWS Elastic Disaster Recovery (DRS)
- Managing Cross-region Copies of Backups With AWS Backup

Related services:

- Amazon CloudWatch
- Amazon DynamoDB
- Amazon Route 53
- Amazon S3
- AWS Auto Scaling
- AWS Transit Gateway
- AWS X-Ray

DSREL06-BP02 Maintain sovereign regulatory adherence during failure states

For highly regulated organizations, maintaining data sovereignty and regulatory adherence during system failures is a business-critical requirement. Organizations must verify that their automated failover mechanisms and disaster recovery procedures respect jurisdictional boundaries and compliance mandates. Failures in maintaining sovereign regulatory adherence during outages can result in financial penalties, loss of operating licenses, and damaged customer trust.

Desired outcome: Organizations maintain resilient architectures with automated safeguards and validated recovery processes. Data sovereignty requirements and regulatory adherence are preserved across each failure scenario. Business operations continue from single component failures to regional disasters without compliance violations.

Common anti-patterns:

- Automatically failing over to non-compliant regions and allowing data replication across unauthorized jurisdictions without regulatory validation.
- Missing real-time compliance monitoring during incidents and relying on manual checks instead of automated enforcement.
- Treating data uniformly during recovery without proper classification and storing unencrypted backups in non-compliant regions.
- Insufficient testing of compliance controls during failure scenarios and inadequate documentation of compliance-related decisions.
- Delayed regulatory notifications and incomplete audit trails during emergency situations.

Benefits of establishing this best practice:

- Reduces exposure to violations and associated penalties while maintaining regulatory standing through automated validation.
- Supports rapid business continuity with automated compliance checks while adhering to data residency laws.
- Maintains comprehensive audit trails and documented controls during emergency situations, simplifying audit processes.
- Demonstrates compliance reliability to regulators and customers while avoiding expensive penalties and business disruption costs.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Organizations should implement a defense in depth approach to sovereign regulatory adherence during failure states by designing disaster recovery architectures that respect regulatory boundaries. Use AWS services for compliance monitoring, data residency enforcement, and automated recovery.

Design multi-Region resilient workloads with automated compliance validation while maintaining detailed audit trails throughout each scenario.

- Design multi-layered compliance validation that operates independently across approved AWS Regions
- Implement automated compliance checks and validation within disaster recovery procedures using IaC
- Configure region-specific encryption and key management for data protection at rest and in transit
- Establish Prescriptive escalation procedures and comprehensive testing frameworks that validate both technical and compliance aspects of recovery
- Create detailed audit trails and monitoring systems that maintain compliance visibility during emergencies

Implementation steps

1. Deploy and configure monitoring to support improved governance and compliance across multiple Regions, maintaining sovereign regulatory adherence during failure states. Use AWS Control Tower, AWS Organizations, AWS Config, and Amazon CloudWatch for an AWS setup.

2. Create automation templates with AWS CloudFormation, then implement AWS CDK. Next define AWS Systems Manager automation runbooks. Set up AWS Lambda functions for compliance testing to automate the validation of compliance requirements across regions.
3. Deploy multi-Region keys using AWS Key Management Service (KMS) and configure AWS CloudHSM clusters. Implement TLS using AWS Certificate Manager and set up AWS Secrets Manager for credentials to protect data and maintain adherence during failure states.
4. Create fault injection simulations using AWS Fault Injection Service experiments and implement regular failover testing schedules. Configure automated recovery validation checks and deploy AWS Audit Manager assessments to validate technical and compliance controls, ensuring readiness for failure scenarios.

Resources

Related best practices:

- REL 13. How do you plan for disaster recovery (DR)?
- SEC 9. How do you protect your data in transit?

Related documents:

- Securing and automating compliance in the public sector with AWS
- Automate continuous compliance at scale in AWS
- Unlock the Power of AWS Config: Centralized Compliance and Resource Management

Related videos:

- AWS re:Invent 2024 - How to maintain and automate compliance on AWS (SEC319)
- HDI Group's Innovative Approach: Automated Security and Compliance Remediation Using AWS Cloud Native Services

Related services:

- AWS Audit Manager
- AWS Certificate Manager
- AWS CDK
- AWS CloudFormation

- AWS CloudHSM
- AWS Config
- AWS Control Tower
- AWS Fault Injection Service
- AWS Key Management Service
- AWS Lambda
- AWS Organizations
- AWS Secrets Manager
- AWS Systems Manager
- Amazon CloudWatch

DSREL07-BP01 Implement transparent failure management for regulated industries

Transparent failure management is essential in highly regulated industries, enabling organizations to detect, respond to, and document system failures while maintaining regulatory adherence and trust. The ability to quickly understand failure impacts and maintain detailed audit trails assists with meeting of regulatory requirements. It transforms challenges into opportunities to demonstrate strong operational controls and governance.

Desired outcome: Organizations maintain transparent failure management systems with automated detection, response, and comprehensive audit trails. System failures are quickly identified and resolved while maintaining regulatory adherence without blindspots. Stakeholders have visibility into failure management processes and recovery capabilities.

Common anti-patterns:

- Systems failing without alerts or relying on human monitoring instead of automated detection mechanisms.
- Poor log quality lacking context and centralization, with non-auditable processes.
- Managing failures in silos without proactive monitoring or cross-service correlation.
- Missing escalation procedures, lack of automation, and absence of incident response playbooks.
- Single-AZ deployments, hardcoded dependencies, and insufficient retry mechanisms.
- Inadequate failure simulation and chaos engineering implementation.

Benefits of establishing this best practice:

- Automated audit trails and comprehensive logging support compliance requirements while reducing minor issues from escalating.
- Systematic failure management and structured documentation drive continuous improvement and preserve institutional knowledge.
- Transparent, automated processes reduce Mean Time to Repair (MTTR) and downtime through predefined workflows.
- Improved stakeholder confidence through demonstrable capabilities, cost optimization through early detection, and scalable resilient architectures.

Level of risk exposed if this best practice is not established: High

Implementation guidance

A comprehensive failure management strategy should create a multi-layered system combining automated monitoring, structured logging, and automated response mechanisms. The implementation should focus on:

- Implement self-healing systems using AWS services for monitoring and recovery
- Maintain comprehensive logging and metrics through AWS CloudTrail and Amazon S3 for analysis and audit trails
- Regularly test failure scenarios and recovery processes to improve system resilience

Implementation steps

1. Use AWS Application Load Balancer for multi-AZ architecture and configure Amazon EC2 Auto Scaling groups across Availability Zones. Implement cross-Region failover with Amazon Route 53 health checks. Set up Amazon S3 cross-Region replication and deploy read replicas for databases to support high availability and fault tolerance.
2. Configure Amazon CloudWatch alarms and create AWS X-Ray traces. Set up AWS Lambda functions for remediation and implement AWS Systems Manager automation runbooks. Configure Amazon SNS topics and create Amazon EventBridge rules to automate detection and response to failures.
3. Enable AWS CloudTrail logs and set up a centralized Amazon S3 bucket for log aggregation. Configure Amazon CloudWatch Log groups and implement structured logging with correlation IDs. Create CloudWatch dashboards and set up Amazon Athena queries for log analysis to maintain visibility into system operations.

4. Implement chaos engineering practices using AWS Fault Injection Service and create a regular failover testing schedule. Conduct load testing across Regions and document incident response playbooks. Schedule regular disaster recovery drills and review recovery procedures based on test results to support system resilience.

Resources

Related best practices:

- REL 7. How do you design your workload to adapt to changes in demand?
- REL 9. How do you back up data?

Related documents:

- Automating disaster recovery of Amazon RDS and Amazon EC2 instances
- Automate post-recovery actions using Amazon Elastic Disaster Recovery
- Automate disaster recovery for your self-managed Active Directory on AWS
- Orchestrate disaster recovery automation using Amazon Route 53 ARC and AWS Step Functions

Related videos:

- AWS re:Inforce 2023 - Managing risk in a regulated environment, feat. Japan Digital Agency (GRC302)
- AWS re:Invent 2025 - AI-powered resilience testing and disaster recovery (COP420)
- AWS re:Inforce 2025-Navigating sovereignty requirements: Architectures and solutions on AWS (DAP202)
- AWS re:Invent 2024 - Chaos engineering: A proactive approach to system resilience (ARC326)
- AWS re:Inforce 2023 - Best practices for cloud governance at scale (GRC305)

Related services:

- AWS Application Load Balancer
- AWS CloudTrail
- AWS Fault Injection Service
- AWS Lambda
- AWS Systems Manager

- AWS X-Ray
- Amazon Athena
- Amazon CloudWatch
- Amazon EC2 Auto Scaling
- Amazon EventBridge
- Amazon Route 53
- Amazon S3
- Amazon SNS

DSREL07-BP02 Continually monitor compliance during failures and failovers

In highly regulated industries, maintaining uninterrupted compliance visibility during system failures and failovers is non-negotiable for meeting regulatory requirements and avoiding penalties. Organizations must implement resilient monitoring architectures that reduce blind spots during outages and maintain continuous adherence. Traditional monitoring approaches often fail to maintain adequate oversight during critical events.

Desired outcome: Organizations maintain continuous compliance monitoring and automated remediation capabilities across each failure scenario. Regulatory visibility and audit trails remain uninterrupted during system outages and Regional failovers. Compliance validation continues without manual intervention.

Common anti-patterns:

- Deploying compliance monitoring tools in a single Availability Zone or Region and relying on manual compliance checks without automation.
- Insufficient cross-Region control of compliance data and inadequate log retention and centralization practices.
- Missing failover testing for compliance systems and lack of validation during disaster recovery scenarios.
- Missing meta-data monitoring capabilities, hardcoded compliance rules, and insufficient automation for compliance checks.
- Manual failover processes and unvalidated compliance tools in backup regions.

Benefits of establishing this best practice:

- Maintains continuous audit trails and compliance evidence across each system state, supporting immediate detection of non-compliant resources.
- Provides automated remediation and consistent compliance controls during Regional failovers without manual intervention.
- Reduces risk, mitigates regulatory penalties, and reduces expensive investigations through continuous monitoring.
- Demonstrates robust governance and audit readiness to regulators through comprehensive compliance documentation.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

A resilient compliance monitoring architecture should use AWS services (like AWS CloudTrail, AWS Config, and AWS Security Hub) across multiple regions using active-passive or active-active deployment patterns. Key implementation points include:

- Deploy compliance monitoring tools with built-in redundancy and automated failover mechanisms across Regions
- Implement cross-Region replication for compliance-critical data, configurations, and audit trails
- Establish automated compliance checks, remediation processes, and regular testing during disaster recovery exercises
- Embed compliance monitoring into architectural resilience mechanisms using AWS tools

Implementation steps

1. Enable AWS Security Hub and configure AWS CloudTrail with multi-Region logging. Set up AWS Config with multi-Region aggregation. Deploy Amazon EventBridge rules for cross-Region event routing to support continuous compliance monitoring during failures and failovers.
2. Configure Amazon S3 bucket replication and set up Amazon DynamoDB global tables. Establish Amazon CloudWatch Logs cross-Region subscriptions and enable automated backups. Configure AWS Backup with cross-Region copy and implement AWS Key Management Service (KMS) multi-Region keys to maintain regulatory adherence, data integrity, and availability.
3. Create AWS Config rules for compliance checks and deploy AWS Lambda functions for automated remediation. Set up AWS Systems Manager automation runbooks and configure Amazon CloudWatch alarms for compliance metrics. Implement AWS Audit Manager

assessments and establish regular compliance testing schedules to support ongoing compliance during failures.

4. Set up centralized logging with Amazon OpenSearch Service and create compliance dashboards in Amazon CloudWatch. Configure Amazon Simple Notification Service (SNS) topics for compliance alerts and implement AWS Control Tower for preventive controls. Set up AWS Health API monitoring and enable AWS Trusted Advisor checks to monitor and maintain regulatory adherence during failures and failovers.

Resources

Related best practices:

- REL 12. How do you test reliability?
- OPS 8. How do you utilize workload observability in your organization?

Related documents:

- Continuous compliance monitoring using custom audit controls and frameworks with AWS Audit Manager
- A multi-dimensional approach helps you proactively prepare for failures, Part 3: Operations and process resiliency
- Verify the resilience of your workloads using Chaos Engineering

Related videos:

- AWS re:Invent 2024 - Intelligent continuous compliance: Redefining cloud security for the modern era
- AWS re:Invent 2019: Designing for failure: Architecting resilient systems on AWS (ARC335-R1)
- AWS re:Inforce 2024 - Cloud compliance journey: Compliance and audits (GRC201)

Related services:

- AWS Audit Manager
- AWS Backup
- AWS CloudTrail
- AWS Config
- AWS Control Tower

- AWS Health
- AWS Key Management Service
- AWS Lambda
- AWS Security Hub
- AWS Systems Manager
- AWS Trusted Advisor
- Amazon CloudWatch
- Amazon DynamoDB
- Amazon EventBridge
- Amazon OpenSearch Service
- Amazon S3
- Amazon SNS

Business continuity planning

DSREL08: How do you maintain continued access to key infrastructure, services, and skills needed to support your digital footprint?

Maintaining continued access to key infrastructure, services, and skills needed to support your digital footprint is crucial for improving business continuity, maintaining competitive advantage, and effectively managing risks.

Best practices

- DSREL08-BP01 Maintain a map of key dependencies
- DSREL08-BP02 Analyze potential causes and impact of disruptions
- DSREL08-BP03 Develop mitigation strategies for geopolitical and regulatory risks

DSREL08-BP01 Maintain a map of key dependencies

Without clear dependency mapping, organizations cannot effectively assess incident impact or meet audit requirements. They also cannot make informed architectural decisions that protect critical business functions and improve regulatory adherence.

Desired outcome: Organizations maintain current, comprehensive dependency maps providing complete visibility into service relationships, data flows, and critical path dependencies. Infrastructure, applications, and third-party services are mapped to proactively manage risks and compliance.

Common anti-patterns:

- Relying on spreadsheets or static documents that quickly become outdated, without automated discovery and updates using AWS tools.
- Different teams maintaining separate, incompatible dependency maps without centralized coordination or cross-functional visibility.
- Focusing only on application dependencies while ignoring infrastructure, network, data, and third-party services and APIs.
- Failing to identify and classify mission-critical paths, single points of failure, pre and post-recovery validation processes.

Benefits of establishing this best practice:

- Rapidly identify downstream impacts, accelerate root cause analysis, and prioritize remediation efforts during outages through clear understanding of service interdependencies.
- Meet regulatory requirements while proactively identifying and reducing single points of failure through comprehensive system documentation.
- Make informed technology decisions and optimize disaster recovery strategies based on complete understanding of critical dependency chains.
- Identify cost optimization opportunities through unused resources while enhancing cross-team collaboration with shared architectural understanding.

Level of risk exposed if this best practice is not established: High

Implementation guidance

To maintain a dependency map, start by inventorying AWS resources, third-party services, and data flows. Use automation to track relationships and changes. Prioritize critical workloads and validate dependencies against compliance frameworks.

- Deploy automated discovery and mapping tools across AWS accounts and Regions
- Establish standardized tagging strategies to enable dependency correlation
- Create regular validation and update processes for dependency accuracy

- Integrate dependency maps into operational dashboards and incident response procedures

Implementation steps

1. Inventory AWS resources, third-party services, and data flows. Gain a comprehensive understanding of your current infrastructure and dependencies. Confirm that you have documented your current infrastructure topology and dependencies before proceeding with changes or optimizations.
2. Deploy automated discovery and mapping tools across AWS accounts and Regions. Consider services like:
   - AWS X-Ray or an alternative application tracing service to discover runtime dependencies.
   - AWS Systems Manager Inventory to discover metadata about your managed nodes. Example metadata include EC2 driver, agents, version, tags assigned to nodes, CPUModel, CPUCores, CPUs, and CPUSpeedMHz.
   - AWS Application Discovery Service to discover on-premises servers and databases.
   - And open-source tools like Steampipe to search and build resource inventories.
3. Establish standardized tagging strategies to enable dependency correlation. Define and apply a consistent AWS tagging schema to each resource, allowing you to categorize, track, and manage relationships between applications and resources.
4. Prioritize critical workloads and validate dependencies against compliance frameworks. Regularly review and update your dependency maps to maintain adherence to regulatory requirements and meet your operational needs.
5. Integrate dependency maps into operational dashboards and incident response procedures. This allows you to quickly identify and mitigate risks and maintain adherence during incidents.

Resources

Related best practices:

- Design interactions in a distributed system to prevent failures
- Operational readiness and change management

Related documents:

- Disaster Recovery of Workloads on AWS: Recovery in the Cloud

Related videos:

- How AWS Application Discovery Service Maps Your IT Environment
- Establishing a Data Perimeter on AWS: USAA's Journey to Automated Security Controls
- AWS re:Invent 2024 - How to maintain and automate compliance on AWS (SEC319)

DSREL08-BP02 Analyze potential causes and impact of disruptions

In highly regulated industries, systematic disruption analysis and proactive risk assessment capabilities are essential. Without these practices, organizations face regulatory penalties, extended recovery times, and potential compliance violations.

Desired outcome: Automated, systematic processes identify root causes, assess blast radius, and quantify business impacts of disruptions. Comprehensive audit trails enable risk mitigation and assist in maintaining continuous regulatory adherence.

Common anti-patterns:

- Waiting for incidents rather than conducting proactive threat modeling, chaos engineering, and failure simulation across interconnected systems.
- Teams conducting isolated analysis without automated tooling, standardized methodology, or cross-functional collaboration.
- Failing to evaluate cascading effects across dependencies, third-party services, and business processes while missing regulatory implications.
- No standardized severity criteria or detailed analysis records to support regulatory adherence, resource allocation, and future deterrence.

Benefits of establishing this best practice:

- Reduce mean time to resolution through systematic root cause analysis and predefined playbooks, while creating organizational learning to reduce recurring issues.
- Demonstrate regulatory due diligence through comprehensive analysis documentation while proactively identifying and addressing vulnerabilities before customer impact.
- Strengthen system reliability through better understanding of interdependencies and failure patterns, improving overall business continuity.
- Build confidence with regulators, customers, and leadership through transparent processes while optimizing costs by avoiding fines and reducing downtime.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Establish a multi-layered approach combining automated monitoring, machine learning–based anomaly detection, and structured analysis frameworks that align with regulatory requirements while maintaining detailed audit trails.

- Deploy centralized logging and monitoring with automated correlation capabilities across services and systems
- Establish standardized severity classification and impact assessment frameworks that integrate with compliance and sovereignty requirements
- Implement automated workflows for notification, escalation, and correlation of technical metrics with business KPIs
- Conduct regular chaos engineering exercises and failure simulations to validate analysis processes and critical workload dependencies

Implementation steps

1. Deploy comprehensive observability tools to gain deep insights into system performance, identify bottlenecks, and visualize metrics. This enables proactive identification and resolution of issues before they impact operations. Use AWS Services such as Amazon CloudWatch to detect performance degradation and resource utilization anomalies, AWS X-Ray a distributed tracing service to map request flows across microservices, identify bottlenecks and failure points during disruptions, Amazon Managed Service for Prometheus for high-performance metrics collection across containerized and microservices environments, and Amazon Managed Grafana for rich data visualization and alerting across multiple data sources with unified dashboards that combine infrastructure, application, and business metrics.
2. Implement automated monitoring to continuously detect threats, manage security posture, log API activities, and track resource configurations. Consider using AWS Services such as Amazon GuardDuty which provides intelligent threat detection using machine learning to identify malicious activities that could cause service disruptions, AWS Security Hub which centralizes security findings from multiple AWS security services and third-party tools, providing a unified view of security posture during disruptions, AWS CloudTrail which creates a comprehensive audit trail of API calls and user activities, essential for root cause analysis during disruptions, and AWS Config which continuously monitors and records AWS resource configurations, enabling rapid identification of configuration drift that may cause disruptions.

3. Integrate machine learning–based anomaly detection with services like Amazon DevOps Guru which provides proactive operational insights by analyzing application metrics, logs, and events to identify anomalies before they cause outages, Amazon Detective which accelerates security incident investigation by automatically collecting and correlating log data from multiple AWS sources, and Amazon SageMaker AI which enables custom machine learning models tailored to your specific business context and operational patterns. Automatically identify unusual patterns and behaviors in both operational and business metrics. This allows for early detection of potential issues and informed decision-making.
4. Establish structured analysis frameworks to create a systematic approach to analyzing disruptions, assessing their impact, and maintaining regulatory adherence. Implement incident classification with AWS Systems Manager Incident Manager and create impact assessment methodologies. Consider building and maintaining Correction of Error documents using Incident Manager to improve operational awareness.

Resources

Related best practices:

- REL11-BP01 Monitor all components of the workload to detect failures
- REL12-BP01 Use playbooks to investigate failures
- OPS08-BP01 Analyze workload metrics
- OPS08-BP02 Analyze workload logs
- SEC04-BP01 Configure service and application logging

Related documents:

- AWS Well-Architected Tool
- AWS Resilience Hub
- AWS Fault Injection Service
- AWS Systems Manager Incident Manager

Related videos:

- Supports You | Introducing AWS Resilience Hub
- Prepare & Protect Your Applications From Disruption With AWS Resilience Hub
- AWS re:Inforce 2023 - Engineer application resilience with compliance in mind (GRC304)

DSREL08-BP03 Develop mitigation strategies for geopolitical and regulatory risks

In today's interconnected global economy, highly regulated industries face critical risks from geopolitical tensions, trade restrictions, regulatory changes, and environmental disasters. These can disrupt operations and compromise compliance overnight. Organizations must implement proactive mitigation strategies to address these multifaceted risks. Failure to do so may result in operational shutdowns, regulatory penalties, supply chain disruptions, and significant reputational damage.

Desired outcome: Dynamic, multi-layered risk mitigation strategies with predefined response plans maintain business continuity, regulatory adherence, and operational resilience. Potential geopolitical, regulatory, and environmental disruption scenarios are addressed.

Common anti-patterns:

- Hosting critical workloads and operations in single regions without considering geopolitical risks, regulatory isolation, or localized disruptions.
- Failing to proactively monitor and update strategies for political, trade, and regulatory changes across international frameworks.
- Over-reliance on single-source vendors or technologies vulnerable to export controls, sanctions, or licensing disputes without alternative migration paths.
- Insufficient data sovereignty planning, residency controls, and encryption mechanisms that adapt to changing regulatory requirements across jurisdictions.
- Relying on manual processes for compliance monitoring, disaster recovery, and mitigation testing without automated controls for rapid adaptation.

Benefits of establishing this best practice:

- Maintain business continuity during geopolitical and climate-related disruptions through geographic distribution and pre-planned alternative operational models.
- Support continuous adherence to evolving international regulations, sanctions, and export controls while protecting data sovereignty across jurisdictions.
- Optimize costs while protecting intellectual property, reducing supply chain dependencies, and maintaining technology independence from licensing changes or embargoes.
- Demonstrate robust risk management to regulators, investors, and customers while preserving competitive advantage and strategic agility during market disruptions.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Establish a multi-layered framework combining automated monitoring, scenario planning, and adaptive infrastructure deployment strategies that enable rapid operational pivoting while maintaining continuous regulatory adherence across multiple jurisdictions.

- Deploy geographically distributed infrastructure using AWS services with automated failover capabilities and data sovereignty controls
- Implement continuous monitoring systems for regulatory changes, sanctions lists, and geopolitical developments using editable compliance controls
- Establish vendor diversification strategies and disaster recovery planning with alternatives and validated mitigation procedures
- Design resilient architectures that can withstand regional disruptions while enforcing data residency and export controls across jurisdictions

Implementation steps

1. Deploy infrastructure across multiple Regions, enhancing resilience against geopolitical risks and verifying data sovereignty through services like AWS Control Tower, Amazon Route 53 for global DNS management, AWS Transit Gateway for cross-Region connectivity, and AWS Global Accelerator for traffic distribution.
2. Enable real-time detection and response to meet regulatory and operational changes. Consider AWS Config, AWS Audit Manager, Amazon EventBridge, and AWS Security Hub for regulatory adherence. Amazon CloudWatch, AWS CloudTrail, Amazon DevOps Guru, and AWS Systems Manager OpsCenter for operational monitoring.
3. Configure a robust backup strategy, automated recovery, failover routing and rapid recover using AWS Backup, Amazon S3 Cross-Region Replication, AWS DMS, and AWS Storage Gateway. Create AWS Systems Manager runbooks, configuring AWS Lambda for automated recovery. Set up Amazon Route 53 failover routing, and implement AWS Elastic Disaster Recovery to support rapid recovery from disruptions.
4. Build a secure and compliant environment that can withstand regulatory and geopolitical challenges. Set up AWS Shield for DDoS protection. Implement AWS WAF (Web Application Firewall) for web application security. Enable AWS Macie for data discovery. Configure Amazon Inspector for vulnerability assessments. Set up AWS Artifact for compliance reports, and implement AWS License Manager.

Resources

Related best practices:

1. MGMT02 - How do you manage compliance?

Related documents:

- Controls that enhance data residency protection

Related videos:

- How AWS helps Customers Meet their Security, Risk, and Compliance Objectives
- AWS re:Inforce 2025 - Best practices for managing governance, risk, and compliance globally (GRC301)
- AWS re:Inforce 2024 - Automation in action: Strategies for risk mitigation (GRC301)

Performance efficiency

The performance efficiency pillar for digital sovereignty focuses on achieving optimal system performance while maintaining compliance with jurisdictional requirements for data residency, local control, and regulatory standards. Organizations must balance traditional performance optimization techniques with sovereignty constraints that limit global distribution, require local processing, and mandate specific security controls.

Topics

Definitions

The following are performance efficiency-specific definitions:

- **Traffic routing:** The process of directing network requests and data flows from source to destination based on predefined rules, policies, or algorithms, implemented at multiple layers including DNS, load balancers, and content delivery networks.
- **Geographic routing:** Also called geolocation routing, this is a traffic management strategy that directs user requests to specific endpoints based on the physical or network location from which the request originates, using the source IP address to determine geographic location and apply predefined routing rules. For sovereign workloads, geographic routing is a mandatory compliance control.
- **Geo-fencing:** A security and compliance technology that creates virtual geographic boundaries and enforces restrictions blocking data, requests, or resources from moving beyond defined geographic limits through network restrictions, service configurations, IAM policies, and monitoring systems. Geo-fencing serves as both a preventative and detective control that verifies that sensitive data never leaves approved jurisdictions while providing visibility into attempts to violate geographic constraints.
- **Edge location:** are geographically distributed points of presence containing caching servers and compute resources positioned close to end users to reduce latency and improve application

performance, numbering in the hundreds of locations worldwide as part of Amazon CloudFront CDN infrastructure. For sovereign workloads, edge locations present compliance considerations because cached data temporarily resides outside primary AWS Regions potentially across jurisdictional boundaries.

- **Cloud-based:** Refers to applications and architectures designed specifically to use cloud computing capabilities through microservices architecture, containerization, dynamic orchestration, and managed cloud services rather than simply migrating traditional applications to cloud infrastructure. For sovereign workloads, cloud-based patterns offer significant advantages including built-in compliance features in managed services, regional isolation with independently deployed microservices, and containerization that facilitates portability across cloud providers to avoid vendor lock-in.
- **Multi-layered approach:** Also called defense in depth, this is a security and reliability strategy that implements multiple independent, complementary controls at different architectural layers —including network boundaries, identity and access, data protection, application security, monitoring, and processes—so that if one control fails, others continue to provide protection. For sovereign architectures, multi-layered approaches are essential because compliance isn't guaranteed by a single technology. Organizations must combine geographic controls (Region selection, geo-fencing), cryptographic controls (encryption, key management), access controls (IAM policies, network segmentation), operational controls (monitoring, audit logging), and contractual controls.
- **Multi-Region:** Architectural pattern where application components, data, and infrastructure are distributed across multiple geographically separated AWS Regions to achieve high availability, disaster recovery, performance optimization, or compliance with data residency requirements, with each Region being completely independent with separate control planes and infrastructure.
- **High-availability (HA) architecture:** System design approach that provides continuous operation and minimal downtime even during component failures through redundancy at every layer, removing single points of failure with multiple application servers across availability Zones, load balancers, database replication, and automated failover mechanisms targeting availability of 99.9% or higher. For sovereign workloads, achieving high availability within geographic constraints is challenging because redundant infrastructure must reside within approved regions, requiring organizations to balance availability requirements against sovereignty constraints.

Design principles

- **Optimize within jurisdictional boundaries:** Design architectures that achieve performance goals using resources and services within approved regions. Use local caching, Regional content delivery, and proximity-based routing while respecting data residency requirements.
- **Validate components for sovereignty:** Establish automated processes to verify that compute resources, open source components, and third-party services meet regulatory requirements and security standards before deployment. Maintain comprehensive audit trails for compliance verification.
- **Build for operational independence:** Design networks and systems using open protocols, open data formats, widely available encryption schemes, and standardized interfaces that promote independence and choice.
- **Empower regional teams with local expertise:** Establish operational teams within approved jurisdictions who understand local regulations and can respond quickly to regional requirements, supported by service providers with strong local presence and compliance capabilities.
- **Implement continuous verification and monitoring:** Deploy automated tools to continuously validate network configurations, access patterns, and data flows against sovereignty requirements. Detect and prevent compliance violations in real time while maintaining performance visibility.

Architecture selection

DSPERF01: How do you optimize for performance without compromising on sovereignty?

Optimizing performance without compromising sovereignty is essential for organizations. This practice maintains legal adherence to data protection regulations and provides national security and strategic independence. It preserves business continuity in an evolving geopolitical landscape. Organizations must balance traditional performance optimization approaches with sovereignty requirements. This balance enables organizations to serve regulated markets effectively and maintain customer trust.

Best practices

- DSPERF01-BP01 Implement optimal traffic routing aligning with regulatory needs

- DSPERF01-BP02 Optimize applications while maintaining data sovereignty

DSPERF01-BP01 Implement optimal traffic routing aligning with regulatory needs

Maintaining data sovereignty and jurisdictional adherence is crucial in highly regulated industries. Improper traffic routing can result in legal penalties and security risks. Organizations must implement strategic routing mechanisms that respect geographic boundaries while minimizing latency. This balance between compliance and performance is essential for avoiding regulatory violations.

Desired outcome: Organizations should implement automated traffic routing that optimizes performance and availability while verifying that data flows exclusively through authorized AWS Regions to maintain regulatory adherence.

Common anti-patterns:

- Relying solely on DNS-based geographic routing without considering regulatory boundaries and data residency requirements.
- Implementing static routing configurations that don't adapt to changing network conditions or regulatory updates.
- Failing to implement proper traffic monitoring and auditing mechanisms to improve compliance.
- Neglecting to establish clear data classification and routing policies based on sensitivity levels.
- Overlooking edge location compliance when utilizing content delivery networks and edge computing services.

Benefits of establishing this best practice:

- Improved regulatory adherence through automated enforcement of jurisdictional boundaries and data residency requirements.
- Improved user experience with optimized latency while maintaining compliance constraints.
- Increased system resilience through intelligent multi-Region failover mechanisms that respect regulatory boundaries.
- Reduced operational overhead through automated routing decisions based on predefined compliance policies.
- Enhanced security posture through controlled data flow and reduced exposure to unauthorized jurisdictions.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Implement a multi-layered traffic routing approach using AWS Global Infrastructure services. This approach combines intelligent routing policies with strict compliance controls.

Start by establishing data classification and regional mapping based on regulatory requirements. Configure DNS-based routing with health checks and latency optimization while keeping traffic within approved jurisdictions. Deploy comprehensive monitoring and automated policy enforcement to maintain continuous adherence.

Implementation steps

1. Document data classification requirements and map compliance needs. Verify that traffic routing meets regulatory requirements while maintaining optimal performance within approved AWS Regions. Configure Amazon Route 53 with health checks, latency-based routing, and geolocation rules.
2. Block unauthorized traffic from restricted regions and protect applications at the edge. Implement origin failover configurations to maintain availability during outages with AWS WAF rules and Amazon CloudFront distributions using geo-fencing restrictions.
3. Implement AWS Global Accelerator with regional endpoint groups and cross-region health checks to route traffic over AWS's private backbone for improved performance. Deploy resources across approved regions with Amazon Route 53 DNS failover to provide high availability and disaster recovery capabilities.
4. Continuously monitor traffic flows, detect policy violations in real-time, and maintain comprehensive audit trails for regulatory adherence verification. Enable AWS CloudTrail, VPC Flow Logs, and Amazon CloudWatch metrics combined with AWS Config rules and Amazon EventBridge alerts.

Resources

Related best practices:

- Network protection

Related documents:

- What is AWS Network Firewall?

- Architecture Best Practices for Networking & Content Delivery
- AWS Networking and Content Delivery Blog

Related videos:

- AWS re:Invent 2023 - Enhancing Web Application Performance and Reliability with AWS Global Accelerator
- AWS re:Inforce 2023 - Advanced approaches to traffic inspection & network diagnosis w/ AWS (NIS304)
- Implementing AWS Well-Architected Network Security at Scale: Mercado Libre Case Study
- The Routing Loop - Centralized network traffic inspection: Key insights and lessons learned

Related services:

- Amazon Route 53
- AWS Global Accelerator
- Amazon CloudFront
- AWS WAF

DSPERF01-BP02 Optimize applications while maintaining data sovereignty

Maintaining data sovereignty while optimizing application performance is crucial for compliance and operational efficiency. By minimizing external data transfers and processing data within the cloud, organizations can reduce their compliance footprint and third-party dependencies. This approach improves adherence to strict data governance requirements while maintaining optimal performance and simplified auditability.

Desired outcome: Deliver high-performance applications using cloud-based optimization services. Maintain data sovereignty by removing external service dependencies.

Common anti-patterns:

- Over-relying on external CDNs, third-party optimization services, or APIs that require data to leave the controlled AWS environment.
- Implementing synchronous processing patterns instead of optimizing through asynchronous workflows and queuing mechanisms.

- Neglecting application-level caching strategies and efficient database query optimization.
- Using inefficient serialization formats and failing to implement compression for internal data transfers.
- Deploying monolithic architectures that block granular scaling and optimization of individual components.
- Failing to implement connection pooling, resource reuse patterns, and parallel processing opportunities.
- Storing sensitive data in unencrypted logs.
- Over-provisioning resources instead of optimizing code efficiency.

Benefits of establishing this best practice:

- Improved regulatory adherence and data privacy by keeping the optimization processes and data within the controlled AWS environments.
- Improved application performance and reduced latency through strategic caching, compression, and resource optimization techniques.
- Reduced operational costs and risks by lowering dependencies on external services, minimizing data egress fees, and optimizing resource utilization.
- Increased system reliability and security posture by maintaining complete control over data flow and processing locations.
- Simplified audit trails and compliance verification with AWS tools.
- Greater scalability and flexibility through AWS service integration and auto scaling capabilities.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Optimize applications through internal caching, compression, asynchronous processing, and intelligent resource management, while maintaining data within controlled boundaries. Implement multi-layered optimization strategies using AWS services for compute, storage, and data processing, with data isolation implemented through encryption and VPC controls.

Key implementation components:

- Deploy comprehensive caching strategies using Amazon ElastiCache and application-level caches
- Optimize database performance through connection pooling and query optimization with Amazon DynamoDB
- Implement asynchronous processing using AWS Lambda and messaging services

- Configure auto scaling and container orchestration for dynamic resource optimization
- Use Amazon S3 for internal data storage with AWS KMS encryption
- Use VPC controls and network isolation for secure data processing
- Refactor applications to remove external service dependencies and APIs
- Implement serverless architectures for event-driven optimizations within VPC boundaries

Implementation steps

1. Deploy comprehensive caching using Amazon ElastiCache and DynamoDB Accelerator (DAX) within Amazon VPC private subnets to reduce database load. This improves application response times while maintaining data isolation and remove dependencies on external caching services.
2. Implement asynchronous processing with AWS Lambda, Amazon SQS, and Amazon SNS configured within VPC boundaries to decouple application components, improve scalability, and handle workload spikes without relying on external messaging services or APIs.
3. Configure auto scaling using Amazon ECS with AWS Auto Scaling policies, and optimize database performance through Amazon DynamoDB partition design to dynamically adjust resources based on demand while maintaining cost efficiency and performance within controlled infrastructure.
4. Establish secure storage using Amazon S3 with AWS KMS encryption. Implement AWS PrivateLink with VPC endpoints to keep traffic within AWS boundaries. Consider reducing external service dependencies and maintain regulatory adherence through network isolation. Always apply encryption at rest and in transit.

Resources

Related best practices:

- Performance Efficiency Pillar - AWS Well-Architected Framework

Related documents:

- Building event-driven architectures with Amazon SNS FIFO

Related videos:

- AWS re:Invent 2023 - Simplifying modern data pipelines with zero-ETL architectures on AWS (PEX203)

- AWS re:Invent 2023: Embracing Change and Innovation with AWS Databases for Future-Proof Applications
- Mastering Serverless: Advanced Best Practices for Building Scalable and Efficient Applications on AWS

Related services:

- Amazon ElastiCache
- AWS Lambda
- Amazon DynamoDB
- AWS Auto Scaling

Compute and hardware

DSPERF02: How do you choose optimal compute solutions without compromising on sovereignty?

Customers must balance between using modern computing capabilities and adhering to regulatory requirements. This includes data protection laws and national security policies. Choosing optimal compute solutions improves adherence while maintaining performance and operational efficiency.

Best practices

- DSPERF02-BP01 Implement confidential computing technologies for data protection during processing
- DSPERF02-BP02 Maintain sovereign control over encryption keys
- DSPERF02-BP03 Validate open-source components for sovereignty compliance

DSPERF02-BP01 Implement confidential computing technologies for data protection during processing

Confidential computing technology extends traditional security models for highly regulated industries. It protects sensitive data not just at rest and in transit, but also during processing by using hardware-enforced isolated runtime environments. This enhanced protection addresses critical security gaps by blocking unauthorized access, including from privileged users and cloud providers.

Desired outcome: Protect sensitive data during processing through hardware-enforced isolation. Maintain cryptographic isolation of sensitive data while enabling secure computation.

Common anti-patterns:

- Running highly-confidential workloads on regular EC2 instances without considering hardware-enforced isolation and encryption, and secure enclaves.
- Underestimating CPU and memory requirements and granting excessive IAM permissions to applications handling sensitive data.
- Failing to properly validate enclave attestation documents and overlooking proper key management practices.
- Not properly segregating confidential data flows and allowing unrestricted network access without proper controls.

Benefits of establishing this best practice:

- Block unauthorized access to sensitive data, including from privileged users and cloud operators, through cryptographic isolation.
- Meet stringent data protection requirements with verifiable evidence of data isolation and protection.
- Process data collaboratively across organizational boundaries while maintaining complete confidentiality.
- Minimize potential breach points through hardware-enforced isolation and limited system access.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Identify sensitive workloads requiring confidential computing protection. Implement AWS Nitro Enclaves with proper isolation, and design architectures that separate confidential processing from general application logic and maintain comprehensive security controls.

Key steps include:

- Assess workloads for confidential computing suitability and regulatory compliance requirements
- Design and implement enclave architecture with proper resource allocation, network isolation, and attestation verification
- Configure end-to-end encryption using AWS KMS or CloudHSM with secure communication protocols
- Establish comprehensive monitoring, logging, and incident response procedures while maintaining confidentiality
- Implement strict IAM policies and access controls for enclave interactions using least privilege principles

Implementation steps

1. Catalog sensitive data workflows and identify workloads requiring data-in-use protection, then select appropriate Amazon EC2 instance types supporting AWS Nitro Enclaves with Amazon VPC network segmentation to properly isolate confidential processing from general application logic and adhere to regulatory requirements.
2. Configure AWS KMS keys with enclave-specific policies and AWS CloudHSM for additional key protection, combined with AWS Certificate Manager for encryption and secure communication channels to establish end-to-end encryption that protects data while it's being processed within the enclave environment.
3. Build AWS Nitro Enclaves images using the **Nitro Enclaves SDK** with proper attestation mechanisms, then deploy comprehensive monitoring through Amazon CloudWatch, AWS CloudTrail, and AWS Systems Manager to track and maintain enclave operations while preserving confidentiality of the processed data.
4. Implement strict AWS IAM roles and policies with AWS Organizations for role-based access control using least privilege principles, then validate security posture using AWS Security Hub, AWS Audit Manager AWS Inspector to control enclave interactions and meet regulatory requirements.

Resources

Related best practices:

- SEC08-BP01 Implement secure key management
- SEC08-BP02 Enforce encryption at rest
- SEC05-BP02 Control traffic at all layers

- PERF02-BP01 Select the best compute options for your workload

Related documents:

- AWS Nitro Enclaves Documentation
- AWS Confidential Computing
- AWS Security Reference Architecture
- AWS Security Blog - Confidential Computing
- AWS Nitro Enclaves – Isolated EC2 Environments to Process Confidential Data

Related videos:

- AWS Nitro Enclaves Overview
- Protecting Sensitive Data with AWS Confidential Computing: Nitro System and Enclaves
- AWS re:Invent 2024 - Dive deep into the AWS Nitro System (CMP301)
- AWS Nitro Enclaves - Getting Started Video

Related services:

- AWS Nitro Enclaves
- AWS KMS
- AWS CloudHSM
- Amazon EC2

DSPERF02-BP02 Maintain sovereign control over encryption keys

Sovereign key management provides organizations full control over their encryption keys, aligning with national data residency and sovereignty laws. For regulated industries, this blocks third-party access to sensitive data, reduces legal risks, and meets mandates for country-specific regulations requiring data to remain within jurisdictional boundaries.

Desired outcome: Implement a sovereign key management architecture. Maintain exclusive control over encryption keys within designated jurisdictions. Enable compliant cloud operations while meeting regulatory requirements.

Common anti-patterns:

- Using default keys without customer-managed keys (CMKs) or FIPS 140-2/3-validated HSMs, and storing keys in non-compliant Regions.
- Mixing sovereign and non-sovereign key operations, granting overly broad IAM permissions, and lacking proper multi-factor authentication.
- Not implementing automated key rotation, proper versioning, or maintaining secure backup and recovery procedures.
- Failing to maintain proper audit trails, access logs, and compliance documentation for sovereign key operations.

Benefits of establishing this best practice:

- Meet data sovereignty, residency, and regulatory requirements (for example, GDPR or CMMC) by keeping encryption keys within specific jurisdictions.
- Maintain exclusive control over hardware-backed cryptographic keys with authorized personnel access, reducing third-party dependencies and potential attack vectors.
- Enable detailed tracking and monitoring of key operations through AWS CloudTrail and AWS Config for compliance and security analysis.
- Implement organization-specific key usage policies and access controls while maintaining cryptographic independence from cloud provider access.
- Reduce dependency on external key management services while improving continuous operation through controlled key management.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Organizations should assess regulatory requirements to determine appropriate key sovereignty needs. Implement AWS CloudHSM or AWS KMS External Key Store (XKS) in compliant Regions while providing proper redundancy and compliance-aligned procedures.

- Evaluate regulatory requirements and design high-availability architecture with appropriate geographic distribution
- Deploy CloudHSM clusters in compliant Regions and integrate with AWS KMS using custom key stores
- Implement secure key lifecycle management with automated rotation and strict access controls
- Establish comprehensive monitoring, audit logging, and compliance validation processes
- Enforce least privilege access through strict IAM policies and key usage controls

Implementation steps

1. Document regulatory and compliance requirements to determine key sovereignty needs, then design a high-availability architecture using AWS Well-Architected Framework with appropriate geographic distribution across compliant regions to verify that cryptographic keys meet data residency requirements while maintaining business continuity.
2. Deploy AWS CloudHSM clusters in compliant regions and integrate with AWS KMS using AWS KMS External Key Store (XKS) or custom key stores. This enables you to maintain control over hardware security modules while using AWS' managed encryption services for improved application integration and regulatory adherence.
3. Configure automated key rotation and lifecycle management through AWS KMS combined with strict AWS IAM policies and AWS Secrets Manager integration. This enforces least-privilege access controls managing cryptographic keys throughout their lifecycle and maintaining security and regulatory standards.
4. Enable AWS CloudTrail for audit logging, Amazon CloudWatch for monitoring, and AWS Config with AWS Audit Manager for compliance validation to maintain complete visibility into key usage. Detect unauthorized access attempts, and generate compliance reports required for regulatory audits and security oversight.

Resources

Related best practices:

- Protecting data at rest
- SEC08-BP01 Implement secure key management

Related documents:

- AWS Key Management Best Practices
- Introduction to the cryptographic details of AWS KMS
- Data Residency Whitepaper

Relate blogs:

- Establishing a European trust service provider for the AWS European Sovereign Cloud

Related videos:

- Encryption and Key Management in AWS
- Navigating the Complex World of Data Regulation and Compliance

DSPERF02-BP03 Validate open-source components for sovereignty compliance

Open-source solutions can provide essential transparency for demonstrating compliance and managing risk in highly regulated industries. These solutions offer inspectable, verifiable components. However, they require rigorous validation to verify that they meet regulatory requirements and security standards. Unvetted components could introduce vulnerabilities or compliance gaps leading to operational or regulatory issues.

Desired outcome: Open-source components meet security and regulatory standards before deployment. Audit trails provide complete traceability for sovereignty verification. Automated validation pipelines and continuous monitoring detect vulnerabilities before they impact production.

Common anti-patterns:

- Failing to audit third-party libraries, nested dependencies, and maintaining updated versions of open-source components.
- Not establishing automated processes for security scanning, vulnerability management, and compliance verification.
- Lacking clear records of component versions, sources, modifications, and compliance certifications.
- Overlooking licensing requirements and depending on single maintainers or small communities without backup plans.
- Relying on manual checks instead of systematic, automated validation procedures for open-source components.

Benefits of establishing this best practice:

- Full access to source code enables comprehensive security reviews, vulnerability management, and complete visibility into codebases.
- Demonstrate due diligence and control effectiveness through auditable solutions aligned with industry frameworks.
- Use collective expertise for rapid vulnerability response and continuous improvement.

- Avoid vendor dependency while maintaining ability to customize solutions for specific requirements.
- Reducing compliance penalties through proactive risk management.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Develop a comprehensive open-source adoption strategy that begins with technology stack inventory and establishes governance processes for evaluating and maintaining components.

Key implementation elements:

- Create evaluation criteria aligned with security and regulatory requirements
- Implement software composition analysis (SCA) tools for dependency scanning
- Integrate automated compliance checks and security testing into CI/CD pipelines
- Use Infrastructure as Code (IaC) to enforce consistency across environments
- Establish continuous monitoring for vulnerabilities and updates
- Develop internal expertise for ongoing evaluation and maintenance

This approach supports safe adoption of open-source solutions while improving security and regulatory standard adherence.

Implementation steps

1. Conduct a technology stack assessment. Document current components, dependencies, licensing requirements, and security implications. Create an inventory database and map component dependencies. Open source option to assist with creation is OCS Inventory NG or consider AWS Systems Manager Inventory can be used to expedite the process.
2. Implement security scanning with open source services like Trivy or AWS services including Amazon CodeGuru and Amazon Inspector. Set up automated scanning and integrate security checks into the CI/CD pipeline with AWS CodePipeline.
3. Develop infrastructure as code using open source options like Terraform or OpenTofu or AWS services including AWS CloudFormation and AWS CDK to create standard templates, security controls, and compliance checks.
4. Configure vulnerability monitoring with open source options like Wazuh or Falco or AWS service includingAmazon GuardDuty, AWS Security Hub, and Amazon Inspector, setting up alerts for detected vulnerabilities.

5. Implement configuration management using either Ansible or AWS Systems Manager for version control, change management, update tracking, and patch management, ensuring consistent and secure configurations across the environment.

Resources

Related best practices:

- ADVPERF01-BP05 Evaluate the choice of open source-based software (self-managed) against using fully managed service
- SEC04-BP02 Capture logs, findings, and metrics in standardized locations
- SEC06-BP01 Perform vulnerability management

Related documents:

- Open government methods, infrastructure, and tools
- Automate AWS resource assessment
- Define a vulnerability management plan

Related videos:

- AWS re:Inforce 2023 - Security in the Open: OSS and AWS (SEC201-L)
- AWS re:Invent 2024 - How to maintain and automate compliance on AWS (SEC319)

Related services:

- Amazon CodeGuru
- Amazon GuardDuty
- Amazon Inspector
- AWS Audit Manager
- AWS CDK
- AWS CloudFormation
- AWS CodePipeline
- AWS Security Hub
- AWS Systems Manager

Network and content delivery

DSPERF03: How do you consider sovereignty requirements when designing networks?

Network architectures must account for data residency restrictions and traffic routing constraints. They must also address cross-border data flow limitations imposed by various jurisdictions. This consideration is crucial for blocking compliance violations and maintaining data privacy. It verifies that network traffic patterns align with regulatory requirements. This approach delivers the necessary performance and reliability for business operations.

Best practices

- DSPERF03-BP01 Choose open protocols and approved encryption schemes
- DSPERF03-BP02 Implement continuous network verification
- DSPERF03-BP03 Design networks that can operate independently of foreign interference
- DSPERF03-BP04 Establish operational network and content delivery teams within the sovereign region
- DSPERF03-BP05 Choose service providers with well-established local presence

DSPERF03-BP01 Choose open protocols and approved encryption schemes

Using open protocols and approved encryption schemes improves your adherence to national cybersecurity standards and industry regulations. This also avoids vendor lock-in and supply-chain bottlenecks, especially where adversarial states might use their position to place technology embargoes.

Desired outcome: Implement industry-standard cryptographic controls and encryption protocols validated by certification bodies. Adhere to regulatory requirements. Adopt pre-vetted implementations for transparency and auditability.

Common anti-patterns:

- Using deprecated protocols (SSL/TLS 1.0/1.1) or implementing proprietary or custom encryption schemes instead of validated standards.

- Hardcoding encryption keys in code, failing to rotate keys regularly, and using weak key exchange protocols.
- Ignoring regulatory requirements and failing to verify encryption methods meet specific standards (FIPS or common criteria).
- Using different encryption approaches across services without a coherent strategy or documentation.
- Neglecting algorithm agility and building systems that are difficult to migrate to stronger encryption schemes.

Benefits of establishing this best practice:

- Uses community-vetted, battle-tested cryptographic implementations to reduce vulnerabilities and block data breaches.
- Adheres to industry standards through certified encryption schemes while simplifying validation.
- Enables seamless integration with third-party systems and provides algorithm agility for evolving standards.
- Reduces vendor lock-in, optimizes costs through proven implementations, and maintains flexibility across providers.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Design a comprehensive cryptographic implementation strategy using AWS services and industry standards.

Key implementation elements:

- Conduct inventory assessment of current cryptographic implementations across AWS services and applications
- Establish organizational standards based on NIST, FIPS, and industry-specific requirements
- Implement AWS KMS for centralized key management with standardized algorithms (AES-256, RSA-2048, ECC)
- Enforce strong protocols (TLS 1.2+) for data in transit across each service
- Create phased migration roadmap prioritizing high-risk areas
- Deploy continuous monitoring and compliance validation processes

This approach provides consistent, compliant cryptographic controls while minimizing operational disruption.

Implementation steps

1. Conduct an inventory assessment to document current cryptography and certificate management practices. Identify encryption methods, map key usage, and list certificate deployments. Identify third-party dependencies that could be of concern from a sovereignty point of view.
2. Implement cryptographic standards by configuring AWS KMS for FIPS 140-2 Security Level 3 compliance, NIST-approved algorithms, and industry-specific requirements.
3. Set up key management using AWS KMS to manage keys, key policies, key rotation, and access controls. Configure AWS CloudHSM when you require shared or dedicated HSM tenancy.
4. Verify transport security by configuring AWS Certificate Manager for TLS certificate management, automated renewal, and domain validation. Implement security policies to control access to certificate stores and to certificates.
5. Monitor compliance using AWS Security Hub for compliance checks, security standards, and automated validation, and configure alerts for non-compliance.

Resources

Related best practices:

- PERF04-BP05 Choose network protocols to improve performance
- SEC09-BP02 Enforce encryption in transit
- SEC08-BP01 Implement secure key management
- SEC08-BP02 Enforce encryption at rest
- SEC08-BP03 Automate data at rest protection
- SEC08-BP04 Enforce access control

Related documents:

- Supported cryptographic algorithms
- Advanced Encryption Standard (AES)
- The importance of encryption and how AWS can help

Related videos:

- AWS re:Invent 2023 - Better together: Using encryption & authorization for data protection (SEC333)
- AWS re:Inforce 2025 - Post-quantum cryptography demystified (DAP222)

Related examples:

- AWS Network Architecture Guide
- AWS Well-Architected Framework
- AWS Reference Architecture Examples
- RSA
- Transport Layer Security (TLS)

Related services:

- AWS KMS
- AWS CloudHSM
- AWS Certificate Manager
- AWS Config
- AWS Security Hub
- Amazon CloudWatch

DSPERF03-BP02 Implement continuous network verification

In highly regulated industries, maintaining integrity of network infrastructure configurations through comprehensive verification processes is crucial for security, compliance, and operational stability.

Customers should implement proactive controls to verify your networking components meet security standards and maintain compliance. Compromised or misconfigured networks can lead to data breaches, regulatory violations, and significant business disruption.

Desired outcome: Network infrastructure maintains verified security, integrity, and compliance status throughout its lifecycle with automated detection of unauthorized configuration changes.

Common anti-patterns:

- Relying on manual, infrequent checks instead of automated processes for configuration drift and unauthorized changes.

- Using default configurations, overly permissive rules, and insufficient network segmentation without zero-trust principles.
- Lacking traffic analysis, anomaly detection, and proper integration with SIEM systems.
- Missing network topology documentation, configuration baselines, and verification of third-party connections.
- Ignoring firmware and software updates and failing to validate encryption for data in transit across network segments.

Benefits of establishing this best practice:

- Continuous validation of network configurations, real-time threat detection, and automated auditing to meet regulatory standards.
- Early identification and remediation of configuration drift, improving incident response capabilities.
- Automated verification processes reduce manual overhead and identify unused or misconfigured resources.
- Improved change control, comprehensive network visibility, and automated response to security events.
- Consistent application of security policies across hybrid environments, enhancing overall security posture.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Develop a comprehensive network infrastructure verification strategy.

Key implementation elements:

- Establish configuration baselines and automated drift detection for network components
- Implement real-time monitoring, logging, and analysis of network traffic
- Deploy automated compliance checking and remediation processes
- Configure network segmentation using VPCs, security groups, and network access control lists (NACL) with least-privilege access
- Enable continuous threat detection and incident response capabilities
- Encrypt and validate network traffic

This approach provides continuous verification of network integrity while adhering to security and regulatory requirements.

Implementation steps

1. Define network resource rules using AWS Config and enable continuous recording. Consider developing your own custom rules using AWS CloudFormation guard rules or Lambda functions.
2. Set up AWS Config Rules for automated drift detection and send Amazon EventBridge alerts.
3. Implement real-time monitoring and traffic analysis by enabling Amazon VPC Flow Logs for VPCs and subnets, configure Amazon CloudWatch for metrics, and use Amazon Detective for traffic analysis with AWS Security Hub providing a unified security view.
4. Configure network segmentation and security controls by designing proper Amazon VPC architecture with separate environments, implement least-privilege Security Groups and NACLs, and deploy AWS Network Firewall for stateful inspection.
5. Enable threat detection and incident response by implementing Amazon GuardDuty across each region for threat detection, configure AWS Systems Manager Incident Manager for response coordination, and use AWS Lambda for automated remediation actions.
6. Establish traffic encryption and validation using Site-to-Site VPN for external connections, implement AWS PrivateLink for service connections, configure TLS for applications, and deploy AWS WAF (Web Application Firewall) for application layer protection.
7. Implement continuous auditing and reporting by setting up AWS CloudTrail with integrity validation across regions, use Amazon Athena for log analysis, create dashboards with Quick Suite, and manage access control through AWS IAM and AWS Organizations.

Resources

Related best practices:

- PERF04-BP02 Evaluate available networking features
- PERF04-BP07 Optimize network configuration based on metrics
- SEC05-BP04 Automate network protection

Related documents:

- Detect drift on individual stack resources
- Detect drift on an entire CloudFormation stack
- Set up AWS CloudFormation drift detection in a multi-Region, multi-account organization
- Amazon VPC Flow Logs

Related videos:

- AWS re:Inforce 2025 - Securing AWS networks: Observability meets defense-in-depth (NIS306)
- AWS re:Invent 2022 Dive deep on AWS networking infrastructure (NET402)

Related services:

- Amazon Athena
- Amazon CloudWatch
- Amazon Detective
- Amazon EventBridge
- Amazon GuardDuty
- Quick Suite
- Amazon VPC
- AWS CloudTrail
- AWS Config
- AWS IAM
- AWS Lambda
- AWS Network Firewall
- AWS Organizations
- AWS PrivateLink
- AWS Security Hub
- AWS Systems Manager Incident Manager
- AWS WAF
- Site-to-Site VPN

DSPERF03-BP03 Design networks that can operate independently of foreign interference

In highly regulated industries, establishing network independence from foreign interference is crucial for maintaining data sovereignty, regulatory adherence, and operational resilience.

Organizations should design robust, autonomous network architectures that minimize external exposure and protect sensitive data. This improves business continuity during connectivity disruptions. It also blocks unauthorized access that could lead to compliance violations or operational disruptions.

Desired outcome: Network architecture is self-contained and resilient, enforcing data sovereignty while maintaining operational capabilities with minimal external risks.

Common anti-patterns:

- Over-reliance on foreign CDNs, DNS providers, and third-party services without consideration of disruptions.
- Routing traffic containing sensitive data through foreign-controlled infrastructure without proper isolation or redundancy.
- Weak access controls and overly permissive security groups that don't properly distinguish or restrict foreign network sources or destinations.
- Insufficient monitoring, logging, and encryption of network traffic, particularly for identifying foreign interference attempts.
- Storing or transferring sensitive data through unapproved regions without proper encryption or compliance validation.

Benefits of establishing this best practice:

- Control over network traffic flows while meeting domestic security requirements and data residency regulations.
- Strengthened security posture through network isolation and comprehensive visibility into potential interference.
- Reduced dependencies on foreign infrastructure while maintaining business operations during international disruptions.
- Traffic is routed through approved regions and services while isolating critical workloads from public networks.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Design a multi-layered, independent network architecture using AWS services.

Key implementation elements:

- Deploy localized DNS resolution and redundant connectivity paths aligned with compliance requirements
- Implement strict network segmentation to isolate critical workloads from untrusted traffic

- Configure comprehensive traffic inspection and monitoring to detect interferences
- Establish private connections using AWS Direct Connect and VPC peering
- Default to using private connections to avoid internet-based routing where possible

Implementation steps

1. Configure DNS and network foundation using Amazon Route 53 with private hosted zones, health checks, and failover routing. Design Amazon VPC architecture with separate security zones and data-sensitivity-based isolation for workload separation.
2. Implement connectivity and traffic management using AWS Direct Connect with redundant connections and private VIFs. Configure VPC Peering, and deploy AWS Network Firewall for stateful inspection and custom traffic filtering rules.
3. Configure security controls and access management using security groups, network ACLs, AWS WAF (Web Application Firewall), and AWS Shield for comprehensive protection.
4. Deploy transit architecture and private connectivity using AWS Transit Gateway with route tables and security domains for cross-region connectivity. Set up AWS PrivateLink with service endpoints and interface endpoints for secure private access.
5. Implement network monitoring and threat detection using VPC Flow Logs, Amazon CloudWatch, and AWS CloudTrail. Configure Amazon GuardDuty and AWS Security Hub for automated threat detection and security insights.
6. Establish compliance and operational excellence using AWS Config rules for compliance monitoring and automated remediation.
7. Maintain comprehensive documentation including network diagrams, disaster recovery procedures, and continuous performance monitoring with security alerts and health checks.

Resources

Related best practices:

- PERF04-BP06 Choose your workload's location based on network requirements
- DRHCOPS03-BP03 Build redundant network connectivity
- ADVPERF01-BP01 Design geographical affinity architecture with external entities (DSPs and SSPs)

Related documents:

- Network ACLs

- Security Groups
- VPC Flow Logs

Related videos:

- Inside AWS Networking: Building a Secure, Reliable, and High-Performance Global Infrastructure

Related services:

- Amazon CloudWatch
- Amazon GuardDuty
- Amazon Route 53
- Amazon VPC
- AWS CloudTrail
- AWS Config
- AWS Direct Connect
- AWS IAM
- AWS Network Firewall
- AWS PrivateLink
- AWS Security Hub
- AWS Shield
- AWS Transit Gateway
- AWS WAF
- VPC Peering

DSPERF03-BP04 Establish operational network and content delivery teams within the sovereign region

In regulated industries, establishing dedicated operational network and content delivery teams within sovereign regions is highly desired. This enables greater oversight and control over infrastructure access.

Organizations should consider incorporating specific personnel vetting requirements, supplier preferences, and geographical location requirements into contracts. This approach provides operational autonomy while reducing external risks.

Desired outcome: Dedicated, self-sufficient operational network and content delivery teams within sovereign regions autonomously manage AWS infrastructure with regulatory adherence and reduced external risks.

Common anti-patterns:

- Relying on teams outside sovereign regions for critical operations and allowing non-resident personnel administrative access.
- Insufficient local expertise of AWS services and regulatory requirements, with inadequate training programs.
- Using global operations centers and storing critical data (logs and backups) outside sovereign regions.
- Insufficient regulatory training and deployments without proper region-specific compliance checks.

Benefits of establishing this best practice:

- Local teams align with regional regulations while maintaining localized audit trails for simplified reporting.
- Complete control over infrastructure management decisions. Faster incident response times and reduced external dependencies.
- Personnel subject to local jurisdiction and security clearance requirements, limiting unauthorized access.
- Demonstrates commitment to network and content delivery sovereignty while building trust with regulators and customers.

Level of risk exposed if this best practice is not established: High

Implementation guidance

Develop a comprehensive strategy for establishing sovereign region operational network and content delivery teams using AWS services and trusted third-party offerings.

Key implementation elements:

- Conduct skills assessment and define clear organizational structure with roles and responsibilities
- Implement AWS IAM policies to enforce regional boundaries and access controls

- Create robust training programs combining AWS certifications with compliance requirements
- Establish detailed operational network and content delivery procedures, runbooks, and knowledge transfer processes

Implementation steps

1. Define team hierarchy, role definitions, and responsibilities matrix documented through AWS Systems Manager. Configure AWS IAM with role-based access policies, permission boundaries, and Regional restrictions. Apply AWS Organizations service control policies for centralized governance.
2. Develop comprehensive operational network and content delivery documentation including standard operating procedures, runbooks, and troubleshooting guides. Develop a knowledge management system with documentation repositories, best practices libraries, and version-controlled training materials. Consider building retrieval-augmented generation (RAG) and graph-based retrieval-augmented generation (GraphRAG) with Amazon Bedrock Knowledge Bases.
3. Create structured training programs covering AWS certification paths, regulatory requirements, and Regional operating procedures. Establish a skills assessment framework with technical competency matrices, development paths, and certification tracking to verify operational network and content delivery readiness and continuous professional growth.
4. Set up compliance and incident management using AWS Audit Manager for regional requirements. Configure AWS Systems Manager Incident Manager with defined response procedures, team responsibilities, and communication protocols for effective incident resolution.
5. Develop automated workflows using AWS Lambda, AWS Step Functions, and AWS Systems Manager Automation to streamline operations. Set up quality assurance processes with review procedures, and audit mechanisms for continuous improvement.
6. Implement comprehensive performance monitoring with KPI tracking, service level metrics, and team performance measurements. Establish feedback mechanisms, process refinement procedures, and reporting frameworks with performance dashboards and compliance reports to drive continuous organizational improvement and operational excellence.

Resources

Related best practices:

- OPS02-BP06 Responsibilities between teams are predefined or negotiated
- OPS03-BP02 Team members are empowered to take action when outcomes are at risk

- OPS03-BP07 Resource teams appropriately
- OPS07-BP03 Use runbooks to perform procedures

Related documents:

- AWS Security Best Practices for Security, Identity and Compliance
- Data Residency Whitepaper
- Permission boundaries for IAM entities

Related services:

- AWS Audit Manager
- AWS IAM
- AWS Lambda
- AWS Organizations
- AWS Step Functions
- AWS Systems Manager
- AWS Systems Manager Incident Manager

DSPERF03-BP05 Choose service providers with well-established local presence

Customers in highly-regulated industries should comply with local data sovereignty laws, industry-specific regulations (such as GDPR and HIPAA), and operational requirements. Service providers with strong local presence align with regional compliance frameworks, reduce latency for critical workloads, and provide faster incident response. This can reduce legal risks and operational disruptions while maintaining trust with regulators and customers.

Desired outcome: Lasting partnerships established with service providers demonstrating strong local presence, expertise in AWS technologies and knowledge of regional compliance requirements. Cloud operations are compliant and efficient while meeting data residency and aligning with cultural norms.

Common anti-patterns:

- Selecting providers without verifying local regulatory knowledge, certifications, and compliance track record.

- Overlooking data residency requirements and contractual obligations for local data protection.
- Choosing providers without local language capabilities or cultural understanding.
- Partnering with providers who lack proper audit trails, compliance documentation, and local reporting capabilities.
- Relying on distant teams without clear local escalation paths for compliance issues or regulatory inquiries.

Benefits of establishing this best practice:

- Deep understanding of regional laws and regulations, reducing violations and penalties while building stakeholder trust.
- Improves proper data handling practices and reduced latency for region-specific workloads.
- Faster issue resolution through local presence and improved disaster recovery capabilities.
- Better understanding of local practices and established relationships with regulatory bodies.
- Reduces compliance consulting needs and audit preparation overhead through local expertise.

Level of risk exposed if this best practice is not established: Medium

Implementation guidance

Develop a comprehensive strategy for selecting and managing local service providers through structured evaluation and governance frameworks.

Key implementation elements:

- Conduct regulatory assessment and map requirements to technical and operational controls
- Use AWS Partner Network to identify providers with demonstrated local presence and certifications
- Evaluate partners based on infrastructure capabilities, technology expertise, and cultural alignment
- Create standardized compliance questionnaires and performance metrics
- Establish clear governance frameworks defining roles and escalation procedures
- Implement regular review cycles and joint governance structures for ongoing collaboration

This approach supports effective partnerships that maintain regulatory adherence while meeting operational requirements.

Implementation steps

1. Document local laws, data sovereignty mandates, and industry-specific compliance requirements while creating a comprehensive compliance matrix. Consider mapping compliance requirements using AWS Audit Manager to technical and operational controls. Collect evidence, perform gap analysis, and align with regulatory frameworks.
2. Use APN Partner Central to search for local partners, review competencies and certifications, and evaluate case studies. Create objective selection criteria covering infrastructure capabilities, local expertise, cultural alignment, and compliance track records with standardized scoring systems for provider selection.
3. Design standardized compliance questionnaires covering security controls, data protection measures, and incident response procedure. Document governance frameworks including roles, responsibilities, and escalation procedures in AWS Systems Manager for centralized management.
4. Use the AWS Marketplace for standardized terms, SLA definitions, and compliance requirements. Create comprehensive onboarding procedures including kickoff processes, knowledge transfer plans, and provisioning documents.
5. Configure ongoing monitoring and incident management using Amazon CloudWatch for performance metrics and AWS Config for compliance tracking. Support incident management through AWS Systems Manager Incident Manager with escalation procedures and root cause analysis processes.
6. Establish joint governance structures with steering committees, operational reviews, and compliance audits, while maintaining comprehensive documentation. Maintain provider profiles, compliance records, and performance reports, supported by continuous improvement mechanisms with feedback collection and best practice sharing.

Resources

Related best practices:

- DRHCOPS01-BP01 Understand your organization's specific legal and compliance requirements specific to data residency

Related documents:

- AWS Partner Network Guide
- Regional Service Delivery Best Practices

- AWS Local Zones Implementation Guide
- Cloud Adoption Framework (CAF)

Related services:

- Amazon CloudWatch
- AWS APN Partner Central
- AWS Artifact
- AWS Audit Manager
- AWS Config
- AWS Marketplace
- AWS Systems Manager
- AWS Systems Manager Incident Manager

Cost optimization

The Digital Sovereignty Lens does not add any new best practices to the cost optimization pillar. Best practices already defined under the cost optimization pillar provide adequate guidance.

Related cost pillar best practices

- COST01-BP05 Report and notify on cost optimization
- COST01-BP07 Keep up-to-date with new service releases
- COST01-BP08 Create a cost-aware culture
- COST04-BP02 Implement a decommissioning process
- COST04-BP05 Enforce data retention policies
- COST08-BP01 Perform data transfer modeling
- COST08-BP02 Select components to optimize data transfer cost
- COST08-BP03 Implement services to reduce data transfer costs
- COST10-BP01 Develop a workload review process

Sustainability

The Digital Sovereignty Lens does not add new best practices to the sustainability pillar. Best practices already defined under the sustainability pillar provide adequate guidance.

Conclusion

The Digital Sovereignty Lens provides architectural best practices across four of the six pillars of the AWS Well-Architected Framework. The Framework provides a set of questions that allows you to review an existing or proposed architecture. It also provides a set of AWS best practices for each pillar. Using the Framework in your sovereign workloads assists you when building compliance-aligned, secure, and reliable systems and allows you to focus on your functional requirements.

Contributors

The following individuals and organizations contributed to this document:

- Swapnonil Mukherjee, Senior Partner Solutions Architect - Sovereignty, Amazon Web Services
- Arfan Azmir, WW Consulting Sales SA, Amazon Web Services
- Michelle Peterson, Security Partner Strategist, World Wide Public Sector, Amazon Web Services
- Gregory Carpenter, Senior Security Partner Strategist, World Wide Public Sector, Amazon Web Services
- Damian Randell, Senior Partner Solutions Architect, Amazon Web Services
- Jonathan Tate, Senior Partner Solutions Architect, Amazon Web Services
- Armin Schneider, Digital Sovereignty Specialist Solutions Architect, Amazon Web Services
- Mayssa Haddad, Senior Manager, World Wide Public Sector - Specializations, Amazon Web Services
- Florian Hettenbach, COS Tech BD, EMEA, AWS Field Enablement, Amazon Web Services
- Stewart Matzek, Senior Technical Writer, Amazon Web Services
- Matthew Wygant, Senior TPM Guidance, Amazon Web Services
- Madhuri Srinivasan, Senior Technical Writer, Amazon Web Services

Document revisions

The following table describes the documentation releases for the Digital Sovereignty Lens.

Change	Description	Date
Initial release	Initial release of the Digital Sovereignty Lens.	January 26, 2026

Notices

AWS Glossary

For the latest AWS terminology, see the AWS glossary in the *AWS Glossary Reference*.