AWS Security Reference Architecture (AWS SRA) – AI security

AWS Prescriptive Guidance



Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Prescriptive Guidance: AWS Security Reference Architecture (AWS SRA) – AI security

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

Introduction **1**
Intended audience 1
Attachments 2
About the AWS SRA library **3**
AWS SRA – core architecture guide 3
AWS SRA – deep dive architectures 3
AWS SRA for AI **5**
Amazon Bedrock 8
Amazon Bedrock Guardrails 9
Security 9
Amazon Bedrock model evaluation 9
Security 10
Amazon Bedrock AgentCore 10
Generative AI capabilities **12**
Capability 1. Model inference 13
Rationale 13
Security considerations 14
Multi-account architecture for AI workloads 16
Defense in depth 20
Model evaluation and validation 24
Capability 2. Model customization 25
Rationale 26
Security considerations 28
Remediations 28
Recommended AWS services 29
Capability 3. RAG 31
Rationale 33
Security considerations 35
Remediations 35
Recommended AWS services 38
Capability 4. Tools 41
Rationale 42
Security considerations 43
Remediations 44

Recommended AWS services ... 46
Capability 5. Generative AI agents ... 48
Rationale ... 49
Security considerations ... 50
Remediations ... 51
Recommended AWS services ... 54
Capability 6. AI applications ... 58
Rationale ... 59
Security considerations ... 60
Remediations ... 60
Recommended AWS services ... 63
Integrating a traditional cloud workload with Amazon Bedrock ... 67
Application account ... 68
Identity propagation and access control ... 68
Application security ... 68
Prompt injection protection ... 69
Data protection ... 69
Network security ... 69
Logging and monitoring ... 70
Generative AI account ... 70
Foundation model inference ... 71
Model customization ... 71
Knowledge bases and RAG ... 71
Tool integration ... 72
Autonomous agents ... 72
End-user AI applications ... 72
Conclusion ... 73
Contributors ... 74
Document history ... 75

AWS Security Reference Architecture (AWS SRA) – AI security

Avik Mukherjee, Amazon Web Services

Influence the future of the AWS Security Reference Architecture (AWS SRA) by taking a short survey.

AI solutions span multiple use cases, each with distinct security requirements. The Generative AI Security Scoping Matrix defines security scope and disciplines for different use cases, and the Agentic AI Security Scoping Matrix defines the autonomy and agency given to an agent.

Depending on your use case, you can use a managed service where the provider handles operations, or build your own. AWS offers a wide range of services to help you build, run, and integrate artificial intelligence and machine learning (AI/ML) solutions of any size, complexity, or use case. These services operate at all three layers of the generative AI stack: infrastructure, large language models (LLMs), and applications.

This guide focuses on the middle layer, which provides access to all the models and tools you need to build and scale generative AI applications and applications on AWS. Although AI (machine learning and LLMs) can be used for security purposes, this guide focuses on the foundational security controls to protect AI workloads deployed on AWS.

Intended audience

The intended audience for this guidance is security professionals, architects, and developers who are responsible for securely integrating generative AI capabilities into their organizations and applications using AWS services.

In this guide:

- About the AWS SRA library
- AWS SRA for AI
- Generative AI capabilities
- Integrating a traditional cloud workload with Amazon Bedrock

- Conclusion

Attachments

To access additional content that is associated with this document, download and unzip the following file:

attachment.zip

About the AWS SRA library

Influence the future of the AWS Security Reference Architecture (AWS SRA) by taking a short survey.

This guide is part of a library that provides architectural blueprints and technical guidance for designing and building security architectures on AWS. The library consists of implementation code (AWS SRA code library), a validation tool (SRA Verify), and two complementary categories of guides that cover the core architecture and deep dive architectures.

AWS SRA – core architecture guide

The AWS SRA – core architecture guide represents a foundation for the recommended AWS security architecture. It is the starting point that applies to all organizations, regardless of their industry, application type, or any other considerations. This foundation helps you build a strong and scalable architecture on AWS and helps create a strong AWS multi-account security baseline that scales securely as your business grows.

AWS SRA – deep dive architectures

The *AWS SRA – core architecture* guide is complemented by additional publications that provide architectural patterns aligned to specific security capabilities, application types, and compliance or regulatory requirements. These patterns extend the core architecture and should be used with the *AWS SRA – core architecture* guide.

The following guides provide architectural patterns aligned to specific security capabilities:

- AWS SRA – identity management provides guidance on how to implement a scalable, robust, and centralized identity and access management solution on AWS.
- AWS SRA – perimeter security discusses architecture patterns and AWS services for implementing edge security in a central account or in individual accounts.
- AWS SRA – cyber forensics describes how to configure an AWS Forensics account as a starting point to develop your organization's forensic capabilities and to help improve your security incident response (IR) preparedness.

The following guides provide architectural patterns for specific application types. You might want to focus on these guides after you build your baseline security architecture:

- *AWS SRA – AI security* (this guide) provides security architectural recommendations to protect AI workloads deployed on AWS.
- AWS SRA – IoT provides security architectural recommendations for designing and building IoT applications on AWS.

In addition, the following guide describes architectural patterns that are aligned with specific compliance or regulatory frameworks:

- AWS Privacy Reference Architecture (AWS PRA) provides a security architecture for applications that process personal data and must support broad privacy compliance requirements such as the General Data Protection Regulation (GDPR), the California Consumer Privacy Act (CCPA), or the Brazilian General Data Protection Law (LGPD). The AWS PRA provides a set of guidelines that are specific to the design and configuration of privacy controls in AWS services.

We recommend that you start with the *AWS SRA – core architecture* guide to understand the foundational architecture. Then consult the complementary guides to take advantage of advanced functionality and implementations. For more information about this content set, see AWS Security Reference Architecture.

> **Note**
>
> To customize the reference architecture diagrams in the AWS SRA library based on your business needs, you can download the following .zip file and extract its contents.

Download the diagram source file (Microsoft PowerPoint format)

AWS SRA for AI

Influence the future of the AWS Security Reference Architecture (AWS SRA) by taking a short survey.

This section provides recommendations for securing AI workloads within the AWS SRA multi-account framework. It covers AWS Identity and Access Management (IAM) permissions, data protection, input/output validation, network isolation, logging, and monitoring for generative AI capabilities. This section also covers integrating these capabilities into traditional AWS workloads.

This guidance covers security for the following capabilities:

- Capability 1. Providing developers and data scientists with secure access to generative AI FMs (model inference)
- Capability 2. Providing secure access, usage, and implementation for generative AI model customization
- Capability 3. Providing secure access to data and systems for generative AI
- Capability 4. Providing secure access, usage, and implementation of tools
- Capability 5. Providing secure access, usage, and implementation of generative AI agents
- Capability 6. Providing secure access, usage, and implementation for AI applications

Most capability sections include the following information:

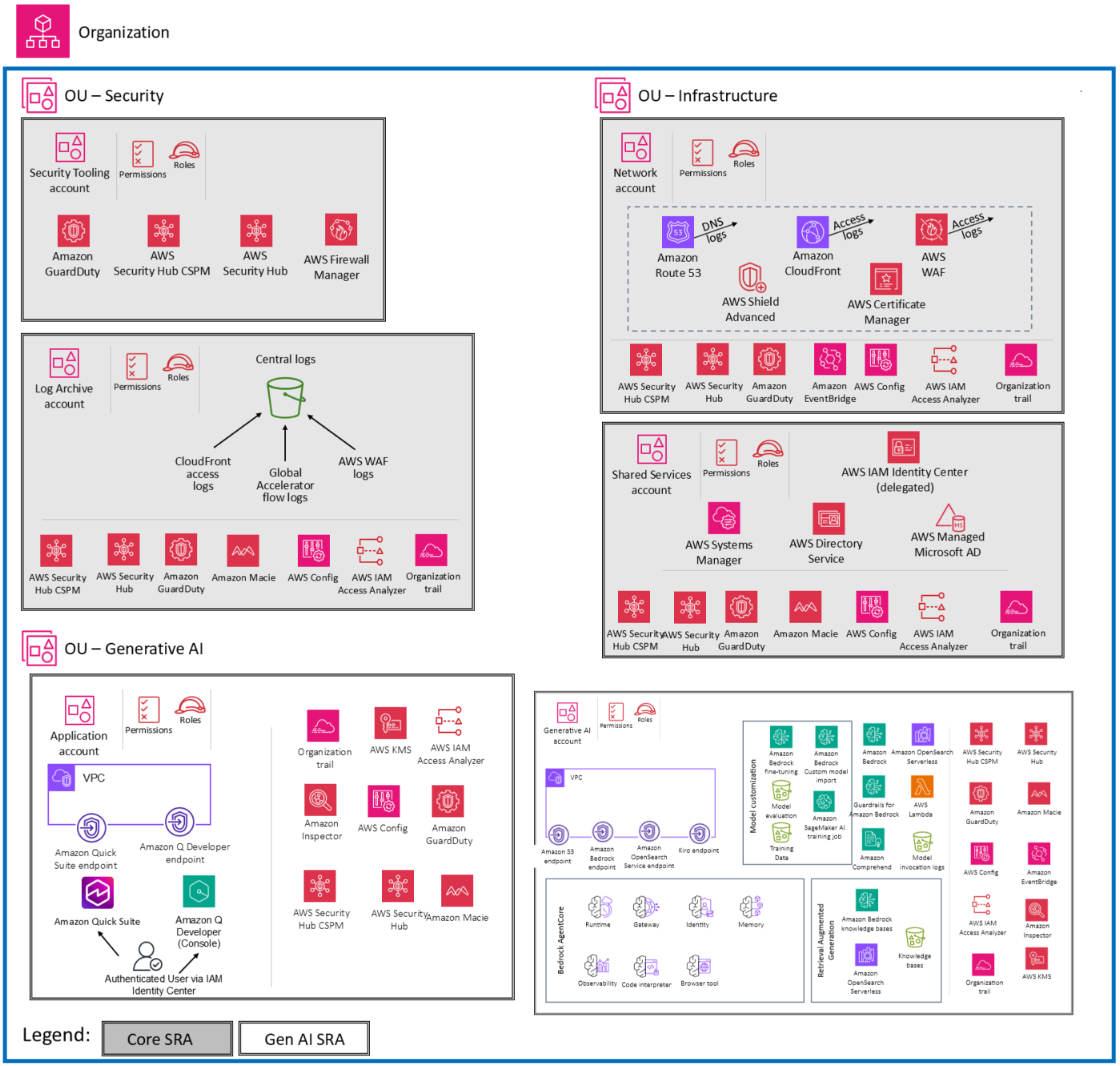
- **Rationale** explains what the capability does and when to use it.
- **Security considerations** describes risks that are specific to the capability.
- **Remediations** reviews the AWS services and features that address the risks.
- **Recommended AWS services** describes the services to build the capability securely.

All capabilities build on Capability 1 (foundation model inference) because they all invoke models. When you combine capabilities, apply security controls from each relevant section. For example, a customized model with Retrieval Augmented Generation (RAG) requires controls from capabilities 1, 2, and 3.

The following diagram shows the extension of the AWS SRA Workloads organizational unit (OU) architecture with a dedicated Generative AI OU. The Generative AI OU contains two accounts that separate concerns:

- The *Application account* hosts your traditional AWS application, which provides specific business functionality.
- The *Generative AI account* hosts Amazon Bedrock and associated AWS services. Your application in the Application account calls Amazon Bedrock capabilities in the Generative AI account through APIs.

This account separation provides two key benefits. First, it enforces security controls through OU-specific and account-specific service control policies. Second, it simplifies implementing least-privilege access by grouping services based on application type. The reference architecture also includes foundational accounts that apply to all application types: Org Management, Security Tooling, Log Archive, Network, and Shared Services. For more information about these accounts, see AWS Security Reference Architecture in the *AWS SRA – core architecture* guide.



Design considerations:

- Merge the Application and Generative AI accounts if your architecture requires consolidating both services in one account, or if your generative AI usage spans your entire organization.
- Separate Generative AI accounts by software development lifecycle (SDLC) environment (development, test, and production) or by model and user community:

  - Use separate OUs for each SDLC environment. This approach controls team access, isolates resources, tracks costs separately, and prevents development issues from affecting production.
  - Use AWS Identity and Access Management (IAM) roles in a single account for pre-trained models. Use separate accounts when user communities have different risk levels, or when customized models contain sensitive training data.
- Use multiple accounts when different user communities have distinct risk profiles, when customized models contain sensitive training data, or when regulatory requirements mandate data isolation.
- Use a single account when you only use pre-trained foundation models, when users have a consistent risk profile, or when IAM roles provide sufficient access control.

> **Note**
>
> This guide assumes a single generative AI account strategy with IAM roles.

Amazon Bedrock

Amazon Bedrock helps you build and scale generative AI applications with foundation models (FMs). As a fully managed service, it provides access to FMs from companies such as AI21 Labs, Anthropic, Cohere, Meta, Stability AI, and Amazon through a single API. You can experiment with pre-trained models, customize them with your data, and deploy them into your applications without training models from scratch or managing specialized infrastructure.

Amazon Bedrock protects your data through the following security capabilities:

- **Data protection** – Amazon Bedrock isolates your content by user, encrypts it at rest in your AWS Region, and encrypts it in transit using TLS 1.2 or higher.
  - **Data privacy** – Amazon Bedrock doesn't store or log your prompts and completions. AWS doesn't use your content to train models or share it with third parties.
  - **Model customization** – When you customize a foundation model, your changes use a private copy. Your data isn't shared with model providers or used to improve base models.
  - **Abuse detection** – Amazon Bedrock uses automated detection to identify potential violations of the AWS Responsible AI Policy.

- **Compliance** – Amazon Bedrock meets these standards: International Organization for Standardization (ISO), System and Organization Controls (SOC), Federal Risk and Authorization Management Program (FedRAMP) Moderate, and Cloud Security Alliance (CSA) Security Trust Assurance and Risk (STAR) Level 2. Amazon Bedrock is Health Insurance Portability and Accountability Act (HIPAA) eligible and in compliance with the General Data Protection Regulation (GDPR).

For more information, see the AWS secure approach to generative AI.

Amazon Bedrock Guardrails

Amazon Bedrock Guardrails help you implement safeguards that align with your use cases and responsible AI policies. You configure filters to block harmful content, define restricted topics, and customize messages that users see when content is blocked.

Guardrails evaluate both user inputs and model outputs across multiple harmful categories (hate, insults, sexual, violence, misconduct, and prompt attacks). The system classifies each input and output into one of four confidence levels: none, low, medium, or high. You set filter strength for each category based on your risk tolerance. When you deploy a guardrail to production, you create a versioned instance and invoke it in your application through the Amazon Bedrock API. For more information about implementation steps, see the Guardrails section in the Amazon Bedrock documentation.

Security

By default, Amazon Bedrock encrypts guardrails with an AWS managed key in AWS Key Management Service (AWS KMS). We recommend that you use a customer managed key to encrypt your guardrails and prevent unauthorized modifications. Combine encryption with least-privilege IAM permissions to restrict who can view or modify guardrail configurations.

Amazon Bedrock model evaluation

Model evaluation helps you compare FM outputs and select the model that best fits your application requirements. Amazon Bedrock supports both automatic evaluation using prompt datasets and human evaluation with subject matter experts.

Automatic evaluation jobs assess model performance using either custom prompt datasets or built-in datasets. This approach provides quantitative metrics for comparing models at scale. For more

information, see Creating an automatic model evaluation job and Use prompt datasets for model evaluation in the Amazon Bedrock documentation.

Human evaluation jobs incorporate input from employees or subject matter experts to assess model quality, relevance, and safety. You organize evaluators into work teams managed through Amazon SageMaker Ground Truth. Create and manage work teams during job setup in Amazon Bedrock, or use the Amazon Cognito or Amazon SageMaker Ground Truth consoles for ongoing management.

Security

We recommend that you run model evaluation in development environments, not production. For information about multi-account strategies, see the Organizing your AWS environment using multiple accounts whitepaper. Both evaluation types require IAM service roles with the following specific permissions:

- Automatic evaluation jobs require permissions to access Amazon Simple Storage Service (Amazon S3) datasets and write results.
- Human evaluation jobs require additional permissions for Amazon SageMaker Ground Truth integration.
- Custom prompt datasets require cross-origin resource sharing (CORS) configuration on Amazon S3 buckets.

For more information, see Service role requirements for automatic model evaluation jobs and Service role requirements for human-based model evaluation jobs in the Amazon Bedrock documentation.

Amazon Bedrock creates a temporary copy of your evaluation data during the job and deletes it after completion. By default, Amazon Bedrock encrypts this data with an AWS managed key. We recommend that you use a customer managed key in AWS KMS for enhanced control over data access. For more information, see Data management and encryption in Amazon Bedrock evaluation job in the Amazon Bedrock documentation.

Amazon Bedrock AgentCore

Amazon Bedrock AgentCore provides infrastructure and controls for deploying AI agents securely at scale. AgentCore consists of the following modular services that work together or independently:

- **AgentCore Runtime** – Serverless execution environment with session isolation
- **AgentCore Identity** – Authentication and credential management for agents and tools
- **AgentCore Memory** – Persistent storage for conversation history and context
- **AgentCore Gateway** – Centralized API integration and tool access
- **AgentCore Policy** – Policy-based governance for agent actions
- **AgentCore tools** – Built-in sandboxed tools (Code Interpreter, Browser)
- **AgentCore Observability** – Trace, debug, and monitor agent performance
- **AgentCore Evaluations** – Assess agent quality through online and on-demand testing

AgentCore works with any agent framework (for example, CrewAI, LangGraph, LlamaIndex, and Strands Agents) and any foundation model (FM). AgentCore provides security capabilities including session isolation through dedicated micro virtual machines (microVMs), encrypted credential storage, namespace-based memory isolation, and comprehensive audit logging.

For detailed security guidance, see Capability 5.

Generative AI capabilities

Influence the future of the AWS Security Reference Architecture (AWS SRA) by taking a short survey.

This section discusses secure access, usage, and implementation recommendations for the following generative AI capabilities:

- Capability 1. Providing developers and data scientists with secure access to generative AI FMs (model inference)
- Capability 2. Providing secure access, usage, and implementation for generative AI model customization
- Capability 3. Providing secure access to data and systems for generative AI
- Capability 4. Providing secure access, usage, and implementation of tools
- Capability 5. Providing secure access, usage, and implementation of generative AI agents
- Capability 6. Providing secure access, usage, and implementation for AI applications

Most capability sections include the following information:

- **Rationale** explains what the capability does and when to use it.
- **Security considerations** describes risks that are specific to the capability.
- **Remediations** reviews the AWS services and features that address the risks.
- **Recommended AWS services** to build the capability securely.

All capabilities build on Capability 1 (foundation model inference) because they all invoke models. When you combine capabilities, apply security controls from each relevant section. For example, a customized model with Retrieval Augmented Generation (RAG) requires controls from Capabilities 1, 2, and 3.

Capability 1. Providing developers and data scientists secure access to generative AI FMs (model inference)

Influence the future of the AWS Security Reference Architecture (AWS SRA) by taking a short survey.

Organizations building AI-powered applications must understand the fundamental differences between traditional AI systems and generative AI foundation models (FMs). Traditional AI systems perform classification, prediction, or optimization tasks with consistent outputs. Generative AI creates new content (text, images, code, or other media) based on learned patterns from training data. FMs are large-scale neural networks trained on vast datasets that generate probabilistic outputs, meaning identical inputs can produce different responses across invocations. This non-deterministic behavior requires security architectures that account for output variability while maintaining consistent protection.

Building applications that integrate generative AI FMs and agent capabilities enables advanced functionality, including natural language processing (NLP), image generation, automated reasoning, and intelligent decision support. This integration drives organizational innovation by allowing developers to build solutions that improve productivity and competitive positioning. However, the probabilistic nature of AI outputs demands security controls that function effectively regardless of model response variability.

Rationale

This use case corresponds to Scope 3 of the Generative AI Security Scoping Matrix. In Scope 3, your organization builds an application or feature that integrates generative AI by using pre-trained FMs, such as those offered on Amazon Bedrock. You control your application and any customer data used by your application, whereas the FM provider controls the pre-trained model and its training data. For data flows pertaining to various application scopes and information about the shared responsibility between you and the FM provider, see Securing generative AI: Applying relevant security controls (AWS blog post).

Organizations can also implement custom AI solutions using Amazon SageMaker AI for model development, training, and deployment. This approach introduces additional security considerations including secure model development environments, protection of training data and model artifacts, and governance of the entire machine learning lifecycle.

Custom models require enhanced monitoring for model drift, bias detection, and performance degradation that could indicate security issues or model compromise. When you customize FMs with your own training data (Scope 4) or train models from scratch (Scope 5), training data security becomes critical. Malicious or poisoned training data can compromise model behavior, introduce bias, or cause models to leak sensitive information during inference. For detailed guidance on securing model customization and training data, see Capability 2.

The security architecture must address both the non-deterministic nature of AI systems and the autonomous capabilities of AI agents. The security architecture must implement layered defenses that maintain effectiveness across the spectrum of possible AI behaviors and outputs.

Security considerations

AI workloads introduce unique attack vectors and operational risks that traditional security controls don't address. Unlike conventional applications with predictable input-output relationships, AI systems process natural language and generate probabilistic responses that attackers can influence through carefully crafted inputs.

Model-specific risk

These risks target the AI model itself, exploiting the probabilistic nature of neural networks and their training methodologies. Attackers can manipulate model behavior without traditional code injection, instead using carefully crafted natural language inputs to achieve malicious outcomes. Risks include the following:

- Resource exhaustion through crafted prompts that trigger excessive token generation
- Data exfiltration through prompt engineering techniques that extract training data or fine-tuning information
- Model behavior manipulation through adversarial inputs designed to bypass safety mechanisms

Application layer risks

AI applications face unique challenges in validating and securing the interface between human users, AI models, and downstream systems. Traditional application security assumes deterministic behavior with predictable input-output relationships, but AI outputs require dynamic validation strategies that can assess content quality, safety, and appropriateness in real-time. Applications must handle scenarios where models generate syntactically valid but semantically problematic

outputs. Examples of such outputs include hallucinated information presented as fact, biased responses that reflect training data patterns, or outputs that inadvertently reveal system architecture details.

The integration of AI into existing application workflows introduces risks when downstream systems consume model outputs without proper validation. This situation can potentially lead to automated execution of flawed recommendations or propagation of incorrect information through business processes. Additionally, conversational AI applications maintain complex session state across multiple interactions, creating opportunities for session manipulation, context poisoning, and unauthorized access to conversation history containing sensitive information.

A systems-thinking approach reveals deeper interdependencies where AI application risks cascade across system boundaries. Model outputs influence not just immediate application behavior but also training data for future models, decision-making processes, and user trust relationships. Security failures at the application layer can create feedback loops where compromised outputs become trusted inputs, gradually degrading system integrity over time.

The temporal nature of AI interactions means that security decisions must account for both immediate threats and long-term systemic impacts. These impacts include how model behaviors evolve through user interactions and how application-level vulnerabilities might be exploited across multiple sessions or user contexts, such as:

- Unvalidated model outputs being passed to downstream systems
- Context injection where malicious content in Retrieval Augmented Generation (RAG) sources influences model behavior
- Session hijacking in conversational AI applications with inadequate state management
- Missing rate limiting enabling resource exhaustion and denial of service attacks
- Inadequate authentication and authorization for model access endpoints
- Insecure storage of conversation history and user interaction data
- Cascading failures when AI-generated content triggers errors in downstream business logic
- Model output caching creating stale or contextually inappropriate responses
- Feedback loop contamination where AI outputs become training data without validation
- Compound security issues where multiple minor issues combine to create potential security issues

Data governance risks

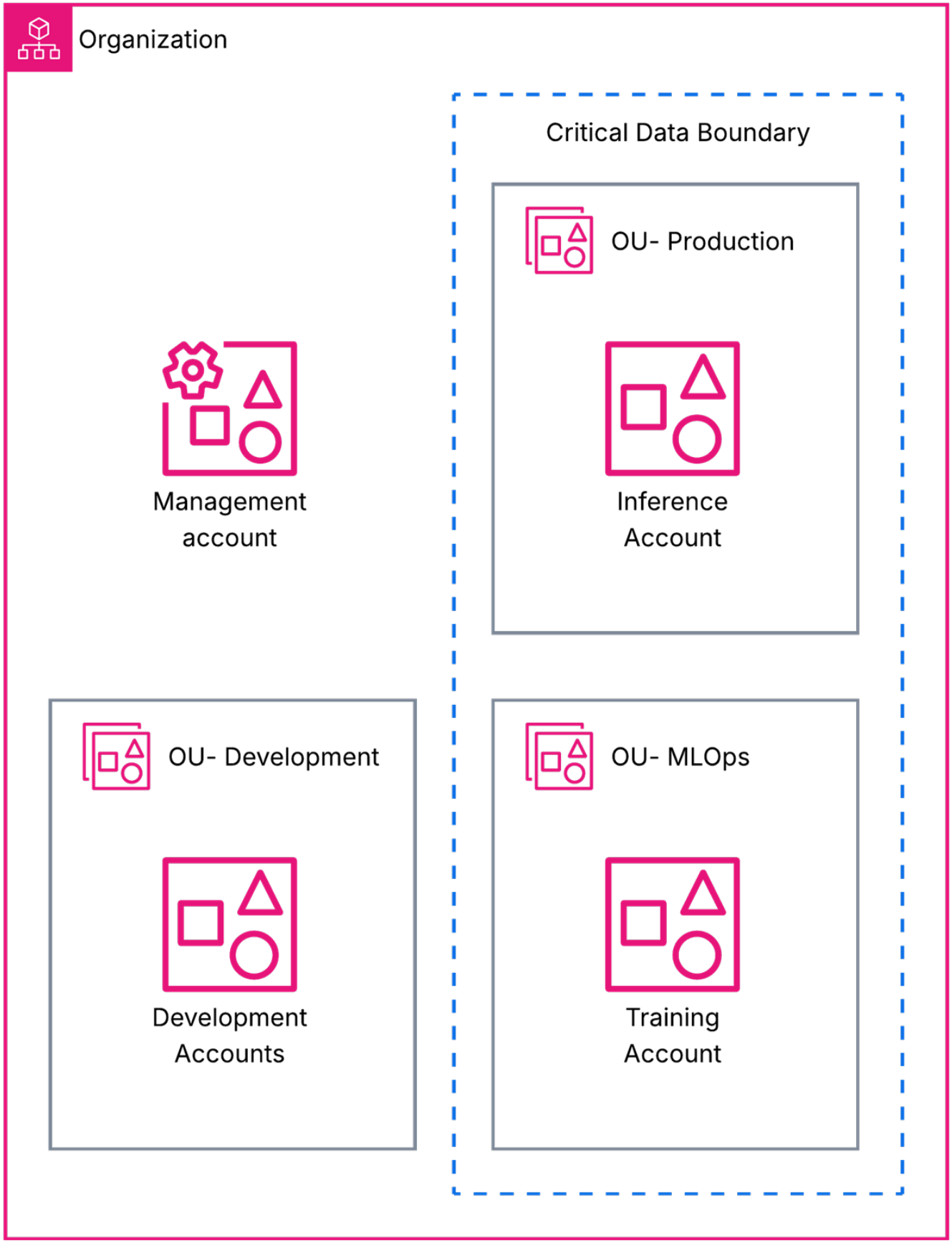
AI systems process and generate data in ways that challenge traditional data classification and protection mechanisms. Models can inadvertently memorize and reproduce sensitive information from training data, while their outputs may contain synthetic but realistic personal information. Risks include the following:

- Sensitive data leakage through model memorization and regurgitation from custom foundation models
- Compliance violations when personal data is processed without proper controls such as overly permissive agents
- Data poisoning in fine-tuning scenarios where malicious training data affects model behavior
- Cross-tenant data exposure in multi-tenant AI applications

Multi-account architecture for AI workloads

Organizations implementing AI at scale should adopt a multi-account strategy that provides clear separation of concerns, enhanced security boundaries, and simplified governance across different AI lifecycle phases. As shown in the following diagram, this architectural approach isolates inference workloads from training activities while maintaining centralized security oversight and cross-account collaboration capabilities:

- **AI development account** – Sandbox for experimentation and prototyping with non-sensitive data
- **AI inference account** – Production environment for AI model consumption and application hosting
- **AI training account** – Secured environment for handling sensitive training data and production model development


Organization
Critical Data Boundary
OU- Production
Management account
Inference Account
OU- Development
OU- MLOps
Development Accounts
Training Account

AI development account

The development account provides a sandbox environment for AI experimentation, prototyping, and initial model development using non-sensitive data. This account enables data scientists and developers to explore AI capabilities, test new approaches, and develop proof-of-concept solutions without access to production or sensitive training datasets.

Deploy Amazon Macie automated data discovery to help security and data science teams identify and classify data in development environments. Configure Macie to scan Amazon Simple Storage Service (Amazon S3) buckets regularly and alert when sensitive data appears in the development account. This approach enables teams to remediate data classification issues before they reach production.

Structure this account with permissive development policies that encourage experimentation while maintaining clear boundaries that prevent access to sensitive data or production systems. Implement cost controls and resource limits to manage experimental workloads and use AWS Budgets to monitor spending on development activities.

Deploy Amazon SageMaker Studio for collaborative development environments, with shared notebooks and experiment tracking capabilities. Configure automated cleanup policies that remove unused resources and temporary datasets, maintaining a clean development environment while controlling costs.

AI inference account

The inference accounts serve as production environments for AI model consumption and application hosting. Organizations typically deploy multiple inference accounts to maintain workload isolation, for example, separate accounts for different business units, applications, or security boundaries. Each inference account contains Amazon Bedrock endpoints, agent orchestration services, and user-facing applications that consume foundation models or custom models deployed from the training account. Security controls in these accounts focus on runtime protection, user access management, and real-time monitoring of AI interactions.

Configure each inference account with restrictive IAM policies that prevent model training activities, while enabling comprehensive inference capabilities. Implement Amazon Cognito or AWS IAM Identity Center for user authentication, and with fine-grained permissions that control access to specific models. Deploy Amazon Bedrock Guardrails and AWS WAF to filter inputs and outputs, ensuring that AI interactions meet organizational security standards.

Establish cross-account trust relationships that allow inference accounts to access approved model artifacts from the training account through secure, audited mechanisms. Use AWS PrivateLink endpoints to maintain private connectivity to AI services while implementing comprehensive logging through AWS CloudTrail and Amazon CloudWatch to monitor all inference activities.

Use Amazon GuardDuty Malware Protection for S3 to scan untrusted files that users submit for processing, such as document uploads, images, or data files that AI workloads analyze. This protection is particularly important for applications that process user-submitted content like mortgage documents, resumes, or customer support attachments.

AI training account

The training account serves as a highly secured staging environment specifically designed for handling sensitive training data and production model development. This account implements the strictest security controls because of the potential presence of personally identifiable information (PII), proprietary datasets, and other sensitive information used in model training processes. Models developed in the development account are promoted to the training account for production-grade training with real datasets before deployment to inference accounts.

Establish secure model promotion workflows that move models from development through training to inference environments with appropriate security validations at each stage. Implement automated security scanning of model artifacts and comprehensive approval processes before any model deployment to production inference systems.

Implement enhanced data protection measures including mandatory encryption at rest and in transit. Use AWS Key Management Service (AWS KMS) customer managed keys that provide granular access control over sensitive training datasets. Deploy Amazon Macie with continuous monitoring to identify and classify sensitive data, to help make sure that all training materials are properly protected and access is appropriately restricted. If possible, redact sensitive data before using it for training to minimize exposure risk.

Configure Amazon SageMaker with private VPC deployments that eliminate internet access for training jobs, using VPC endpoints for necessary AWS service communication. Implement strict IAM policies that limit access to authorized personnel only, with multi-factor authentication requirements and session-based access controls for all training activities.

Establish secure data ingestion pipelines that validate and sanitize incoming training data while maintaining comprehensive audit trails of all data access and processing activities. Use Amazon S3 with Object Lock and versioning to help ensure training data integrity and provide immutable audit records of all training dataset modifications.

Implement temporary elevated access management for access to training data when feasible, granting time-limited permissions that automatically expire after use. Log all user activity through CloudTrail and configure CloudWatch alarms to detect anomalous access patterns to sensitive training datasets.

Cross-account security and governance

Implement centralized security monitoring through AWS Security Hub and Amazon GuardDuty deployed across all three account types, with findings aggregated in a dedicated security account. Use AWS Config to enforce consistent security baselines while allowing account-specific security enhancements, particularly for the training account's heightened security requirements.

Configure cross-account logging aggregation that forwards all AI-related logs to a centralized log archive account, with enhanced retention and protection for training account logs due to their potential sensitivity. Use Amazon EventBridge rules to orchestrate security responses across all accounts while maintaining appropriate isolation between environments.

Defense in depth

As shown in the following diagram, a defense-in-depth strategy implements security controls at different layers within each account to protect AI workloads. This section details security controls in the Application, Data, and Network layers.

Application security layer

Deploy AWS WAF as the first line of defense against malicious requests targeting your AI applications. Configure rate limiting to prevent resource exhaustion attacks and implement AWS Managed Rules for the Core rule set and Known bad inputs managed rule groups. Create custom AWS WAF rules to detect common prompt injection patterns such as instruction override attempts, delimiter manipulation, and context escape sequences. For applications handling critical business functions or experiencing high request volumes, enhance this protection with AWS Shield Advanced to guard against DDoS attacks.

Implement comprehensive input validation through Amazon API Gateway request validators. Configure validators to enforce JSON schema requirements and establish appropriate character limits for prompts and metadata fields. This validation prevents malformed requests from reaching your AI models and helps mitigate prompt injection attacks.

Strengthen authentication and authorization by deploying AWS Lambda authorizers that validate user context and session state. Alternatively, implement Amazon Verified Permissions for policy-

based authorization that evaluates fine-grained permissions dynamically based on user attributes, resource context, and request parameters before model invocation. This approach enables centralized policy management and consistent authorization decisions across your AI applications.

Configure response transformation to strip sensitive metadata from model outputs, helping to ensure that internal system information never reaches end users. This approach includes removing debug information, internal identifiers, and system prompts that could reveal application architecture or security controls.

Monitor the effectiveness of these controls through CloudWatch custom metrics that track prompt characteristics, response times, and error rates. Create CloudWatch alarms to identify anomalous patterns that potentially indicate attacks or system degradation, enabling rapid response to emerging threats.

Data security

Deploy Amazon Macie automated data discovery to identify and classify sensitive data in your AI inference workloads. Configure Macie to scan Amazon S3 buckets that contain the following:

- User prompts and conversation logs
- Model responses and generated content
- RAG knowledge base documents
- Agent memory and session data
- Application configuration and prompt templates

Enhance detection capabilities with custom data identifiers that recognize your organization's specific sensitive data patterns. Review Macie findings regularly and establish automated remediation workflows using EventBridge to alert security teams when sensitive data appears in unexpected locations.

Implement encryption using AWS KMS with customer managed keys for all inference-related data at rest. Organize your encryption strategy by using separate keys for the following:

- Conversation history and session data
- RAG knowledge base documents
- Agent memory and context storage
- Application logs and audit trails

- Cached model responses, if applicable

Establish key rotation policies that balance security requirements with operational efficiency. Implement cross-region key replication to support disaster recovery scenarios without compromising data protection.

Extend your data protection to real-time processing by deploying Amazon Comprehend PII detection or Amazon Bedrock Guardrails on both model inputs and outputs. Configure automatic redaction capabilities that operate in real time for interactive applications or in batch mode for stored conversations.

Amazon Comprehend detects common PII types including names, addresses, credit card numbers, and Social Security numbers. Amazon Bedrock Guardrails provides additional capabilities including custom regex patterns for organization-specific sensitive data and contextual filtering based on conversation flow.

Monitor PII detection rates through CloudWatch metrics to identify potential data handling issues and help ensure compliance with privacy regulations. Create CloudWatch alarms when PII detection rates exceed expected baselines, which may indicate users attempting to share sensitive information or applications inadvertently processing restricted data.

Configure Amazon S3 bucket policies that enforce encryption requirements, restrict access appropriately, and require multi-factor authentication for critical operations such as bucket deletion or policy modification. Implement Amazon S3 access points with VPC endpoints to provide role-based access control for different workload types. For example, create separate access points for application workloads accessing RAG knowledge bases, security teams reviewing conversation logs, and compliance auditors accessing audit trails.

Enable S3 Versioning for conversation logs and knowledge base documents to support audit requirements and incident investigation. Enable Amazon S3 data event logging through CloudTrail to maintain comprehensive access records, capturing who accessed what data, when, and from which source. For applications with data retention requirements, configure Amazon S3 Lifecycle policies to archive or delete conversation logs automatically after appropriate retention periods. This approach balances compliance needs with data minimization principles.

Network security enhancement

Design your network security architecture around the principle of defense in depth. Begin with restrictive virtual private cloud (VPC) security groups that allow only necessary traffic

between application tiers. Structure these security groups to create clear boundaries between web, application, and data tiers, with controlled inter-tier communication flowing only through designated ports and protocols. This segmented approach limits the potential impact of any security breach while maintaining operational functionality.

Architect your network topology using dedicated subnets for AI workloads. Design routing carefully so that traffic is directed through NAT gateways for secure outbound internet access and VPC endpoints for efficient AWS service communication. Implement network ACLs as an additional defensive layer, using explicit allow rules for required traffic while maintaining a default-deny posture for all other communications.

Enhance your network defenses by deploying AWS Network Firewall. Use its intrusion detection and prevention capabilities for east-west traffic between application tiers, north-south traffic for ingress and egress, and lateral movement detection within your VPC. Configure rules that identify unusual request characteristics, detect high-frequency automated attacks, and recognize other indicators of malicious activity targeting AI systems. This deep packet inspection capability provides visibility into threats that might bypass application-layer controls.

Deploy Resolver DNS Firewall, a feature of Amazon Route 53 Resolver, to block malicious domain queries and enforce DNS-level security policies for your AI infrastructure. Configure DNS Firewall to block known malicious domains, prevent data exfiltration through DNS tunneling, and alert on suspicious DNS patterns that may indicate compromised systems or command-and-control communications.

Maintain comprehensive network visibility through VPC Flow Logs configured with custom formats that capture relevant metadata for security analysis. Enable VPC Flow Logs for all subnets hosting AI workloads. Configure VPC Flow Logs to capture accepted traffic, rejected traffic, and all traffic to provide complete visibility into network communications.

Integrate VPC Flow Logs with your security information and event management (SIEM) solution for automated pattern analysis and threat detection. You can use Amazon OpenSearch Service for log aggregation and analysis, or integrate with third-party SIEM platforms that support AWS log ingestion. Configure your SIEM to detect anomalous patterns including unusual traffic volumes, connections to unexpected destinations, or communication patterns that deviate from established baselines.

Connect your threat detection system to EventBridge for orchestrated incident response. Configure EventBridge rules to trigger automated responses when security events are detected such as the following:

- Invoke AWS Lambda functions to isolate compromised resources.
- Execute AWS Systems Manager Automation runbooks to remediate common security issues.
- Send notifications to security teams through Amazon Simple Notification Service (Amazon SNS) for manual investigation.

This approach creates a closed-loop security monitoring and response system that reduces time to detection and response.

Model evaluation and validation

Model evaluation represents a critical security checkpoint in AI implementations, requiring comprehensive assessment of model behavior, output quality, and adherence to organizational policies before deployment. Evaluate foundation models (FMs) in the context of your specific use cases to ensure they meet security and quality requirements.

Before deploying an FM to production, establish evaluation frameworks that test model behavior against your security requirements. Use Amazon Bedrock model evaluation to compare different FMs and select the one that best meets your needs. Create standardized evaluation datasets that include adversarial examples to test model robustness against prompt injection, jailbreak attempts, and other manipulation techniques.

Test models against your organization's responsible AI policies by evaluating outputs for bias, toxicity, and alignment with ethical guidelines. Use Amazon SageMaker Clarify to analyze model outputs for potential bias across different demographic groups or use cases. Document evaluation results and obtain appropriate approvals before deploying models to production environments.

Implement continuous monitoring through CloudWatch to identify performance degradation or unusual output patterns in production environments. Configure CloudWatch metrics to track model invocation rates, response latencies, error rates, and token usage patterns. Create CloudWatch alarms that trigger when metrics deviate from established baselines, which may indicate security issues, service degradation, or unexpected usage patterns.

Monitor Amazon Bedrock Guardrails metrics to track how frequently content is filtered or blocked, providing visibility into potential security threats or policy violations. Analyze trends in guardrail activations to identify emerging attack patterns or areas where additional security controls may be needed.

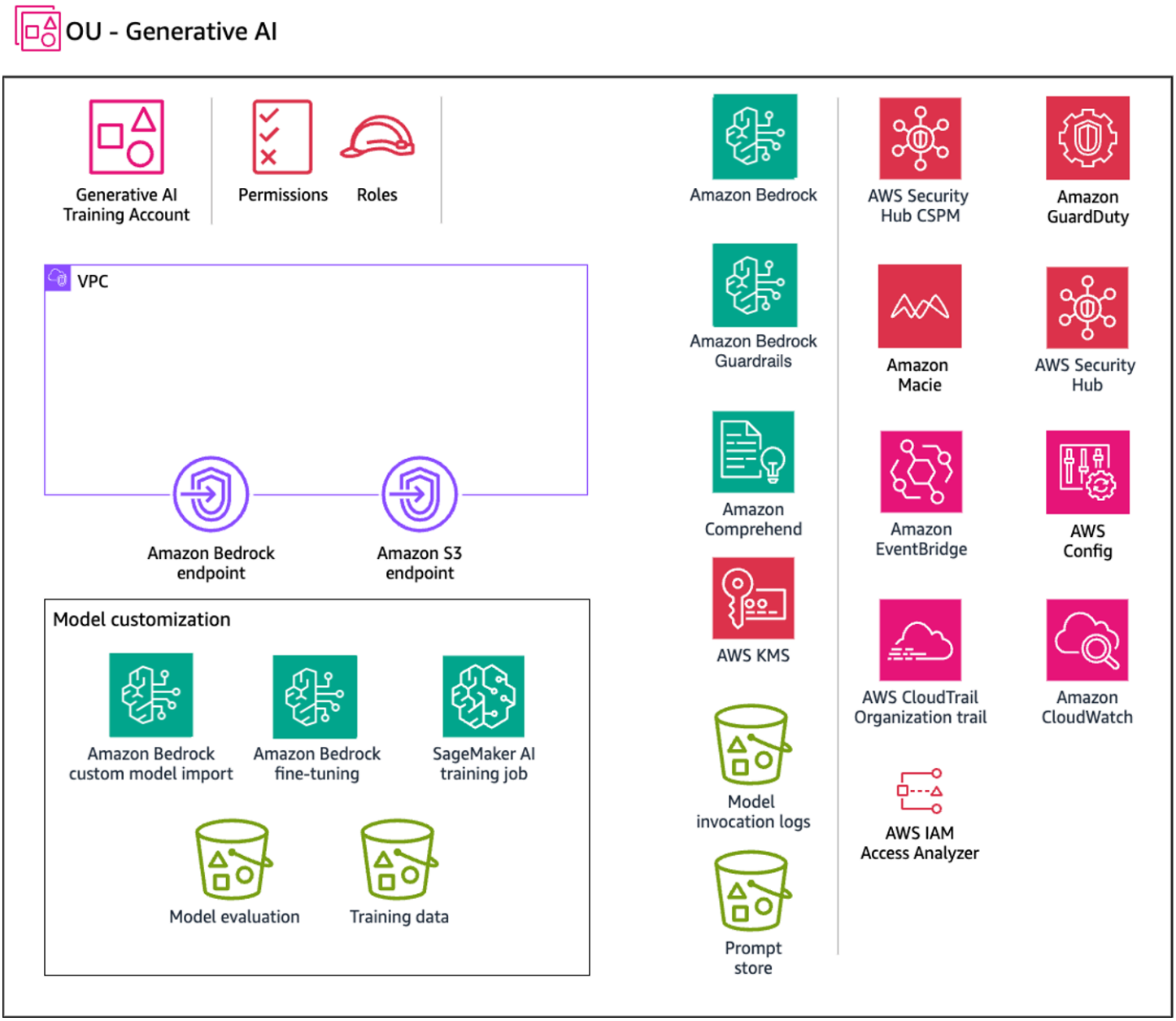
Use AWS Step Functions to establish automated pipelines that orchestrate regular security assessments, performance benchmarks, and compliance validation. Configure these pipelines to

run on a schedule or trigger based on specific events such as significant changes in usage patterns or the availability of new model versions.

Capability 2. Providing secure access, usage, and implementation for generative AI model customization

Influence the future of the AWS Security Reference Architecture (AWS SRA) by taking a short survey.

The scope of this scenario is to secure model customization. This use case focuses on securing the resources and training environment for a model customization job as well as securing the invocation of a custom model. The following diagram illustrates the AWS services recommended for the Generative AI account for this capability.



The Generative AI account includes services required for customizing a model along with a suite of required security services to implement security guardrails and centralized security governance. To allow for private model customization, you should create Amazon S3 gateway endpoints for the training data and evaluation Amazon S3 buckets that a private VPC environment is configured to access.

Rationale

Model customization improves foundation model (FM) performance for specific use cases by providing training data. Amazon Bedrock offers two customization methods:

- Continued pre-training with unlabeled data to enhance domain knowledge
- Fine-tuning with labeled data to optimize task-specific performance

Customized models require Provisioned Throughput for inference.

This capability addresses the following scenarios from the Generative AI Security Scoping Matrix:

- **Scope 4 - Model customization** – You customize an FM (from Amazon Bedrock or Amazon SageMaker Jumpstart) with your data to improve performance for specific tasks or domains. You control the application, customer data, training data, and customized model. The FM provider controls the pre-trained model and its training data.
- **Scope 5 - Model training from scratch** – You train a model from scratch using datasets you provide. You control the training data, model algorithm, training infrastructure, application, customer data, and related infrastructure.

Beyond customizing models within Amazon Bedrock, you can use the Custom Model Import feature to import models customized in other environments, such as Amazon SageMaker AI. Use Safetensors for the imported model serialization format. Unlike `pickle`, Safetensors stores only tensor data, not arbitrary Python objects. This approach eliminates vulnerabilities from unpickling untrusted data because Safetensors can't execute code.

To detect potential training data leakage, introduce canaries into your training data. Canaries are unique, identifiable strings that should never appear in model outputs. Configure prompt logging to alert when these canaries are detected, indicating the model may be memorizing and reproducing training data inappropriately.

Amazon Bedrock model customization

You can privately and securely customize FMs with your own data in Amazon Bedrock to build applications specific to your domain, organization, and use case. Fine-tuning increases model accuracy by providing your own task-specific, labeled training dataset to further specialize FMs. Continued pre-training trains models using your own unlabeled data in a secure and managed environment with customer managed keys. For more information, see Customize your model to improve its performance for your use case in the Amazon Bedrock documentation.

Model training or fine-tuning with SageMaker AI

You can train new models or fine-tune existing models by using Amazon SageMaker AI training jobs. This solution creates models customized for your business needs while maintaining control of all resources, including Amazon Elastic Compute Cloud (Amazon EC2) instances, training code, and training infrastructure.

Security considerations

Model customization creates artifacts, including the model and its weights, that are used in production workloads. This stage faces the following threats:

- **Data and model poisoning** – A threat actor injects malicious data to alter model behavior, introducing bias and causing unintended outputs.
- **Sensitive information disclosure** – A model trained on datasets containing personally identifiable information (PII) leaks sensitive information during inference.

SageMaker AI and Amazon Bedrock provide features that mitigate these risks, including data protection, access control, network security, logging, and monitoring.

Remediations

This section reviews the AWS services and features that address the risks that are specific to this capability.

Data protection

Encrypt the model customization job, output files (training and validation metrics), and resulting custom model. For this encryption, use an AWS Key Management Service (AWS KMS) customer managed key that you create, own, and manage.

When you use Amazon Bedrock to run a model customization job, you store the input files (training and validation data) in your Amazon S3 bucket. When the job is completed, Amazon Bedrock stores the output metrics files in the S3 bucket that you specified when you created the job. Amazon Bedrock stores the resulting custom model artifacts in an S3 bucket controlled by AWS. By default, input and output files are encrypted with Amazon S3 SSE-S3 server-side encryption using an AWS managed key. You can choose to encrypt these files with a customer managed key.

Identity and access management

Create a custom AWS Identity and Access Management (IAM) service role for model customization or model import that follows the principle of least privilege.

To create a service role for model customization, follow the instructions in the Amazon Bedrock documentation.

To create a service role for importing pre-trained models, follow the instructions in the Amazon Bedrock documentation.

Network security

Use a VPC with Amazon Virtual Private Cloud (Amazon VPC) to control access to your data. When you create your VPC, use the default DNS settings for your endpoint route table so that standard Amazon S3 URLs resolve.

If you configure your VPC with no internet access, create an Amazon S3 VPC endpoint. Use this VPC endpoint to allow your model customization jobs to access the S3 buckets that store your training and validation data and model artifacts.

For SageMaker AI, configure the training job with a VPC configuration, including private subnets and security groups that restrict both inbound and outbound traffic. This approach helps to ensure that Amazon EC2 instances can only access the resources that you define. Combined with Amazon S3 VPC endpoints, this approach helps to ensure that EC2 instances only access specified S3 buckets.

After you set up your VPC and endpoint, attach permissions to your model customization IAM role. After you configure the VPC and required roles and permissions, you can create a model customization job that uses this VPC. By creating a VPC with no internet access and an associated Amazon S3 VPC endpoint for training data, you can run your model customization job with private connectivity without internet exposure.

Recommended AWS services

This section discusses the AWS services that are recommended to build this capability securely. In addition to the services in this section, use Amazon OpenSearch Service and Amazon Comprehend as discussed in Capability 3.

Amazon S3

When you run a model customization job, the job accesses your Amazon S3 bucket to download input data and upload job metrics. You can choose fine-tuning or continued pre-training as the model type when you submit your model customization job on the Amazon Bedrock console or API. After a model customization job completes, analyze the training process results. To do this, you can view the files in the output S3 bucket that you specified when you submitted the job or view details about the model.

Encrypt both buckets with a customer managed key. Use Amazon S3 Object Lock or versioning to ensure data integrity. For additional network security hardening, create a gateway endpoint for the S3 buckets that the VPC environment accesses. Log and monitor all access. Use resource-based policies to control access to your Amazon S3 files.

Amazon Macie

Amazon Macie is a fully managed data security and data privacy service that uses machine learning and pattern matching to discover and help protect your sensitive data in AWS. You need to identify the type and classification of data that your workload is processing to ensure that appropriate controls are enforced. Macie can help identify sensitive data in your prompt store and model invocation logs stored in S3 buckets.

You can use Macie to automate discovery, logging, and reporting of sensitive data in Amazon S3. You can do this in two ways: Configure Macie to perform automated sensitive data discovery, or create and run sensitive data discovery jobs. For more information, see Discovering sensitive data with Amazon Macie in the Macie documentation.

Amazon EventBridge

Use EventBridge to configure SageMaker to respond automatically to model customization job status changes in Amazon Bedrock. Events from Amazon Bedrock are delivered to EventBridge in near real time. You can write simple rules to automate actions when an event matches a rule.

AWS KMS

Use a customer managed key to encrypt the model customization job, output files (training and validation metrics), resulting custom model, and Amazon S3 buckets that host the training, validation, and output data. For more information, see Encryption of custom models in the Amazon Bedrock documentation.

A key policy is a resource policy for an AWS KMS key. Key policies are the primary way to control access to KMS keys. You can also use IAM policies and grants to control access to KMS keys, but every KMS key must have a key policy. Use a key policy to provide permissions to a role to access the custom model encrypted with the customer managed key. This approach allows specified roles to use a custom model for inference.

Amazon CloudWatch

Use CloudWatch to monitor training job metrics in SageMaker and fine-tuning metrics in Amazon Bedrock. Create alarms to receive notifications when a job fails or when a metric deviates from baseline.

AWS CloudTrail

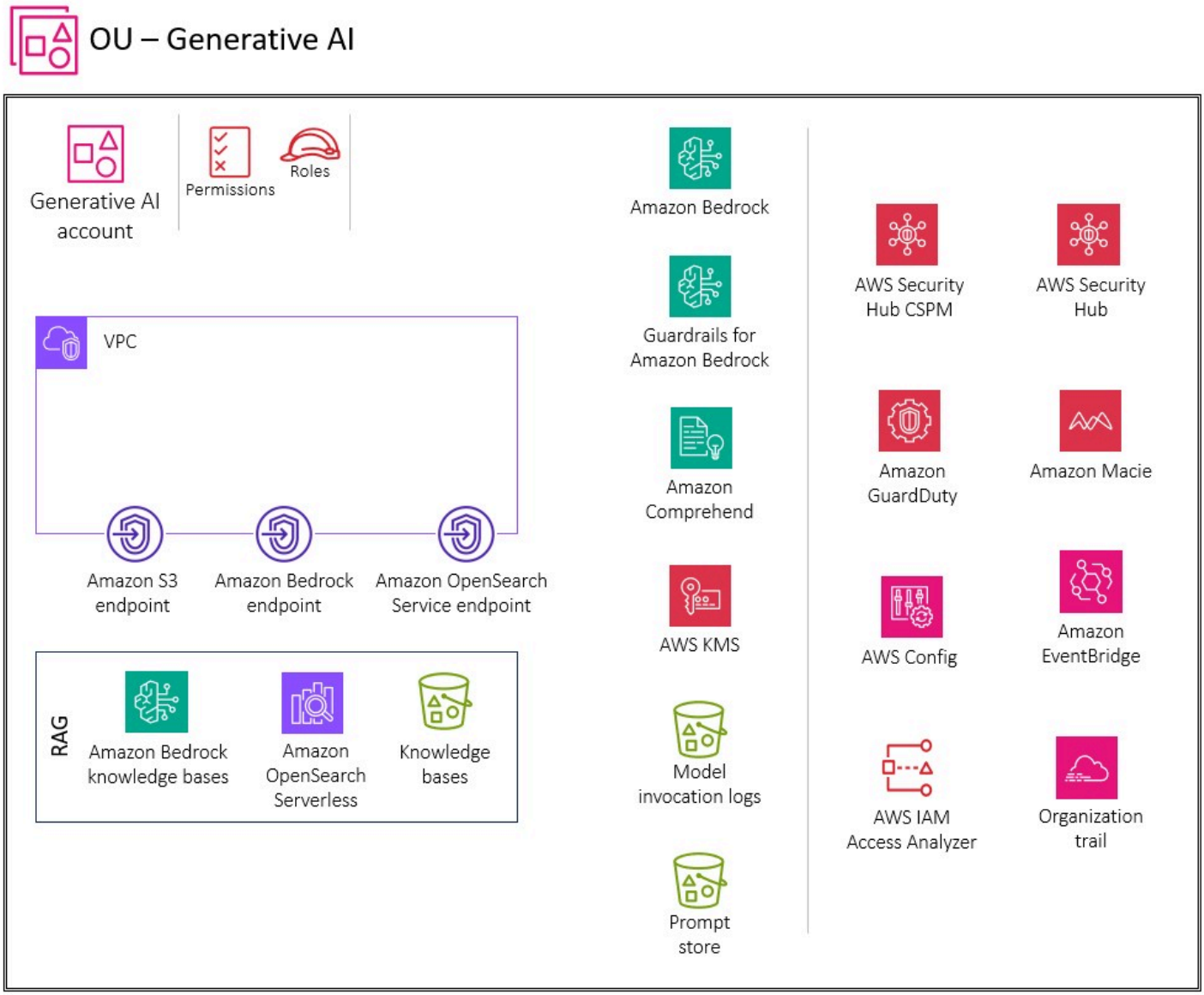
Use CloudTrail to log all events on your AWS resources. Create a trail filtered on your training resources, including datasets on Amazon S3. This trail enables you to act on suspicious activity surrounding your resources.

Capability 3. Providing secure access to data and systems for generative AI

> Influence the future of the AWS Security Reference Architecture (AWS SRA) by taking a short survey.

Retrieval Augmented Generation (RAG) is a foundational pattern that enhances large language model (LLM) responses by retrieving information from external knowledge bases before generating answers. This approach addresses a core limitation of foundation models (FMs): They are trained on data with a fixed knowledge cutoff and lack access to current enterprise data such as customer records, product catalogs, internal documentation, and business systems.

RAG enables the LLM to provide up-to-date, context-specific responses by dynamically pulling relevant information from enterprise data sources. However, this integration introduces critical security challenges. Securing RAG implementations requires extending defense-in-depth principles from Capability 1 and Capability 2 to address how LLMs securely use data from external sources. The following diagram illustrates recommended AWS services for the Generative AI account RAG capability.



The Generative AI account includes services for storing embeddings in a vector database, storing conversations for users, and maintaining a prompt store. The account includes security services to implement security guardrails and centralized security governance. Create Amazon Simple Storage Service (Amazon S3) gateway endpoints for the model invocation logs, prompt store, and knowledge base data source buckets in Amazon S3 that the VPC environment accesses. Create an Amazon CloudWatch Logs gateway endpoint for the CloudWatch logs that the VPC environment accesses.

Rationale

RAG enhances FM responses by retrieving information from external, authoritative knowledge bases before generating answers. This approach overcomes FM limitations by providing access to up-to-date, context-specific data, improving the accuracy and relevance of generated responses.

RAG can be implemented across Scopes 2-5 of the Generative AI Security Scoping Matrix. Scope 2 applications represent scenarios where organizations use third-party AI services (like Salesforce Einstein or ChatGPT) where the service provider controls both the FM and the application layer. You control only the prompts and customer data you provide to the service. You can enhance responses from third-party enterprise applications by implementing RAG to extract information from internal data, which augments queries processed by the third-party application. In Scope 2, you implement RAG either by connecting to your organization's data sources or by uploading and referencing custom documents.

In Scope 3, you build a generative AI application using a pre-trained FM such as those offered on Amazon Bedrock. You control your application and any customer data your application uses. The FM provider controls the pre-trained model and its training data.

RAG systems face the following unique security risks:

- Data exfiltration of RAG data sources by threat actors
- Poisoning of RAG data sources with prompt injections or malware
- Unauthorized access to sensitive information through inadequate access controls
- Sensitive information disclosure through uncontrolled model outputs
- Lack of data provenance leading to compliance and auditability challenges

Design considerations

Avoid customizing an FM with sensitive data (for more information, see Capability 2). Instead, use the RAG technique to interact with sensitive information. RAG provides the following advantages:

- **Tighter control and visibility** – Keep sensitive data separate from the model. You can edit, update, or remove data without retraining the model, ensuring data governance and compliance with regulatory requirements.
- **Reduced sensitive information disclosure** – RAG controls interactions with sensitive data during model invocation. This reduces the risk of unintended disclosure that occurs when you incorporate data directly into the model's parameters.

- **Flexibility and adaptability** – Update or modify sensitive information as data requirements or regulations change without retraining or rebuilding the language model.
- **Enhanced security posture** – Implement multiple security layers including metadata filtering, access controls, and data redaction at different stages of the RAG pipeline.

Multi-layered security strategy

Implement a defense-in-depth approach with security controls at the following stages:

- **Ingestion time** – Filter and validate data before it enters the knowledge base.
- **Storage level** – Encrypt data at rest and implement access controls.
- **Retrieval time** – Apply metadata filtering and role-based access controls.
- **Inference time** – Use guardrails to filter model outputs and detect sensitive information.

Amazon Bedrock Knowledge Bases

Amazon Bedrock Knowledge Bases provides a fully managed solution for building RAG applications by securely connecting FMs to your organization's data. This service uses vector stores (such as Amazon OpenSearch Serverless) to retrieve relevant information efficiently. The FM uses this information to generate responses. Amazon Bedrock synchronizes your data from Amazon S3 to the knowledge base and generates embeddings for efficient retrieval.

Key features of Amazon Bedrock Knowledge Bases include the following:

- **Source attribution** – Knowledge bases include source attribution for all retrieved information to improve transparency and minimize hallucinations. This provenance tracking enables you to:
  - Verify the accuracy of generated responses.
  - Maintain audit trails for compliance.
  - Build user trust in AI-generated content.
  - Support troubleshooting and investigations during security events.
- **Automated vector store management** – Amazon Bedrock automatically creates and manages vector stores in OpenSearch Serverless, synchronizing data from Amazon S3 and generating embeddings for efficient retrieval.
- **Metadata filtering** – Knowledge bases support metadata filtering capabilities that enable access control by pre-filtering the vector store based on document metadata before searching for

relevant documents. This filtering reduces noise, improves retrieval accuracy, and enforces data access policies.

- **Multimodal support** – Knowledge bases process documents with visual resources, extracting and retrieving images in responses to queries, which supports comprehensive document understanding.

For each vector database option, configure the following:

- Field mappings for vector embeddings, text chunks, and metadata
- Customer managed AWS KMS keys for encrypting secrets and data
- AWS Secrets Manager secrets for authentication credentials
- Network connectivity through AWS PrivateLink where supported

Security considerations

Generative AI RAG workloads face unique risks, including data exfiltration of RAG data sources. Another risk is indirect prompt injection attacks where threat actors insert malicious documents into the knowledge base to manipulate model outputs.

Amazon Bedrock knowledge bases provide security controls for data protection, access control, network security, logging and monitoring, and metadata filtering for secure retrieval. These controls address data exfiltration and unauthorized access risks. To mitigate indirect prompt injection attacks, implement input validation and content filtering on documents before ingestion.

Remediations

This section reviews the AWS services and features that address the risks that are specific to this capability.

Data protection

Encrypt your knowledge base data in transit and at rest using an AWS Key Management Service (AWS KMS) customer managed key. When you configure a data ingestion job for your knowledge base, encrypt the job with a customer managed key. If you let Amazon Bedrock create a vector store in Amazon OpenSearch Service for your knowledge base, Amazon Bedrock passes an AWS KMS key of your choice to OpenSearch Service for encryption.

You can encrypt sessions in which you generate responses from querying a knowledge base with an AWS KMS key. You store the data sources for your knowledge base in your Amazon S3 bucket. If you encrypt your data sources in Amazon S3 with a customer managed key, attach the required policies to your knowledge base service role.

If you configure vector stores with AWS Secrets Manager secrets, encrypt the secrets with customer managed keys and attach decryption permissions to the knowledge base service role. Ensure all data in transit uses TLS 1.2 or higher with secure cipher suites.

For more information and the policies to use, see Encryption of knowledge base resources in the Amazon Bedrock documentation.

Data classification and handling

Implement data classification schemes to categorize data based on sensitivity and criticality. Establish clear classification tiers (for example, Public, Internal, Confidential, and Restricted) with specific handling requirements for each level.

Classify data at the point of ingestion. Use automated tools like Amazon Macie to detect and classify sensitive data in Amazon S3 buckets that contain knowledge base data sources.

Use AWS resource tags to categorize sensitive data and monitor compliance with protection requirements. AWS Organizations tag policies enforce tagging standards across accounts.

Maintain a data catalog that maps data in your organization, its location, sensitivity level, and the controls in place to protect it. AWS Glue Data Catalog supports metadata storage and management.

Data lineage and provenance tracking

Implement comprehensive data provenance tracking to record the history of data as it progresses through your RAG workload.

Data lineage provides the following benefits:

- **Regulatory compliance** – Demonstrates data handling practices for audits and certifications
- **Troubleshooting** – Enables root cause analysis when data quality issues arise
- **Security investigations** – Provides audit trails during security incidents
- **Data quality** – Ensures confidence in data origin, transformations, and ownership

- **Impact analysis** – Identifies downstream effects of data changes

Implementation approaches for data provenance tracking include the following:

- **AWS Glue Data Catalog** – Store metadata and track lineage across data processing pipelines.
- **Amazon SageMaker ML Lineage Tracking** – Track model training data, hyperparameters, and deployment artifacts.
- **AWS CloudTrail** – Capture API activities across AI services for audit trails.
- **Amazon CloudWatch** – Monitor data quality, usage, and model drift with generative AI-driven debugging and root cause analysis.
- **Third-party integration** – Support open telemetry with integration to third-party observability tools.

Identity and access management

Create a custom service role for knowledge bases for Amazon Bedrock following the principle of least privilege. Create a trust relationship that allows Amazon Bedrock to assume this role, and create and manage knowledge bases.

Attach identity policies to the custom knowledge base service role that grant permissions to access Amazon Bedrock models, data sources in Amazon S3, vector databases, and encryption keys. For the complete list of required permissions, see Create a service role for Amazon Bedrock Knowledge Bases in the Amazon Bedrock documentation.

Knowledge bases support security configurations to set up data access policies for your knowledge base and network access policies for your private Amazon OpenSearch Serverless knowledge base. For more information, see Create a knowledge base by connecting to a data source in Amazon Bedrock Knowledge Bases in the Amazon Bedrock documentation.

Metadata filtering for secure retrieval

Amazon Bedrock Knowledge Bases supports metadata filtering to refine and secure contextual retrieval from vector stores. For every document added, you can supply metadata files (up to 10KB each) with attributes such as tags, dates, project IDs, and business units.

Metadata filtering enables fine-grained access control for RAG systems. By attaching metadata as key-value pairs to each vector during ingestion, you can do the following:

- **Filter queries** – Filter queries based on user attributes such as department, role, or clearance level. For example, metadata can include `{"department": "finance", "classification": "confidential"}` to restrict access to financial data.
- **Enforce data classification policies** – Tag vectors with sensitivity levels (public, internal, confidential, and restricted) and filter based on user permissions.
- **Support multi-tenant architectures** – Use metadata to isolate data between different tenants or business units, ensuring data segregation in shared infrastructure.
- **Enable temporal access controls** – Include timestamp metadata to implement time-based access restrictions or data retention policies.

It's up to the application or agent to add the correct metadata to each API call with Amazon Bedrock to filter results based on required key-value pairs.

Input and output validation

Input validation protects Amazon Bedrock knowledge bases from malicious content. Use malware protection in Amazon S3 to scan files for malicious content before uploading them to a data source. For an example implementation, see Integrating Malware Scanning into Your Data Ingestion Pipeline with Antivirus for Amazon S3 (AWS Blog post).

Use Amazon Comprehend to detect and redact sensitive information in documents before indexing them in your RAG knowledge base. For an example implementation, see Protect sensitive data in RAG applications with Amazon Bedrock (AWS blog post). For more information, see Detecting PII entities in the Amazon Comprehend documentation.

Use Amazon Macie to detect and generate alerts on potential sensitive data in Amazon S3 data sources to enhance security and compliance.

Recommended AWS services

This section discusses the AWS services that are recommended to build this capability securely. In addition to the services in this section, use Amazon CloudWatch and AWS CloudTrail as explained in Capability 2.

Amazon OpenSearch Serverless

Amazon OpenSearch Serverless is an on-demand, auto-scaling configuration for Amazon OpenSearch Service. An OpenSearch Serverless collection is an OpenSearch cluster that scales

compute capacity based on your application's needs. Amazon Bedrock knowledge bases use OpenSearchServerless for embeddings and Amazon S3 for the data sources that sync with the OpenSearch Serverless vector index.

Implement authentication and authorization for your OpenSearch Serverless vector store following the principle of least privilege. With data access control in OpenSearch Serverless, you can allow users to access collections and indexes regardless of their access mechanisms or network sources. Access permissions are done at the generative AI application layer.

OpenSearch Serverless supports server-side encryption with AWS KMS to protect data at rest. Use a customer managed key to encrypt that data. To allow the creation of an AWS KMS key for transient data storage during data ingestion, attach a policy to your knowledge bases for the Amazon Bedrock service role.

Private access can apply to OpenSearch Serverless-managed VPC endpoints, supported AWS services such as Amazon Bedrock, or both. Use AWS PrivateLink to create a private connection between your VPC and OpenSearch Serverless endpoint services. Use network policy rules to specify Amazon Bedrock access.

Monitor OpenSearch Serverless using Amazon CloudWatch, which collects raw data and processes it into readable, near real-time metrics. OpenSearch Serverless integrates with AWS CloudTrail, which captures API calls for OpenSearch Serverless as events. OpenSearch Service integrates with Amazon EventBridge to notify you of events that affect your domains.

Amazon S3

Store your data sources for your knowledge base in an Amazon S3 bucket. If you encrypted your data sources in Amazon S3 using a custom AWS KMS key (recommended), attach a policy to your knowledge base service role.

Use malware protection in Amazon S3 to scan files for malicious content before uploading them to a data source. Host your model invocation logs and commonly used prompts as a prompt store in Amazon S3. Encrypt all buckets with a customer managed key.

For additional network security hardening, create a gateway endpoint for the S3 buckets that the VPC environment accesses. Log and monitor all access. Enable versioning if you have a business need to retain the history of Amazon S3 objects. Apply object-level immutability with Amazon S3 Object Lock. Use resource-based policies to control access to your Amazon S3 files.

Amazon Comprehend

Amazon Comprehend uses natural language processing (NLP) to extract insights from document content. You can use Amazon Comprehend to detect and redact PII entities in English or Spanish text documents.

Integrate Amazon Comprehend into your data ingestion pipeline to automatically detect and redact PII entities from documents before you index them in your RAG knowledge base. This approach helps to ensure compliance and protects user privacy. Depending on the document types, you can use Amazon Textract to extract and send text to Amazon Comprehend for analysis and redaction.

With Amazon S3, you can encrypt your input documents when creating a text analysis, topic modeling, or custom Amazon Comprehend job. Amazon Comprehend integrates with AWS KMS to encrypt the data in the storage volume for `Start*` and `Create*` jobs. Amazon Comprehend encrypts the output results of `Start*` jobs by using a customer managed key.

Use the `aws:SourceArn` and `aws:SourceAccount` global condition context keys in resource policies to limit the permissions that Amazon Comprehend gives another service to the resource. Use AWS PrivateLink to create a private connection between your virtual private cloud (VPC) and Amazon Comprehend endpoint services. Implement identity-based policies for Amazon Comprehend with the principle of least privilege.

Amazon Comprehend integrates with AWS CloudTrail, which captures API calls for Amazon Comprehend as events.

Amazon Macie

Macie identifies sensitive data in your knowledge bases that is stored as data sources, model invocation logs, and prompt stores in Amazon S3 buckets. For Macie security best practices, see the *Amazon Macie* section in Capability 2.

AWS KMS

Use AWS Key Management Service (AWS KMS) customer managed keys to encrypt the following:

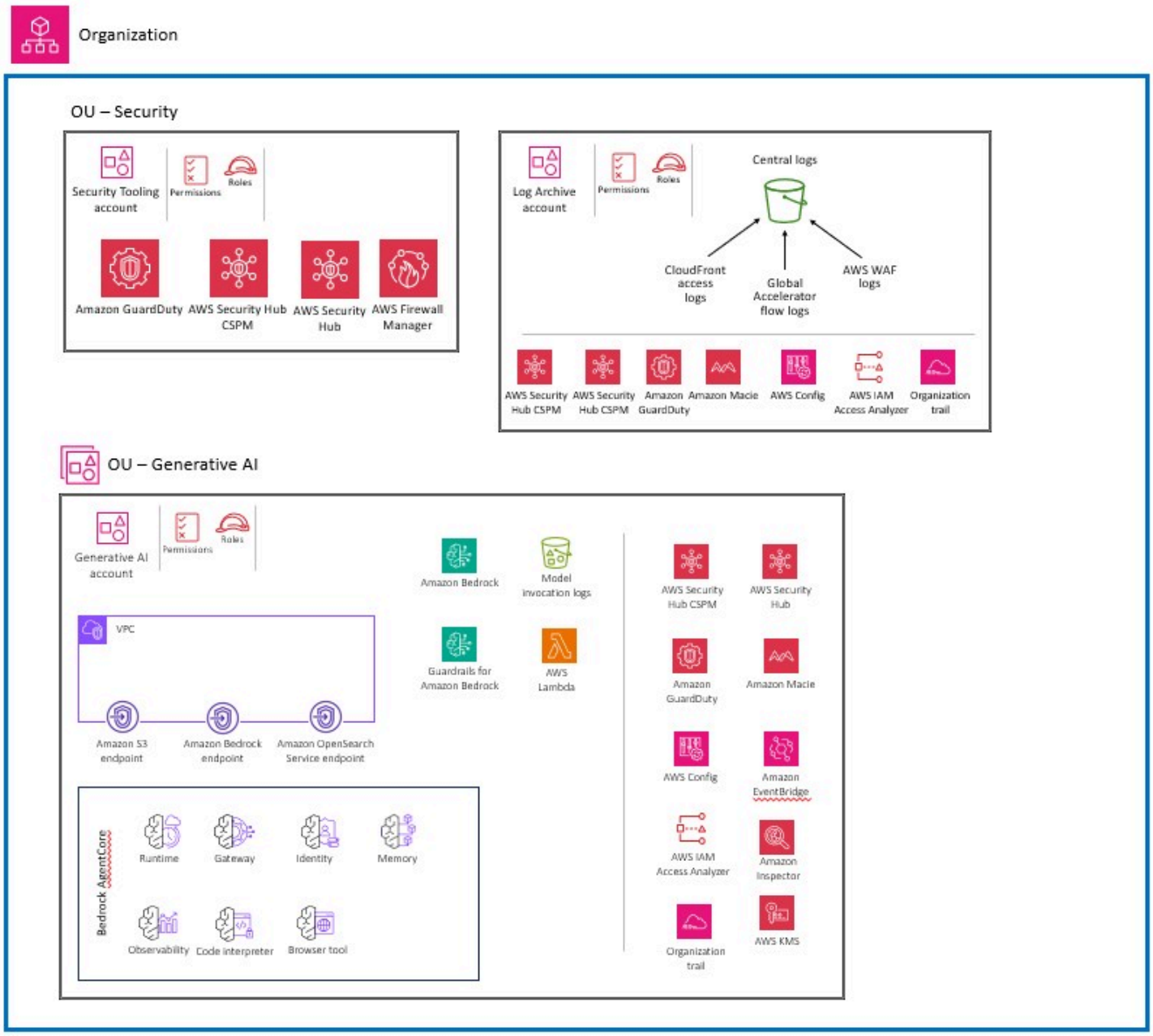
- Data ingestion jobs for your knowledge base
- Amazon OpenSearch Service vector database
- Sessions in which you generate responses from querying a knowledge base

- Model invocation logs in Amazon S3
- Amazon S3 bucket that hosts the data sources

Capability 4. Providing secure access, usage, and implementation of tools

Influence the future of the AWS Security Reference Architecture (AWS SRA) by taking a short survey.

The scope of this capability is to secure tool access and authentication for AI applications. The following diagram illustrates the AWS services recommended for the Generative AI account for this capability.



Rationale

Tool integration extends AI capabilities by connecting foundation models (FMs) to external functions and services. AI applications integrate tools through the following patterns:

- AWS Lambda functions for serverless business logic
- Model Context Protocol (MCP) servers for standardized tool interfaces
- External APIs for real-time data access

- Operating system tools for system-level operations
- Agent-to-agent (A2A) communication protocols for multi-agent workflows

This capability addresses Scope 3 of the Generative AI Security Scoping Matrix. In Scope 3, your organization builds a generative AI application using a pre-trained FM such as those offered in Amazon Bedrock while integrating external tools and services. You control your application, the tools it accesses, customer data, and permissions granted to the AI application. The FM provider controls the pre-trained model and its training data.

> **Note**
>
> Although this guidance focuses on application-level tool integration with Amazon Bedrock FMs (Scope 3), similar principles apply to fine-tuned and self-trained models (Scopes 4 and 5).

For user-facing AI applications that provide tool access to end users, see Capability 6.

Security considerations

AI applications with tool access face unique security risks that extend beyond traditional application vulnerabilities. When you grant AI applications the ability to invoke external functions and services, you create new attack surfaces. Adversaries can exploit these surfaces through both technical vulnerabilities and manipulation of the AI's reasoning process:

- Tool access introduces authentication and authorization challenges across multiple integration points. Unauthorized tool access can occur when authentication mechanisms fail to properly validate AI application identities, or when authentication credentials are exposed during tool invocation chains. Adversaries who gain unauthorized access can execute privileged operations, access sensitive data, or manipulate business logic.
- Prompt injection attacks represent a threat vector specific to AI applications with tool access. Attackers craft malicious inputs designed to manipulate the AI's reasoning process, causing it to misuse tools or generate malicious parameters for tool invocations. The AI application may interpret attacker-controlled prompts as legitimate instructions, leading to unintended tool executions that compromise security controls.
- Privilege escalation risks emerge when AI applications chain multiple tools with varying permission levels. An attacker who compromises a low-privilege tool can potentially leverage the

AI's orchestration capabilities to access higher-privilege tools through unintended combinations. This risk intensifies in autonomous agent scenarios where the AI makes independent decisions about which tools to invoke and in what sequence.

- Resource exhaustion and API abuse pose operational and security risks when AI applications make excessive tool calls. AI-driven workloads can generate high volumes of tool invocations through reasoning loops or self-perpetuating execution patterns. Adversaries can exploit this behavior to launch denial-of-service attacks by crafting prompts that trigger resource-intensive tool chains, exhausting API limits and consuming compute resources.
- Supply chain vulnerabilities affect both upstream and downstream components in tool integration architectures. Upstream risks include compromised tool dependencies, malicious MCP servers, or vulnerable third-party APIs. Downstream risks involve insecure network routes between AI applications and external tools, man-in-the-middle attacks on tool communication channels, and exposure of sensitive data in transit.

Remediations

This section reviews the AWS services and features that address the risks that are specific to this capability.

Data protection

Encrypt tool inputs, outputs, and execution contexts in transit and at rest using AWS Key Management Service (AWS KMS) customer managed keys. Amazon Bedrock AgentCore encrypts all data at rest and in transit by default. Use TLS 1.2 or higher with AES-256 encryption for all tool communications.

Implement session isolation to prevent data leakage between tool executions. Amazon Bedrock AgentCore Runtime provides dedicated microVM architecture that isolates each session with separate CPU, memory, and file system resources. Sessions terminate automatically and purge all state data to prevent cross-contamination.

Store authentication credentials for external tool access in AWS Secrets Manager encrypted with customer managed keys. Configure Amazon Bedrock AgentCore Identity as a secure credential broker that retrieves credentials at runtime without exposing them to AI applications.

Apply Amazon Bedrock Guardrails to validate and filter tool inputs and outputs across all integration patterns. Configure guardrails to detect and block malicious parameters, sensitive data exposure, and policy violations before tools execute.

Identity and access management

Create custom service roles for AI application tool integration following the principle of least privilege. Grant permissions only for specific tools and AWS services that are required for your use case. Implement permission boundaries to prevent privilege escalation through unintended tool combinations.

Configure AgentCore Identity as a secure credential broker supporting Signature Version 4 (SigV4) signing for AWS services and OAuth 2.0 authentication for external APIs. Store credentials in AWS Secrets Manager with automatic rotation where supported by external services.

Implement fine-grained access controls through Amazon Bedrock AgentCore Gateway centralized tool management. Register tools explicitly and configure which AI applications can invoke each tool. Apply rate limiting and resource quotas at the identity level to prevent resource exhaustion from excessive tool calls.

Apply guardrails with identity context for persona-based content filtering. Configure your orchestration and agent layers to require identity invocation and creation for each scoped task rather than using default settings.

Network security

Use AWS PrivateLink to establish private connectivity to Amazon Bedrock AgentCore services. Create VPC endpoints for AgentCore Gateway and AgentCore Runtime to help ensure tool integration occurs through private network paths without internet exposure.

Deploy AI applications and AWS Lambda function tools within private subnets using restrictive security groups. Configure security group rules to allow only necessary communication between AgentCore Gateway and registered tools. Use AgentCore Gateway native VPC support for secure, isolated tool access.

Configure VPC endpoint policies to restrict service access to authorized AI applications only. Implement network-level rate limiting and traffic controls to prevent resource exhaustion. Use AWS Network Firewall to inspect traffic between AI applications and external tools for malicious patterns.

Logging and monitoring

Enable AWS CloudTrail to log tool invocation activities with user context attribution. Configure organization trails to capture cross-account tool access and maintain comprehensive audit trails. Forward all logs to the Log Archive account for centralized security analysis.

Configure Amazon CloudWatch to monitor tool executions and detect anomalous behavior. Create metrics for tool invocation rates, execution duration, failure patterns, and resource consumption across different integration types. Set CloudWatch alarms to alert when metrics deviate from established baselines.

Implement Amazon Bedrock AgentCore Observability for MCP servers integrated with AgentCore Gateway. Monitor agent behavior, multi-agent workflows, and tool chain executions. Use trace data to identify security issues, performance bottlenecks, and unusual access patterns.

For operating system (OS) tools, use AWS Systems Manager Session Manager to log session activity to Amazon CloudWatch Logs or Amazon S3. Deploy CloudWatch agents to collect OS-level metrics and logs. Use AWS Systems Manager Run Command to maintain history of commands and outputs for audit purposes.

Recommended AWS services

This section reviews the AWS services that are recommended to build this capability securely.

Amazon Bedrock AgentCore Runtime

AgentCore Runtime provides secure, serverless hosting environments for AI agents with complete session isolation using dedicated microVMs. Each user session runs with isolated CPU, memory, and file system resources, ensuring separation between users regardless of tool type.

Configure customer managed KMS keys for enhanced encryption control over session data. AgentCore Runtime automatically terminates sessions and sanitizes memory after completion. The service supports both real-time interactions and long-running workloads up to 8 hours while maintaining security isolation throughout execution.

Amazon Bedrock AgentCore Gateway

AgentCore Gateway provides centralized tool discovery and invocation using the Model Context Protocol (MCP). It supports multiple tool types including AWS Lambda functions, OpenAPI specifications, Smithy models, and MCP servers through a standardized interface.

Configure OAuth authorizers for gateway access and manage authentication credentials securely with AgentCore Identity. Create VPC endpoints for private connectivity and apply endpoint policies to restrict access to authorized AI applications. The gateway enforces mandatory TLS 1.2+ encryption for all communications by default.

Register tools explicitly through the gateway console or API. Configure tool-specific access controls, rate limits, and timeout values. Monitor tool usage through integrated CloudWatch metrics and CloudTrail logging.

Amazon Bedrock AgentCore Identity

AgentCore Identity serves as a secure credential broker supporting multiple authentication methods. These methods include AWS Signature Version 4 (SigV4) signing for native AWS services and OAuth 2.0 with JWT bearer tokens for external APIs. AgentCore Identity maintains a protected token vault using AWS KMS encryption for credential storage.

Configure integration with enterprise identity providers including Amazon Cognito, Okta, and Microsoft Entra ID. AgentCore Identity ensures complete separation between ingress authentication (verifying user identity) and egress authorization (accessing tools), preventing customer credentials from being forwarded to target services.

AWS Lambda

Lambda functions serve as custom tools for AI applications, providing serverless compute for business logic execution. Create AWS Identity and Access Management (IAM) execution roles with permissions scoped to invoke only registered tools and access required AWS services.

Configure Lambda functions within virtual private clouds (VPCs) for network isolation and apply resource-based policies to control which principals can invoke functions. Use environment variable encryption with customer managed KMS keys for sensitive configuration data. Set appropriate timeout values and memory limits to prevent resource exhaustion.

AWS Secrets Manager

Secrets Manager provides secure storage and automatic rotation of authentication credentials for external tool access. Store API keys, OAuth tokens, and database credentials with encryption using customer managed KMS keys.

Configure automatic credential rotation where supported by external services. Use fine-grained IAM policies to control which AI applications can retrieve specific credentials. Enable CloudTrail logging for all secret access operations to maintain audit trails.

Amazon Bedrock Guardrails

Amazon Bedrock Guardrails enables content filtering and validation for tool inputs and outputs. Configure content filters to block harmful content across multiple categories: hate, insults, sexual,

violence, misconduct, and prompt attacks. Set filter strength for each category based on your risk tolerance.

Define restricted topics to prevent AI applications from discussing sensitive subjects or internal systems. Create custom word filters tailored to your organization's sensitive terminology. Configure custom response messages that users see when content is blocked.

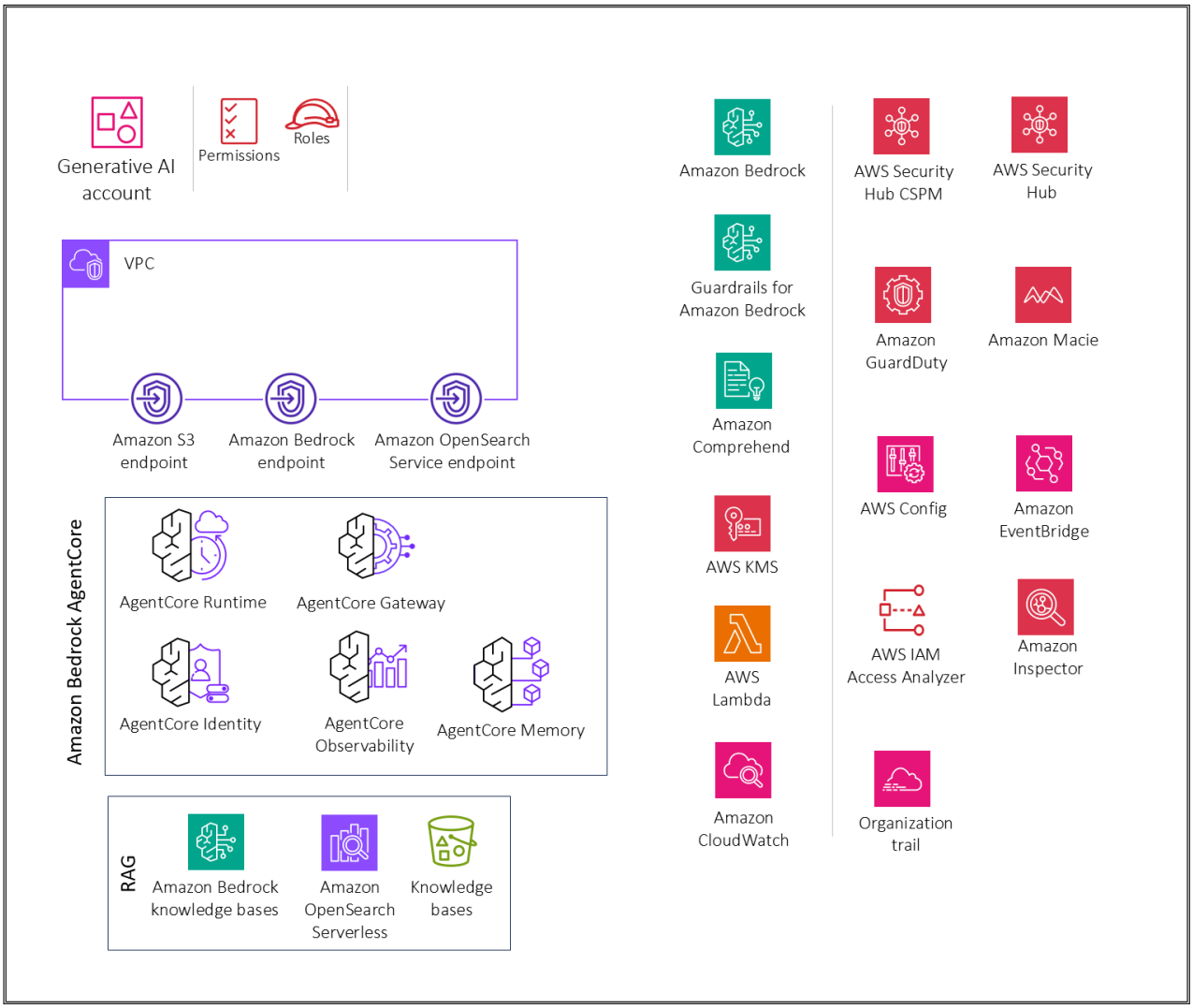
Apply guardrails consistently across all tool integration patterns by invoking them through the `InvokeModel` API with the `guardrailConfig` parameter. For AgentCore Gateway integrations, configure guardrails directly within gateway settings to filter both tool inputs and outputs before execution.

Use guardrail metrics in CloudWatch to monitor filtering effectiveness and identify potential security threats. Create alarms when guardrail activation rates exceed expected thresholds, which may indicate attack attempts or policy violations.

Capability 5. Providing secure access, usage, and implementation of generative AI agents

> Influence the future of the AWS Security Reference Architecture (AWS SRA) by taking a short survey.

The scope of this capability is to secure autonomous agent functionality for generative AI applications. The following diagram illustrates the AWS services recommended for the Generative AI account for this capability.



Rationale

AI agents extend foundation model (FM) capabilities by orchestrating chains of reasoning steps, tool invocations, and decision-making processes to accomplish complex tasks autonomously. Unlike simple model inference, agents maintain conversation context, make independent decisions

about which tools to invoke, and execute multi-step workflows based on user goals rather than explicit instructions.

Agents solve problems that require multiple interactions with external systems. For example, a customer service agent might retrieve order information from a database, check inventory through an API, process a refund through a payment system, and update a CRM. These actions are all based on a single customer request. The agent determines which tools to use, in what sequence, and how to handle errors or unexpected responses.

This capability addresses Scope 3 of the Generative AI Security Scoping Matrix. In Scope 3, your organization builds an agentic AI application using a pre-trained FM such as those models offered in Amazon Bedrock. You control your application, the tools agents can access, and customer data. The FM provider controls the pre-trained model and its training data.

Amazon Bedrock AgentCore provides a comprehensive platform for deploying and managing AI agents securely. AgentCore Runtime hosts agents with session isolation, AgentCore Gateway centralizes tool access, AgentCore Memory stores conversation history, AgentCore Identity manages authentication, and AgentCore Observability monitors agent behavior. Combined with Amazon Bedrock Guardrails, these services address the unique security challenges of autonomous agent systems.

Security considerations

Agentic AI applications face security risks that extend beyond those of traditional generative AI applications because of their autonomous decision-making capabilities and persistent state management. The combination of autonomy, tool access, and memory creates attack surfaces that require specialized security controls.

- Session isolation becomes critical when agents serve multiple users concurrently. Without proper isolation, one user's sensitive data could leak into another user's session through shared memory, cached context, or persistent state. Agents that maintain conversation history across sessions require secure memory stores. These memory stores prevent unauthorized access to historical interactions and protect against memory poisoning attacks where adversaries inject false information to manipulate future agent behavior.
- Excessive autonomy introduces risks when agents make independent decisions about tool invocations without sufficient constraints. An agent with broad tool access and minimal oversight can do the following:
  - Execute unintended operations.

  - Chain tools in ways that developers did not anticipate.
  - Escalate privileges by combining low-privilege tools to achieve high-privilege outcomes.

  The autonomous nature of agents makes it difficult to predict all possible execution paths, requiring defense-in-depth controls that limit scope when agents behave unexpectedly.
- Identity and access management complexity increases as agents authenticate with multiple systems on behalf of users. Improper credential management can expose user credentials to the agent runtime, fail to properly scope agent permissions, or allow agents to access resources beyond their intended authorization. Multi-agent architectures compound this complexity when orchestrator agents invoke subordinate agents, each requiring appropriate authentication and authorization at every step in the chain.
- Secure execution environments become necessary when agents run code or interact with websites. Code execution capabilities enable powerful agent functionality but create risks of arbitrary code execution, resource exhaustion, or access to the underlying host system. Browser automation allows agents to interact with web applications but introduces risks of credential exposure, cross-site scripting, or unintended actions on behalf of users.

Remediations

This section reviews the AWS services and features that address the risks that are specific to this capability.

Data protection

Implement session isolation through Amazon Bedrock AgentCore Runtime, which runs each user session in a dedicated microVM with isolated CPU, memory, and file system resources. This architecture provides complete separation between user sessions, preventing cross-session data contamination even when agents process requests concurrently. After session completion, AgentCore Runtime terminates the microVM and sanitizes memory, ensuring no data persists between sessions.

Secure agent memory through the Amazon Bedrock AgentCore Memory namespace structure for logical data isolation. Memory is organized by session ID, actor ID, and strategy ID, preventing users from accessing data belonging to other users. Configure short-term memory retention periods to the minimum required for your use case (up to 365 days). For long-term memory, which lacks built-in retention, implement automated deletion workflows using the AgentCore Memory API to comply with data retention policies.

Prevent memory poisoning by ensuring that users can't modify their session ID or actor ID. Don't include `ActorID` or `SessionID` values in system prompts where users could manipulate them. Implement input validation that rejects attempts to inject false information designed to corrupt the agent's memory and influence future behavior.

Encrypt agent data at rest by using AWS Key Management Service (AWS KMS) customer managed keys for AgentCore Memory resources, AgentCore Identity token vaults, AgentCoreGateway configuration, and Amazon CloudWatch log groups containing agent logs. This approach provides enhanced control over encryption key management and enables detailed audit trails of key usage.

Identity and access management

Design authentication architecture that addresses the following distinct authentication points:

- User authentication to invoke the agent
- Agent authentication to access tools and resources
- Tool authentication to access downstream systems

Each authentication point requires appropriate identity providers and credential management strategies.

Configure inbound authentication using AgentCore Identity, which supports both AWS credentials and OAuth 2.0. For AWS credentials, limit AWS Identity and Access Management (IAM) principals who can invoke agents by controlling access to the `InvokeAgentRuntime` API. Add IAM conditions to policies that specify the ARN of agents that are hosted by AgentCore, preventing unauthorized invocations. For OAuth 2.0, federate with your corporate identity provider for internal applications or select an identity provider that meets your requirements for external applications. Amazon Cognito integrates natively with AgentCore Runtime to facilitate OAuth authentication.

Assign IAM roles to agents running on AgentCore Runtime that provide minimum permissions required for agent functions. Follow the principle of least privilege by granting access only to specific tools, AWS resources, and secrets that the agent needs. Avoid broad permissions that enable privilege escalation through unintended tool combinations.

Centralize tool access through AgentCore Gateway, which manages both inbound authentication (verifying agent identity) and outbound authorization (connecting to tools). Configure the gateway with a separate identity store from the one used for user authentication. This identity store authenticates agents making calls to gateway targets using the OAuth Client Credentials flow.

Store client IDs and client secrets in AWS Secrets Manager rather than in code or environment variables and configure agents to retrieve credentials at runtime.

Implement outbound authorization using authentication based in IAM with AWS Signature Version 4 (SigV4) for AWS services, OAuth 2.0 for external APIs, or, if needed, API keys for third-party services. When using IAM-based authorization, scope the gateway service role to invoke only registered AWS Lambda functions and access only required secrets. When using API keys, store them securely in AgentCore Identity instead of in application code. Add granularity to tool access using OAuth scopes that limit which tools specific agents can invoke.

Configure tool authentication by assigning IAM policies to Lambda functions that interact with AWS resources. Scope policies to only the permissions that each tool needs, preventing tools from accessing resources beyond their intended function.

Network security

Deploy VPC endpoints for AgentCore Runtime and AgentCore tools to enable private connectivity without internet exposure. This architecture allows agents to access private resources, maintain secure communications within your network boundaries, and connect to enterprise data stores while preserving security isolation.

Configure AWS WAF to protect public-facing agent applications from common web exploits. Create custom AWS WAF rules that detect prompt injection patterns, rate limit requests to prevent abuse, and block malicious traffic before it reaches your agents.

Implement network-level monitoring through VPC Flow Logs to track traffic patterns between agents and tools. Configure flow logs to capture accepted and rejected traffic, providing visibility into network communications for security analysis and threat detection.

Application security

Evaluate tool architecture by assessing which capabilities agents require and whether those capabilities justify the associated risks. Limit agent access to mutative or destructive operations, implementing additional security controls for high-impact actions. Controls include instruction hardening that makes agent prompts resistant to manipulation, human-in-the-loop approval for sensitive operations, and least-privilege IAM roles that reduces access risks.

Deploy Amazon Bedrock Guardrails to protect agents from prompt attacks, prevent unwanted behavior, and limit hallucinations. Configure guardrails with content filters appropriate for your use case, define denied topics that agents should not discuss, and create custom word filters for

organization-specific sensitive terms. Deploy guardrail versions to production and configure agents to invoke the versioned guardrail as part of their response generation.

Implement pre-processing validation through Lambda functions that sanitize and validate input before passing it to agents. This additional layer of defense detects malicious prompts that attempt to bypass guardrails or manipulate agent behavior. Regularly test applications for prompt attacks using adversarial testing techniques.

Use AWS Security Agent to accelerate security reviews by analyzing architecture documents against AWS best practices and organizational requirements during the planning phase. It scales secure code analysis by automatically reviewing pull requests for common vulnerabilities and providing immediate remediation guidance within developer workflows. Additionally, the agent enables on-demand penetration testing to discover and report validated security vulnerabilities through tailored, multi-step attack scenarios.

Logging and monitoring

Enable AgentCore Observability to trace, debug, and monitor agent activity. Configure Transaction Search in CloudWatch and enable observability for agents hosted by AgentCoreRuntime. This approach provides visibility into agent behavior, including input and output prompts, reasoning traces, and tool invocations.

Monitor tool usage through AWS CloudTrail and CloudWatch to detect anomalous patterns. Create CloudWatch metrics that track tool invocation rates, execution duration, and error rates. Set alarms that trigger when metrics deviate from established baselines, indicating potential security issues or agent misbehavior.

Configure AgentCore Memory to emit logs to CloudWatch, providing visibility into data plane events such as `CreateEvent`, `DeleteEvent`, and `RetrieveMemoryRecords`. Use these logs to audit memory access patterns and detect unauthorized attempts to access or manipulate agent memory.

Implement centralized log aggregation by forwarding all agent-related logs to the Log Archive account. This approach enables security teams to correlate events across multiple agents and detect attack patterns that span multiple sessions or users.

Recommended AWS services

This section discusses the AWS services and features that address the security risks that are specific to this capability. In addition to the services in this section, use Amazon CloudWatch, AWS

CloudTrail, Amazon OpenSearch Serverless, Amazon S3, and Amazon Comprehend as explained in Capability 1 (model inference) and Capability 3 (RAG).

Amazon Bedrock AgentCore Runtime

AgentCore Runtime provides secure, serverless hosting for AI agents with complete session isolation using dedicated microVMs. Each user session runs with isolated CPU, memory, and file system resources, ensuring separation between users and preventing cross-session data contamination.

Configure customer managed AWS KMS keys for enhanced encryption control over session data. AgentCore Runtime automatically terminates sessions and sanitizes memory after completion, providing deterministic security even with non-deterministic AI processes. The service supports both real-time interactions and long-running workloads up to 8 hours while maintaining security isolation.

Amazon Bedrock AgentCore Memory

AgentCore Memory provides secure storage for agent conversation history and context across sessions. The service offers two memory types:

- *Short-term memory* for turn-by-turn interactions within a single session
- *Long-term memory* for persistent knowledge retention across multiple sessions

Configure short-term memory retention periods to the minimum required for your use case. Implement automated deletion workflows for long-term memory to comply with data retention policies. Use the namespace structure (session ID, actor ID, and strategy ID) to enforce logical data isolation between users. Encrypt memory resources with customer managed KMS keys and restrict IAM access to memory APIs (such as `ListMemoryRecords`, `GetMemoryRecord`, `CreateMemoryRecord`, and `DeleteMemoryRecord`) to authorized services only.

Amazon Bedrock AgentCore Gateway

AgentCore Gateway centralizes tool access and management, providing a single point of control for agent-tool interactions. The gateway manages both inbound authentication (verifying agent identity) and outbound authorization (connecting to tools), simplifying security architecture.

Configure the gateway with a separate identity store for agent authentication using OAuth Client Credentials flow. Implement outbound authorization using IAM-based authentication for AWS

services, OAuth 2.0 for external APIs, or API keys for third-party services. Create VPC endpoints for private connectivity and apply endpoint policies to restrict access to authorized agents. Encrypt gateway configuration with customer managed KMS keys.

Amazon Bedrock AgentCore Identity

AgentCore Identity serves as a secure credential broker for agents, supporting AWS Signature Version 4 (SigV4) signing, OAuth 2.0 with JWT bearer tokens, and API key authentication. The service maintains a protected token vault using AWS Key Management Service (AWS KMS) encryption for credential storage.

Configure integration with enterprise identity providers including Amazon Cognito, Okta, and Microsoft Entra ID. Implement credential rotation policies through integration with AWS Secrets Manager. AgentCore Identity ensures complete separation between ingress authentication (verifying user identity) and egress authorization (accessing tools), preventing credential exposure.

Amazon Bedrock AgentCore Observability

AgentCore Observability provides comprehensive monitoring, tracing, and debugging capabilities for agent behavior. Enable Transaction Search in CloudWatch to track agent execution paths, tool invocations, and reasoning traces.

Configure observability to capture input and output prompts, tool call parameters, and error conditions. Use trace data to identify security issues, performance bottlenecks, and unusual access patterns. Integrate with CloudWatch alarms to trigger automated responses when agents exhibit anomalous behavior.

Amazon Bedrock Guardrails

Amazon Bedrock Guardrails provides configurable safeguards that detect and filter harmful content, prevent prompt attacks, and reduce hallucinations. Configure content filters across multiple categories (hate, insults, sexual, violence, misconduct, and prompt attacks) with filter strength appropriate for your risk tolerance.

Define denied topics to prevent agents from discussing sensitive subjects or internal systems. Create custom word filters for organization-specific sensitive terminology. Implement contextual grounding checks to detect hallucinations and verify response accuracy against source documents. Deploy guardrail versions to production and configure agents to invoke versioned guardrails for consistent protection.

AWS Security Agent

AWS Security Agent is an autonomous agent that provides continuous application security validation across the software development lifecycle (SDLC). It functions as a virtual security engineer by conducting automated architectural reviews against organizational standards and performing on-demand penetration testing to identify exploitable vulnerabilities.

Configure the agent to analyze code bases and design documents for early vulnerability detection. It leverages context-aware reasoning to execute multi-step attack chains, discovering complex risks that traditional scanners miss. The agent integrates developer workflows to provide actionable remediation guidance and automated pull requests. The agent helps scale security validation with development velocity without using customer data for underlying model training.

Amazon GuardDuty

GuardDuty provides threat detection for agentic applications by monitoring AWS CloudTrail management events for suspicious and malicious activity. The service detects unauthorized access attempts, unusual API call patterns, and potential compromises of agent infrastructure.

Enable GuardDuty in the Security Tooling account as the delegated administrator for centralized management across the organization. Configure automated responses to GuardDuty findings using Amazon EventBridge rules that trigger remediation workflows when threats are detected.

Amazon Inspector

Amazon Inspector scans agent code for known software vulnerabilities, identifying security issues in Lambda functions, container images, and Amazon EC2 instances hosting agent components. The service provides continuous vulnerability assessment and prioritized findings based on risk.

Enable Amazon Inspector in the Security Tooling account as the delegated administrator for centralized vulnerability management. Configure automated scanning for all agent-related resources and integrate findings with your security information and event management (SIEM) system for comprehensive security monitoring.

AWS KMS

AWS Key Management Service (AWS KMS) helps you create and control cryptographic keys to help protect your data. Use customer managed AWS KMS keys to encrypt AgentCore Memory resources, AgentCore Identity token vaults, AgentCore Gateway configuration, and CloudWatch log groups

that contain agent logs. Customer managed KMS keys provide enhanced control over encryption key management, enable detailed audit trails of key usage, and support key rotation policies.

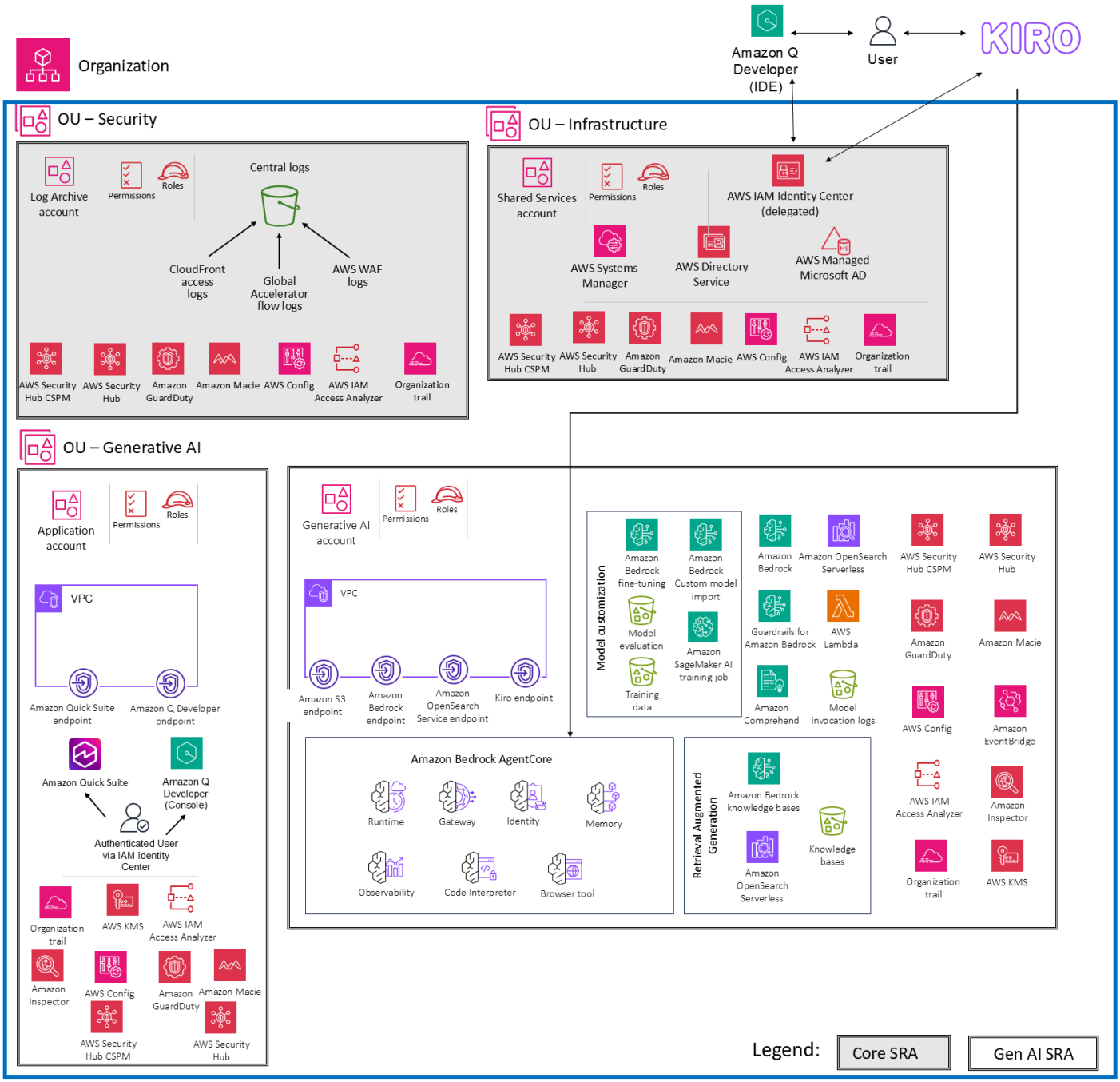
Configure key policies that grant encryption and decryption permissions only to authorized services and IAM roles. Enable CloudTrail logging for all KMS key usage to maintain comprehensive audit trails of data access.

Use CloudWatch, CloudTrail, Amazon OpenSearch Serverless, Amazon S3, and Amazon Comprehend as explained in Capability 1 (model inference) and Capability 3 (RAG).

Capability 6. Providing secure access, usage, and implementation for AI applications

> Influence the future of the AWS Security Reference Architecture (AWS SRA) by taking a short survey.

The scope of this capability is to secure user-facing AI applications that provide direct access to AI capabilities. The following diagram illustrates the AWS services recommended for the Generative AI account for this capability.



Rationale

User-facing AI applications enable organizations to deliver generative AI capabilities directly to end users through web interfaces, mobile applications, and integrated workflows. These applications include Amazon Q Developer for AI-assisted software development, Amazon Quick for enterprise productivity and business intelligence, and Kiro for agentic development environments. Each

application provides distinct capabilities while requiring consistent security controls to protect user data, prevent misuse, and maintain organizational governance.

This use case refers to Scope 3 of the Generative AI Security Scoping Matrix, where your organization deploys user-facing AI applications using pre-trained foundation models. In this scope, you control the application interface, user authentication, data access permissions, and usage policies, whereas the AI service provider controls the underlying models and infrastructure.

> **Note**
>
> Although this guidance focuses on AI applications managed by AWS, similar principles apply to custom-built AI applications and third-party AI services integrated into your environment.

Security considerations

When you provide users with direct access to AI applications, you should address these key security considerations:

- User authentication and authorization across multiple AI application types with varying sensitivity levels
- Data protection for user inputs, conversation history, and AI-generated outputs that might contain sensitive organizational information
- Content filtering and guardrails to prevent inappropriate use, prompt injection attacks, and generation of harmful content
- Usage monitoring and governance to track AI application adoption, detect anomalous behavior, and maintain compliance with organizational policies and controls

Remediations

This section reviews the AWS services and features that address the risks that are specific to this capability.

Data protection

Encrypt user inputs, conversation history, and AI-generated outputs in transit and at rest using AWS Key Management Service (AWS KMS) customer managed keys and TLS 1.2. Amazon Q

Developer, Quick, and Kiro provide comprehensive encryption by default, with options for customer managed keys to maintain enhanced control over encryption.

Implement session isolation to prevent data leakage between user sessions and maintain separation of user contexts across different AI applications. Configure data retention and memory policies that align with organizational requirements and regulatory obligations for AI-generated content and user interaction history. For more information about user-level context separation and conversation history isolation, see Enabling identity-enhanced console sessions in the AWS IAM Identity Center documentation.

Store application credentials and API keys in AWS Secrets Manager with customer managed key encryption. Configure automatic credential rotation where supported and implement fine-grained access controls to limit which users and applications can retrieve specific credentials.

Apply content filtering and validation for user inputs and AI-generated outputs across all application types.

Identity and access management

Use AWS IAM Identity Center for centralized identity management across all AI applications. Integrate with enterprise identity providers including Amazon Cognito, Okta, and Microsoft Entra ID to provide consistent authentication and single sign-on capabilities. For information about Amazon Q Developer integration, see Getting started with IAM Identity Center in the Amazon Q Developer documentation. For information about integrating Quick with IAM Identity Center, see Granting Quick access through IAM Identity Center integration in the *Choosing the right access approach for Amazon Quick* AWS Prescriptive Guidance guide. For information about Kiro, see its onboarding quickstart documentation. For more information, see Configure access to your applications in the IAM Identity Center documentation.

Create custom IAM policies that implement least-privilege access for AI application usage. Define granular permissions that control which users can access specific AI features, applications, and data sources based on their organizational roles and responsibilities. Implement permission data boundaries and service control policies to prevent privilege escalation through AI application features.

Configure access controls that limit AI applications to accessing only the data sources and AWS services necessary for their intended functionality. For more information, see How Amazon Q Developer works with IAM in the Amazon Q Developer documentation. For information about Quick, see Using IAM in the Quick documentation. For information relevant to Kiro, see How Kiro

works with IAM in the Kiro documentation. For more information about implementing least-privilege access with IAM for both human and workload users, see Security best practices in IAM in the IAM documentation.

Apply rate limiting and usage quotas at the user and application level to prevent resource exhaustion and control costs. Monitor usage patterns to detect anomalous behavior that might indicate compromised credentials or policy violations. For information about monitoring of API quota usage against service limits for Quick, see Monitoring and maintenance in the Quick documentation.

Network security

Deploy AI applications within private subnets using AWS PrivateLink for private connectivity to AWS services. Create VPC endpoints for Amazon Bedrock, Amazon Q Developer, and other AI services to help ensure that all traffic remains within the AWS network. For more information about VPC endpoints, see the following resources:

- Amazon Q Developer and interface endpoints (AWS PrivateLink) in the Amazon Q Developer documentation
- Quick and interface VPC endpoints (AWS PrivateLink) in the Quick documentation
- Kiro and interface endpoints (AWS PrivateLink) in the Kiro documentation
- Access an AWS service using an interface VPC endpoint in the Amazon Virtual Private Cloud documentation

Configure security groups and network access control lists that restrict traffic to only necessary communication paths. Implement network segmentation to isolate AI application infrastructure from other organizational workloads, based on data sensitivity and compliance requirements.

Use AWS WAF to protect web-based AI application interfaces from common attacks including SQL injection, cross-site scripting, and bot traffic. Configure custom rules to detect and block potential prompt injection patterns and implement rate limiting at the network edge. For information about an example pattern that integrates AWS WAF with a web-based AI application, see Securing Amazon Q Business Web Experiences with AWS Amplify and AWS WAF (AWS Blog post).

Enforce TLS 1.2 or higher for all user connections to AI applications. Use AWS Certificate Manager for certificate issuance and automatic rotation to maintain secure encrypted communications between users and AI services.

Logging and monitoring

Enable AWS CloudTrail to log all AI application access and usage activities with user context attribution. Configure organization trails to capture cross-account access and maintain comprehensive audit trails for compliance and security investigations.

Configure Amazon CloudWatch to monitor AI application usage patterns, error rates, and performance metrics. Create custom metrics for tracking user adoption, feature usage, and potential security events across different AI applications.

Implement application-specific observability features including Amazon Q Developer usage analytics, Quick audit logging, and the telemetry collection available in Kiro. Use these specialized monitoring capabilities to gain visibility into AI-specific behaviors and usage patterns.

Configure Amazon EventBridge rules to automate responses to security events including unauthorized access attempts, policy violations, and anomalous usage patterns. Forward all logs to the Security Tooling account for centralized analysis and long-term retention. For more information, see AWS service events.

Recommended AWS services

This section reviews the AWS services and features that address the security risks that are specific to this capability:

Amazon Q Developer

Amazon Q Developer is an AI-powered productivity tool for software development teams that integrates directly into integrated development environments (IDEs) and command line interfaces (CLIs). It provides context-aware code suggestions, automated code reviews, security scanning, and documentation generation while maintaining enterprise security controls.

Configure Amazon Q Developer with IAM Identity Center for centralized authentication and access control. Enable customer managed AWS KMS keys for conversation history encryption and code analysis data. Implement resource-based policies to control which code repositories Amazon Q Developer can access. Configure code scanning sensitivity levels and customize security scanning policies to align with organizational security requirements.

Amazon Quick

Quick combines conversational business intelligence with generative AI capabilities to transform enterprise data into actionable insights. The suite includes Amazon Quick Sight for data analysis

and visualization, enabling users to interact with business data using plain language questions while maintaining comprehensive security controls.

Implement row-level security (RLS) in Quick Sight to ensure users can only access authorized data based on their role and permissions. Configure column-level security to mask sensitive fields from unauthorized users. Use private virtual private cloud (VPC) connectivity to establish secure connections to data sources. Enable embedded analytics with identity federation to maintain consistent access controls when integrating Quick capabilities into custom applications.

Kiro

Kiro provides an agentic development environment that accelerates software delivery through AI-assisted workflows and automated implementation planning. Kiro transforms high-level specifications into detailed implementation plans with automated code generation while maintaining security through comprehensive isolation and encryption.

Configure Kiro with customer managed AWS KMS keys for session data encryption and persistent storage. Implement fine-grained access controls to limit which users can initiate agentic workflows and access generated code. Enable VPC connectivity to establish private network paths between Kiro and internal code repositories. Configure audit logging to track all code generation activities and link them to originating user requests for comprehensive traceability.

AWS IAM Identity Center

IAM Identity Center provides centralized identity management for all AI applications with consistent authentication and authorization. It enables single sign-on across multiple AWS accounts and business applications including Amazon Q Developer, Quick, and Kiro.

Configure IAM Identity Center with your enterprise identity provider to maintain consistent user access controls. Create permission sets that define specific access levels for different user roles. Implement attribute-based access control (ABAC) to dynamically adjust permissions based on user attributes. Enable multi-factor authentication (MFA) for all AI application access to enhance security posture and protect against credential theft.

AWS Secrets Manager

Secrets Manager securely stores and manages API keys, database credentials, and service tokens that are required by AI applications. It automatically rotates credentials according to configured schedules and provides a centralized service for secure credential distribution.

Store all AI application credentials in Secrets Manager with encryption by using customer managed KMS keys. Configure automatic rotation for database credentials, API keys, and OAuth tokens where supported. Implement fine-grained access policies to control which AI services can retrieve specific secrets. Enable CloudTrail logging for all secret access operations to maintain a comprehensive audit trail.

AWS WAF

AWS WAF protects AI application interfaces from common web vulnerabilities and specialized attacks against generative AI systems. It provides customizable security rules to filter malicious traffic and protect against distributed denial-of-service (DDoS) attacks.

Configure AWS WAF with managed rule groups to protect against common vulnerabilities including SQL injection and cross-site scripting. Create custom rules to detect and block prompt injection patterns targeting AI applications. Implement rate-based rules to prevent abuse and resource exhaustion from automated or excessive queries. Enable logging to Amazon Simple Storage Service (Amazon S3) for comprehensive traffic analysis and security investigation.

Amazon CloudWatch

CloudWatch provides comprehensive monitoring and observability for all AI applications through metrics collection, log aggregation, and automated alerting. It enables detection of anomalous usage patterns and security events across your AI application portfolio.

Create custom dashboards to monitor key AI application metrics including usage rates, error frequencies, and performance indicators. Configure metric filters to extract actionable data from application logs. Implement CloudWatch alarms to detect potential security incidents including unusual access patterns or policy violations. Set up composite alarms that correlate multiple metrics to identify complex security scenarios with higher confidence. For more information, see the following resources:

- Monitoring Amazon Q Developer with Amazon CloudWatch in the Amazon Q Developer documentation
- Monitoring Amazon Quick usage using CloudWatch Logs in the Quick documentation
- Monitoring and tracking on the Kiro website

AWS CloudTrail

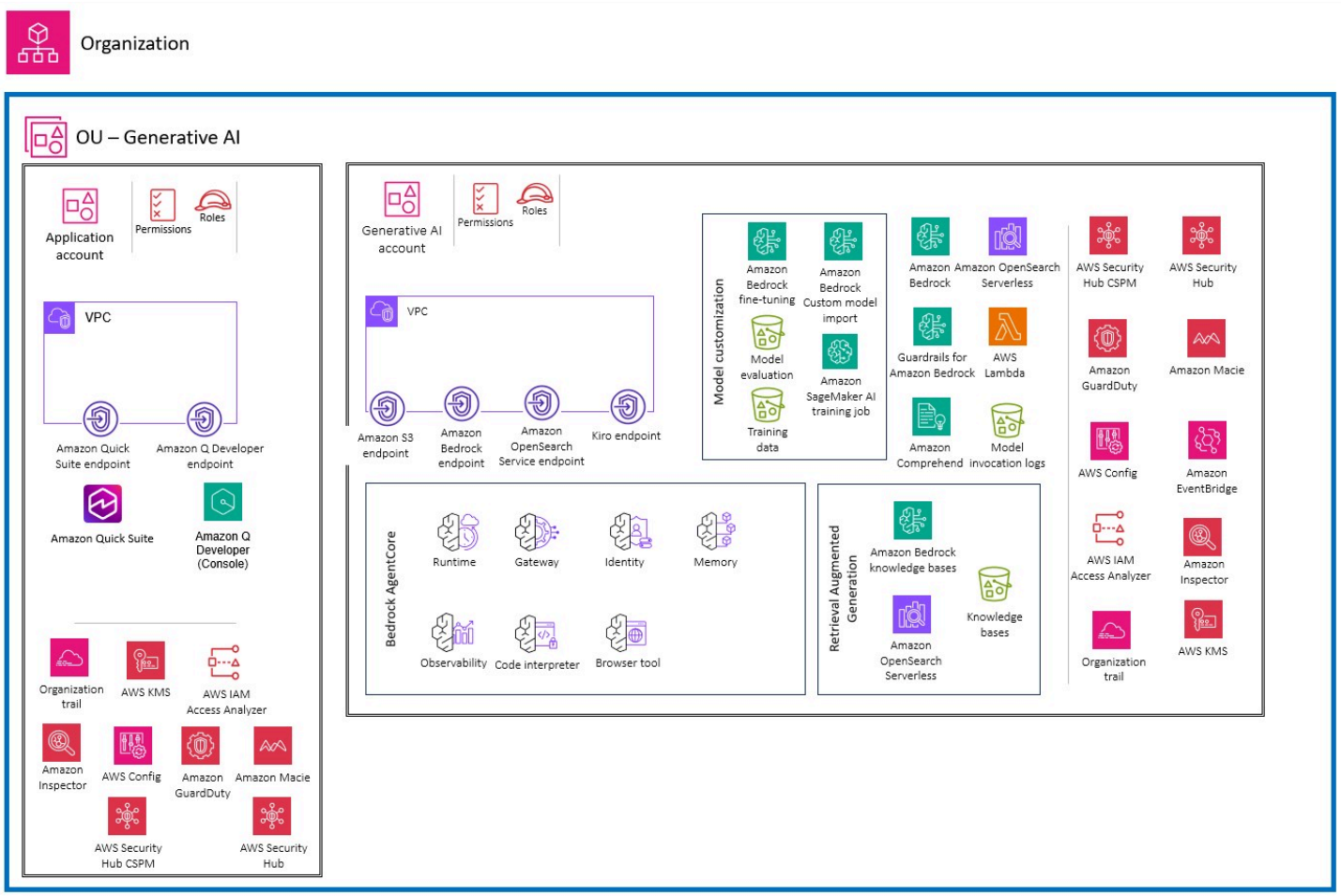
CloudTrail provides comprehensive audit logging for all API calls and user activities across your AI application environment. It captures detailed information about each action including the identity, IP address, timestamp, and parameters used.

Enable organization trails to capture activities across all AWS accounts and forward them to centralized storage in the Log Archive account. Configure log file validation to ensure integrity of audit trails. Implement event selection to capture both management and data events related to AI application usage. Use CloudTrail Lake to create SQL-based queries for security investigations and compliance reporting on AI application activities. For more information, see the AWS CloudTrail section of Security OU - Security Tooling account in the *AWS SRA – core architecture* guide.

Integrating a traditional cloud workload with Amazon Bedrock

Influence the future of the AWS Security Reference Architecture (AWS SRA) by taking a short survey.

The scope of this use case is to demonstrate a traditional cloud workload that is integrated with Amazon Bedrock to take advantage of generative AI capabilities. The following diagram illustrates the Generative AI account in conjunction with an example application account.



The *Generative AI account* provides generative AI functionality by using Amazon Bedrock. The *Application account* hosts an example workload. The AWS services that you use in this account depend on your requirements. Interactions between the Generative AI account and the Application account use the Amazon Bedrock APIs.

The *Application account* is separated from the Generative AI account to help group workloads based on business purposes and ownership. This separation constrains access to sensitive data in the generative AI environment and supports the application of distinct security controls by environment. Keeping the traditional cloud workload in a separate account also helps limit the scope of impact of adverse events.

You can build and scale enterprise generative AI applications around various use cases that are supported by Amazon Bedrock. Common use cases include text generation, virtual assistance, text and image search, text summarization, and image generation. Depending on your use case, your application component interacts with one or more Amazon Bedrockcapabilities such as foundation model (FM) inference, knowledge bases, agents, and model customization.

Application account

The Application account hosts the primary infrastructure and services to run and maintain an enterprise application. In this context, the Application account acts as the traditional cloud workload, which interacts with the Amazon Bedrock managed service in the Generative AI account. For general security best practices for securing this account, see Workloads OU - Application account in the *AWS SRA – core architecture guide*.

Identity propagation and access control

Implement identity propagation throughout your generative AI application architecture. When your application uses Retrieval Augmented Generation (RAG), propagate the user's identity from the application in the form of metadata to the knowledge base. The knowledge base enforces controls through metadata filtering, ensuring users only access data they are authorized to view.

For agentic applications, make sure every agent propagates the application user's identity to all systems it interacts with, including data sources, knowledge bases, and external APIs. Each system must understand the user identity, limit its responses to actions the user is authorized to perform, and respond with data the user is authorized to access. Use Amazon Bedrock AgentCore Identity to manage authentication and authorization across the agent workflow, maintaining separation between ingress authentication (verifying user identity) and egress authorization (accessing tools and resources).

Application security

Apply standard application security best practices as you would for other applications. Protect your web application infrastructure using AWS WAF to defend against common web exploits including

SQL injection, cross-site scripting, and request floods. Configure rate limiting to prevent resource exhaustion and control costs, as application invocations trigger model inference API calls that incur charges.

Restrict direct access to foundation model inference endpoints to control costs and monitor activity. Use AWS Identity and Access Management (IAM) policies to control permissions to invoke inference actions on Amazon Bedrock base models. Implement least-privilege access by granting only the minimum permissions required for your application to function.

Prompt injection protection

Traditional web application firewalls don't protect against prompt injection threats because these threats use natural language text rather than code patterns that firewalls detect. Implement Amazon Bedrock Guardrails to protect against prompt injection attacks and ensure model safety. Configure guardrails with prompt attack detection enabled, set appropriate filter strength based on your risk tolerance, and deploy versioned guardrails to production.

Add pre-processing validation through AWS Lambda functions that sanitize and validate input before passing it to foundation models or agents. This additional layer of defense detects malicious prompts that attempt to bypass guardrails or manipulate model behavior. Regularly test your applications for prompt attacks using adversarial testing techniques.

Data protection

Encrypt data in transit between the Application account and Generative AI account by using TLS 1.2 or higher. Encrypt data at rest using AWS Key Management Service (AWS KMS) customer managed keys for application data stores, conversation logs, and any cached model responses.

Implement data classification schemes to categorize data based on sensitivity and criticality. Use Amazon Macie to detect and classify sensitive data in Amazon S3 buckets that contain user prompts, conversation logs, and application data. Deploy Amazon Comprehend personally identifiable information (PII) detection or Amazon Bedrock Guardrails to detect and redact sensitive information in both model inputs and outputs before processing or storage.

Network security

Deploy your application within a virtual private cloud (VPC) by using private subnets for application tiers that don't require direct internet access. Use VPC endpoints to establish private

connectivity to Amazon Bedrock services, to help make sure traffic between your application and Amazon Bedrock doesn't traverse the public internet.

Configure security groups with restrictive rules that allow only necessary traffic between application tiers. Implement network access control lists (network ACLs) as an additional defensive layer with explicit allow rules for required traffic and a default-deny posture. Deploy AWS Network Firewall for deep packet inspection of traffic between application tiers and to detect unusual request patterns that might indicate attacks.

Logging and monitoring

Enable comprehensive logging and monitoring for your generative AI application. Configure AWS CloudTrail to log all API calls to Amazon Bedrock, capturing user identity, request parameters, and response metadata. Forward CloudTrail logs to the Log Archive account for centralized security analysis.

Configure CloudWatch to monitor application metrics including model invocation rates, response latencies, error rates, and token usage patterns. Create CloudWatch alarms that trigger when metrics deviate from established baselines, indicating potential security issues, service degradation, or unexpected usage patterns.

Monitor Amazon Bedrock Guardrails metrics to track how frequently content is filtered or blocked, providing visibility into potential security threats or policy violations. Analyze trends in guardrail activations to identify emerging attack patterns or areas where additional security controls might be needed.

Implement centralized log aggregation using Amazon OpenSearch Service or integrate with third-party security information and event management (SIEM) platforms. Configure automated pattern analysis and threat detection to identify anomalous behavior including unusual traffic volumes, connections to unexpected destinations, or communication patterns that deviate from established baselines.

Generative AI account

Depending on the use case, the Generative AI account hosts all generative AI activities. These include model inference (Capability 1), model customization (Capability 2), Retrieval Augmented Generation (RAG) with knowledge bases (Capability 3), tool integration (Capability 4), autonomous agents (Capability 5), and end-user AI applications (Capability 6).

Foundation model inference

Implement the security controls described in Capability 1 for foundation model (FM) inference. Deploy AWS WAF as the first line of defense against malicious requests targeting your AI applications. Configure rate limiting to prevent resource exhaustion attacks and implement AWS Managed Rules for the Core rule set managed rule group and the Known bad inputs managed rule group.

Use Amazon Bedrock Guardrails to filter inputs and outputs across multiple harmful categories: hate, insults, sexual, violence, misconduct, and prompt attacks. Configure filter strength for each category based on your risk tolerance. Define restricted topics to prevent models from discussing sensitive subjects or internal systems.

Model customization

If your use case requires model customization, implement the security controls described in Capability 2. Encrypt the model customization job, output files, and resulting custom model by using customer managed keys in AWS Key Management Service (AWS KMS). Store training and validation data in Amazon S3 buckets with encryption, versioning, and access logging enabled.

Use a virtual private cloud (VPC) with no internet access for model customization jobs. Create Amazon S3 VPC endpoints to allow customization jobs to access training data buckets without internet exposure. This approach helps to ensure that training data and model artifacts remain private throughout the customization process.

Knowledge bases and RAG

For applications using RAG, implement the security controls described in Capability 3. Encrypt knowledge base data in transit and at rest using customer managed AWS KMS keys. Configure data ingestion jobs with customer managed keys and implement metadata filtering for secure retrieval based on user attributes.

Use Amazon Macie to detect and classify sensitive data in Amazon S3 buckets that contain knowledge base data sources. Implement input validation to protect knowledge bases from malicious content by scanning files for malware before uploading them to data sources. Use Amazon Comprehend to detect and redact sensitive information in documents before indexing them in your knowledge base.

Tool integration

For applications that extend AI capabilities through tool integration, implement the security controls described in Capability 4. Use Amazon Bedrock AgentCore Gateway to centralize tool discovery and invocation through the Model Context Protocol (MCP). Configure OAuth authorizers for gateway access and use Amazon Bedrock AgentCore Identity to manage authentication credentials securely.

Deploy AI applications and AWS Lambda function tools within private subnets by using restrictive security groups. Create VPC endpoints for AgentCore Gateway and Amazon Bedrock AgentCore Runtime to help make sure that tool integration occurs through private network paths. Apply rate limiting and resource quotas at the identity level to prevent resource exhaustion from excessive tool calls.

Autonomous agents

For agentic applications, implement the security controls described in Capability 5. Use Amazon Bedrock AgentCore Runtime to host agents with complete session isolation by using dedicated microVMs. Configure customer managed KMS keys for Amazon Bedrock AgentCore Memory resources, AgentCore Identity token vaults, and AgentCore Gateway configuration.

Implement authentication architecture that addresses user authentication to invoke the agent, agent authentication to access tools and resources, and tool authentication to access downstream systems. Assign IAM roles to agents that provide minimum permissions required for agent functions. Enable Amazon Bedrock AgentCore Observability to trace, debug, and monitor agent activity.

End-user AI applications

For end-user AI applications described in Capability 6, implement appropriate authentication and authorization controls based on your user population. For internal applications, federate with your corporate identity provider. For external applications, use Amazon Cognito or another identity provider that meets your requirements.

Implement user session management that maintains security boundaries between users. Use Amazon Bedrock Guardrails to filter content based on user context and persona. Monitor user interactions for anomalous patterns that might indicate account compromise or malicious activity.

Conclusion

The architectures presented in this guide offer a comprehensive framework for organizations that use AWS services to take advantage of generative AI capabilities securely and efficiently. These architectures combine the fully managed functionality of Amazon Bedrock with security best practices to provide a solid foundation for integrating generative AI into traditional cloud workloads and organizational processes.

The specific capabilities covered—foundation model inference, model customization, retrieval augmented generation, tool integration, autonomous agents, and end-user applications—address a wide range of potential applications and scenarios. This guidance equips organizations with the necessary understanding of Amazon Bedrock services and their inherent and configurable security controls. With this knowledge, organizations can make informed decisions tailored to their unique infrastructure, applications, and security requirements.

By implementing the security controls described across all six capabilities, organizations can build generative AI applications that maintain strong security postures while delivering innovative functionality. The separation of concerns between Application and Generative AI accounts, combined with defense-in-depth security controls at every layer, provides the foundation for secure, scalable generative AI deployments.

Contributors

The following individuals contributed to this guide.

- Avik Mukherjee, AWS Senior Security SA
- Abdullah Ali, AWS Technical Program Manager
- Emmanuel Isimah, AWS Senior Solutions Architect
- Mac Stevens, AWS Senior Solutions Architect
- Pierre-Yves Gillier, AWS Senior Startup Solutions Architect
- Victor Okonyia, AWS Technical Account Manager
- Lilian Alba Rodriguez, AWS Associate Solutions Architect
- Kevin Low, AWS Security Solutions Architect
- James Ferguson, AWS Principal Solutions Architect
- Anthony Harvey, AWS Senior Security Specialist SA
- Maitreya Ranganath, AWS Principal Security SA
- Joe Wagner, AWS Senior Solutions Architect
- Pravin Nair, AWS Senior Security SA
- Riggs Goodman III, AWS Principal AI Security SA
- Christopher Rae, AWS, Principal Worldwide Security Specialist

Document history

The following table describes significant changes to this guide.

Change	Description	Date
Reorganization and new content	Reorganized guide and added new content throughout.	February 24, 2026
Initial publication as standalone guide	Converted from a chapter in the AWS SRA – core architecture guide to an individual guide.	December 22, 2025