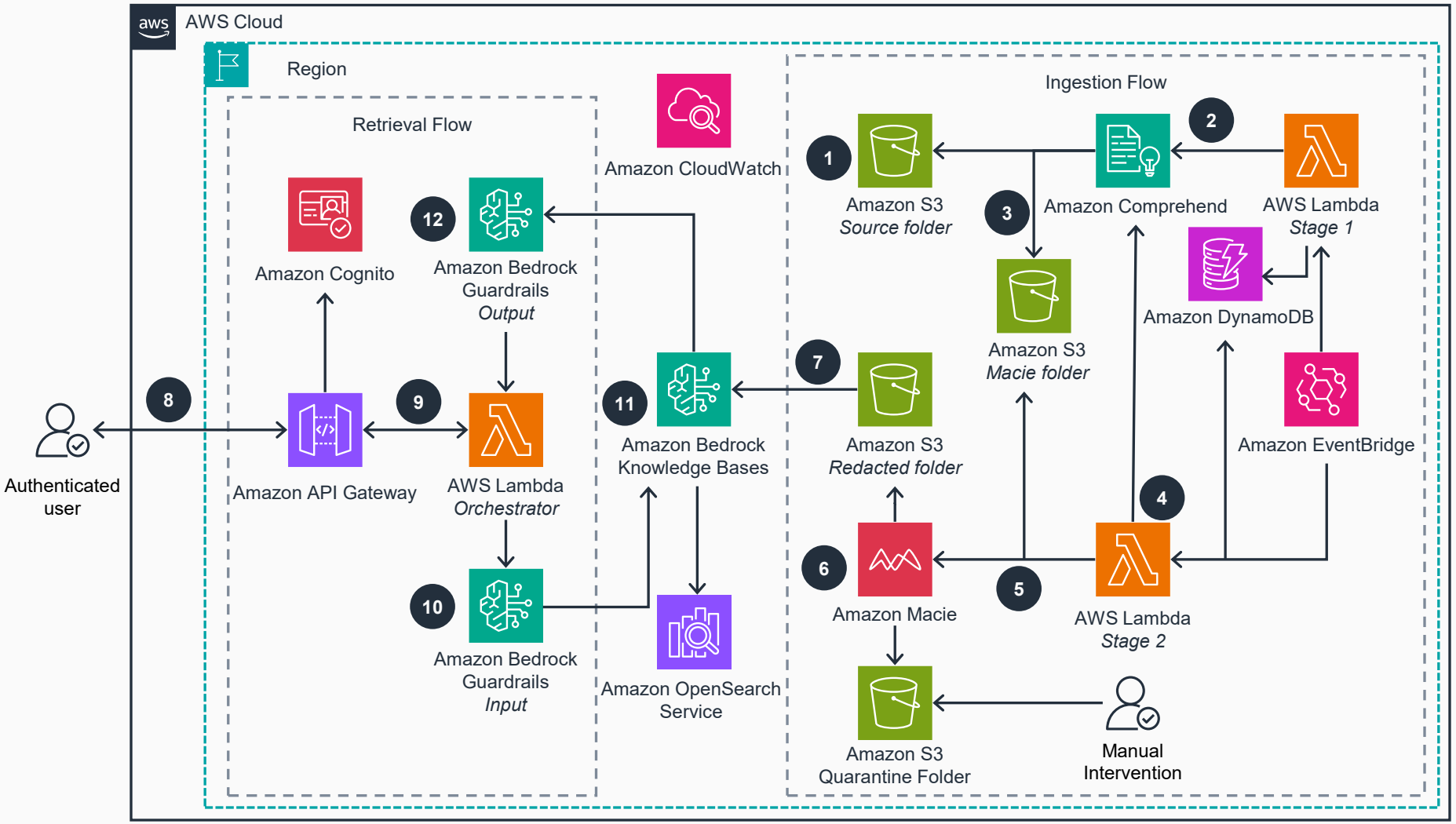
Guidance for Securing Sensitive Data in RAG Applications Using Amazon Bedrock

Data redaction at storage level

This architecture diagram shows how customers can safely ingest sensitive documents through automated redaction and verification processes while enabling secure, guardrail-protected access to their knowledge base without compromising sensitive information. This slide shows Steps 1-8.



1. The document ingestion flow initiates when documents containing sensitive data are uploaded to the source folder of the **Amazon Simple Storage Service (Amazon S3)** bucket.
2. **Amazon EventBridge** triggers **AWS Lambda** Stage 1 function, which initiates the **Amazon Comprehend** personally identifiable information (PII) redaction process and records the job ID information in the **Amazon DynamoDB** table.
3. The **Amazon Comprehend** job redacts PII entities, and redacted documents are moved to the macie_folder in the **S3** bucket.
4. During the **Amazon Comprehend** redaction process, **Amazon EventBridge** triggers **Lambda** Stage 2 function to monitor the redaction job status.
5. Upon completion of PII redaction, the **Lambda** Stage 2 function initiates a secondary verification using **Amazon Macie**.
6. The **Amazon Macie** job scans for sensitive information. Documents with severity ≥ 3 are moved to a quarantine folder, while documents with severity < 3 are moved to the redacted folder.
7. **Amazon Bedrock Knowledge Bases** processes documents from the **S3** bucket redacted folder, segments them into chunks, and securely indexes them in the **Amazon OpenSearch Service** vector store for RAG applications.
8. Users submit requests through **Amazon API Gateway** with their prompt and login credentials.

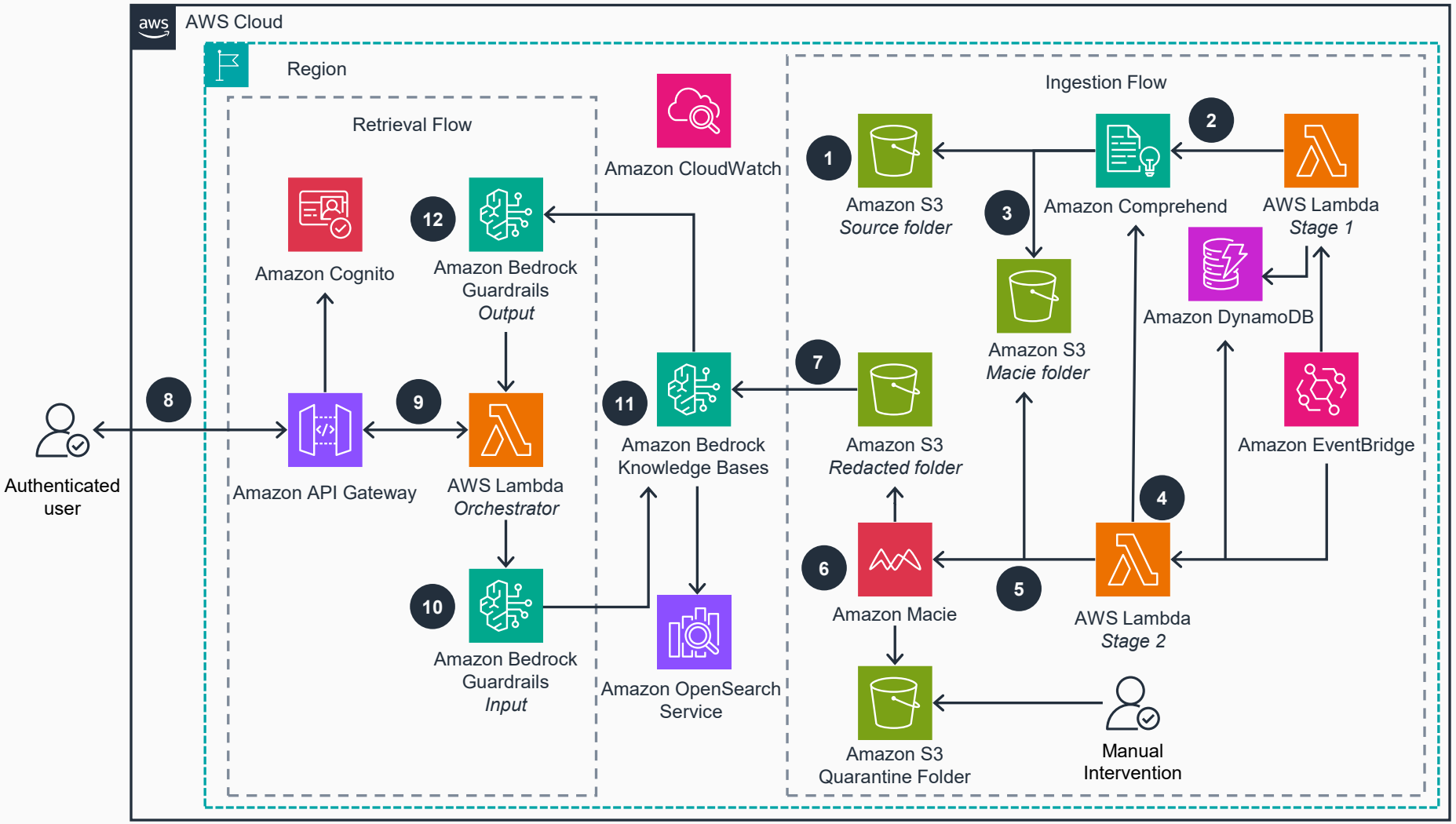
Reviewed for technical accuracy July 23, 2025

AWS Reference Architecture

Guidance for Securing Sensitive Data in RAG Applications Using Amazon Bedrock

Data redaction at storage level

This architecture diagram shows how customers can safely ingest sensitive documents through automated redaction and verification processes while enabling secure, guardrail-protected access to their knowledge base without compromising sensitive information. This slide shows Steps 9-12.



9 **API Gateway** authenticates credentials through **Amazon Cognito**, then forwards the user claims and prompt to the **Lambda** orchestrator.

10 The input prompt undergoes validation against pre-configured **Amazon Bedrock Guardrails**, with requests being blocked if validation fails.

11 Following successful guardrail validation, the prompt is sent for contextual retrieval and response generation using **Amazon Bedrock Knowledge Bases**.

12 The generated response undergoes output guardrail evaluation; failed validations result in blocked responses. Successfully validated responses are securely transmitted back to the user through **API Gateway**.

.

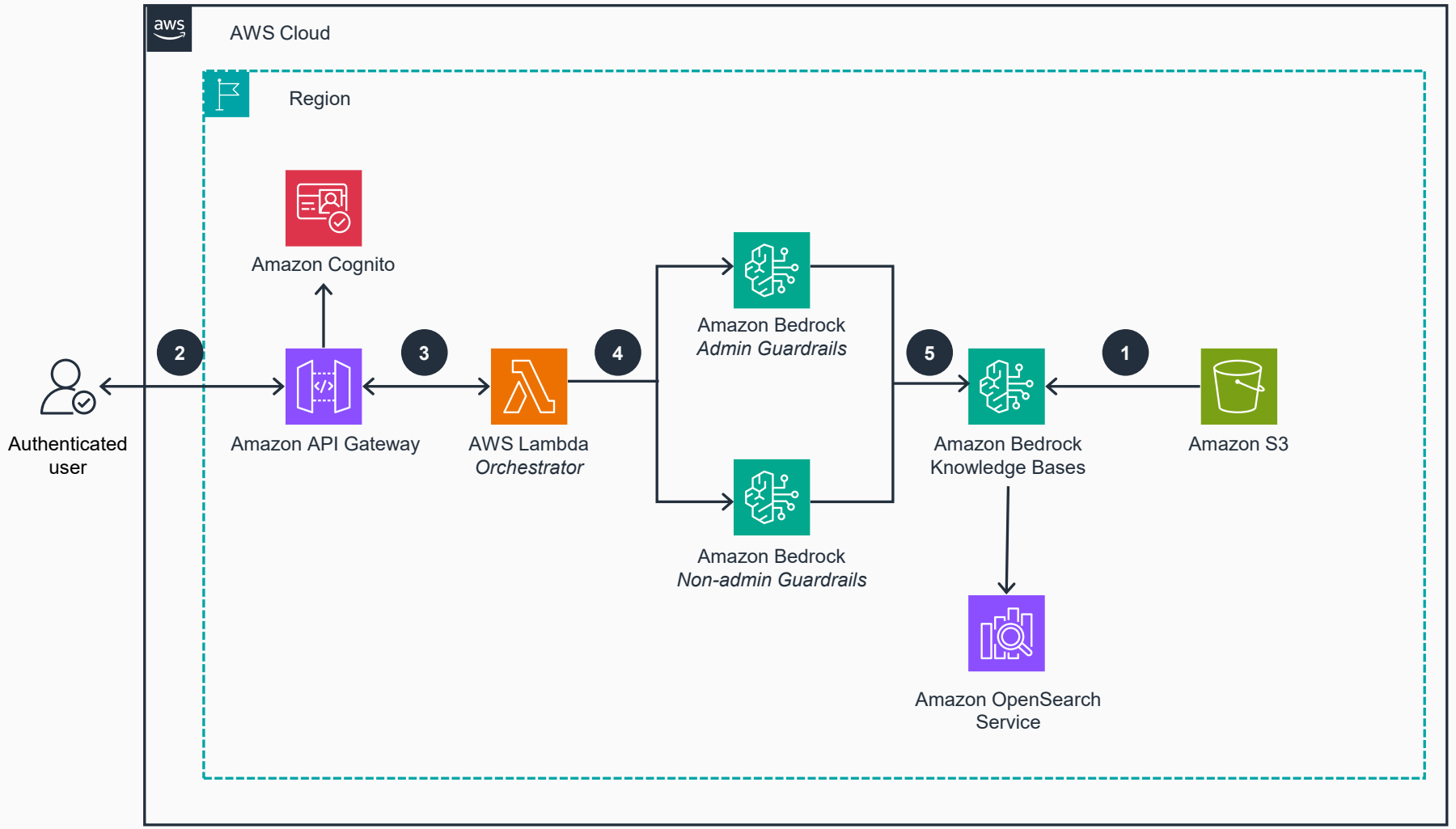
Reviewed for technical accuracy July 23, 2025

AWS Reference Architecture

Guidance for Securing Sensitive Data in RAG Applications Using Amazon Bedrock

Role based access to sensitive data

This architecture diagram shows how customers can implement role-based access control for sensitive data in RAG applications using metadata filtering and personalized guardrails, ensuring users only access information appropriate for their authorization level while maintaining the security of sensitive content.



1. Documents in the **S3** bucket are processed and indexed by **Amazon Bedrock Knowledge Bases** with appropriate metadata attributes defining access permissions.
2. Users authenticate through **API Gateway**, with **Amazon Cognito** validating credentials and determining the user's role (admin/non-admin).
3. **API Gateway** forwards the authenticated request with user claims to the **Lambda** Orchestrator function.
4. The **Lambda** Orchestrator analyzes user role information and applies the appropriate guardrail configuration—either admin guardrails (with full access) or non-admin guardrails (with restricted access).
5. **Amazon Bedrock Knowledge Bases** processes the query with the appropriate guardrails, retrieving relevant documents from **OpenSearch Service** according to role-based metadata filters.

Reviewed for technical accuracy July 23, 2025

AWS Reference Architecture