



Securing and Scaling the Data Pipeline

AWS Academy Data Engineering

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

~ This module reinforces at a high level the security and resiliency best practices that are captured in the Well-Architected Framework, and highlights considerations for working with a data pipeline and an analytics or AI/ML workload. This module introduces the key AWS services and service features that provide security and scalability including IAM and CloudFormation. It focuses on course objectives 4. Identify the risks and approaches for securing and governing data at each step and each transition of the data pipeline and 5. Identify scaling considerations and best practices for building pipelines that handle large-scale datasets

~ Key references include the WAF lens papers and especially the security sections <https://docs.aws.amazon.com/wellarchitected/latest/analytics-lens/security.html> and the ML WAF paper - <https://docs.aws.amazon.com/wellarchitected/latest/machine-learning-lens/security-pillar-best-practices-2.html>

~ [4 - Classify and protect data](#)

~ [5 - Control the data access](#)

~ [6 – Control the access to workload infrastructure](#)

~ [MLSEC-03: Secure data and modeling environment](#)

~ [MLSEC-04: Protect sensitive data privacy](#)

~ [MLSEC-05: Enforce data lineage](#)

~ [MLSEC-06: Keep only relevant data](#)

| Slide number 1

| Instructor notes

|

| Student notes

Welcome to the Securing and Scaling the Data Pipeline module.



| Slide number 2

| Instructor notes

|

| Student notes

This introduction section describes the content of this module.

Module objectives

This module prepares you to do the following:

- Highlight how cloud security best practices apply to analytics and machine learning (ML) data pipelines.
- List AWS services that play key roles in securing a data pipeline.
- Describe how infrastructure as code (IaC) supports the security and scalability of a data pipeline infrastructure.
- Identify the function of common AWS CloudFormation template sections.



~~Script: Handle parentheses and bulleted list.

| **Slide number 3**

| **Instructor notes**

|

| **Student notes**

This module prepares you to do the following:

- Highlight how cloud security best practices apply to analytics and machine learning (ML) data pipelines.
- List AWS services that play key roles in securing a data pipeline.
- Cite factors that drive performance and scaling decisions across each layer of a data pipeline.
- Describe how infrastructure as code (IaC) supports the security and scalability of a data pipeline infrastructure.
- Identify the function of common AWS CloudFormation template sections.

Module overview

Presentation sections

- Cloud security review
- Security of analytics workloads
- ML security
- Scaling: An overview
- Creating a scalable infrastructure
- Creating scalable components

Knowledge checks

- Online knowledge check
- Sample exam question



~~Script: Remove references to demos and knowledge check.

| Slide number 4

| Instructor notes

| Each module has an introduction, content sections, and a wrap-up. The wrap-up for this module contains a sample exam question for you to review with the students.

|

| Student notes

The objectives of this module are presented across multiple sections.

The module also includes two demos that show <explain the purpose of the demos>.

The module wraps up with a sample exam question and an online knowledge check that covers the presented material.



| Slide number 5

| Instructor notes

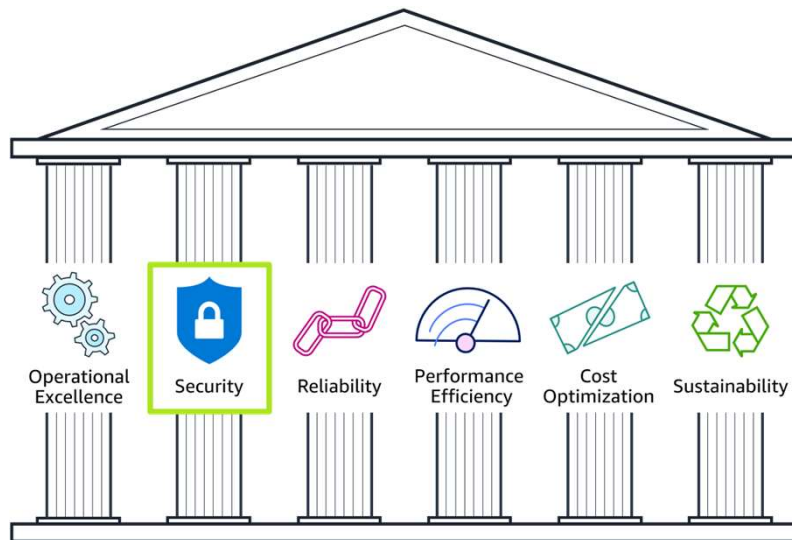
| This section of the module provides a review of security principles that are found in the security pillar of the AWS Well-Architected Framework.

|

| Student notes

This section reviews cloud security concepts that are highlighted in the AWS Well-Architected Framework.

AWS Well-Architected Framework: Security



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

6

| Slide number 6

| Instructor notes

| Suggestion: Show students where to find the security information for AWS Well-Architected Framework:

1. | Go to the main AWS Well-Architected Framework website, and go to the pillars section of the page to show the six pillars.
2. | Open the Framework Overview page.
3. | To show the security design principles, go to **The Pillars of the Framework > Security > Design Principles**.
4. | To show the questions and best practices of the security pillar, go to **Appendix: Questions and Best Practices > Security**.

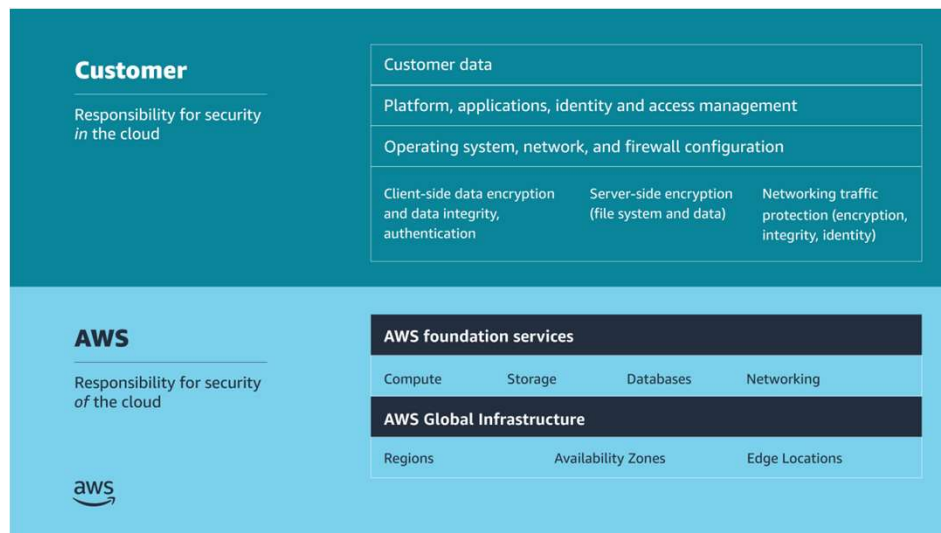
| Student notes

The AWS Academy Cloud Foundations course introduced the AWS Well-Architected Framework. The framework is available on the AWS website, and you can find the link on the Content Resources page in your online course. You might find it helpful to open the website now to look at the way that the pillars are organized.

The framework comprises six pillars: Operational Excellence, Security, Reliability,

Performance Efficiency, Cost Optimization, and Sustainability. The framework provides best practices and design guidance across each pillar to help you make choices and review existing architectures. This module focuses on using the best practices from the security pillar to secure your data analytics and ML workloads by using the corresponding AWS Well-Architected Lenses.

Shared responsibility model



~~Script: Handle parentheses.

| Slide number 7

| Instructor notes

| This slide re-introduces the shared responsibility model. Discuss which security responsibilities lie within each portion of the model to reinforce which resources and infrastructure the account holder must secure. Refer to this model when discussing the different elements of a pipeline, and have students analyze the different areas of responsibility that exist in securing the pipeline.

|

| Student notes

Security and compliance are shared responsibilities between AWS and customers. AWS operates, manages, and controls security *of* the cloud. This responsibility includes securing components—from the host operating system and virtualization layer down to the physical security of the facilities where the service operates. AWS is responsible for protecting the global infrastructure that runs all the services that are offered in the AWS Cloud. This infrastructure is composed of the hardware, software, networking, and facilities that run AWS Cloud services.

Customers assume responsibility and management *in* the cloud. The security steps that you must take depend on the services that you use and the complexity of your system.

Customers responsibilities include selecting and securing operating systems that run on Amazon Elastic Compute Cloud (Amazon EC2) instances, and securing the applications that are launched on AWS resources. Customers must also select and handle security group configurations, firewall configurations, network configurations, and secure account management. Customers are also responsible for managing their data, including encryption options.

To reiterate, AWS secures the hardware, software, facilities, and networks that run all AWS products and services. You are responsible for what you implement by using AWS products and services, and for the applications that you connect to AWS. The security steps that you must take depend on the services that you use and the complexity of your system.

Design principles for data security

- Implement a strong identity foundation.
- Enable traceability.
- Apply security at all layers.
- Automate security best practices.
- Protect data in transit and at rest.
- Keep people away from data.
- Prepare for security events.



| Slide number 8

| Instructor notes

| This slide introduces the AWS Well-Architected Framework design principles for the security pillar. You might use these design principles as a point of reference to reinforce the materials that are discussed in this module.

| Student notes

In the cloud, you can apply a number of principles to help you strengthen your data security:

- **Implement a strong identity foundation:** Implement the principle of least privilege and enforce separation of duties with appropriate authorization for each interaction with your AWS resources. Centralize identity management, and aim to eliminate reliance on long-term, static credentials.
- **Enable traceability:** Monitor, alert, and audit actions and changes to your environment in real time. Integrate log and metric collection with systems to automatically investigate and take action.
- **Apply security at all layers:** Apply a defense-in-depth approach with multiple security controls, and apply it to all layers (for example, edge of network, virtual private cloud [VPC], load balancing, every instance and compute service, operating system, application, and code).

- **Automate security best practices:** Automated software-based security mechanisms improve your ability to securely scale more rapidly and cost-effectively. Create secure architectures, which includes implementing controls that are defined and managed as code in version-controlled templates.
- **Protect data in transit and at rest:** Classify your data into sensitivity levels, and use mechanisms, such as encryption, tokenization, and access control, where appropriate.
- **Keep people away from data:** Use mechanisms and tools to reduce or eliminate the need for direct access or manual processing of data. This reduces the risk of mishandling or modification and human error when handling sensitive data.
- **Prepare for security events:** Prepare for an incident by having incident management and investigation policies and processes that align to your organizational requirements. Run incident response simulations, and use tools with automation to increase your speed for detection, investigation, and recovery.

Access management

Authentication

- Uses credentials to establish the identity of the requestor
- Grants or denies access to resources based on identity
- Utilizes usernames, passwords, and multi-factor authentication (MFA) among other methods

Authorization

- Takes place only after authentication
- Determines the level of access that an identity has to a resource
- Common methods include attribute-based access control (ABAC) and role-based access control (RBAC)

Principle of least privilege

- Grant only the permissions that are required to perform a task
- Start with a minimum set of permissions
- Grant additional permissions as necessary
- Revoke unnecessary permissions



| Slide number 9

| Instructor notes

| Slides 9 (Access management) and 10 (AWS Identity and Access Management [IAM]) review the concepts of managing access to AWS services and resources. Access management is a key concept that is reiterated throughout this module.

| Student notes

The security of your data pipeline relies on your control of who and what can access your services and resources and determining what they are able to do with that access. These control measures are commonly referred to as *authentication* and *authorization*, and both of these measures are guided by the *principle of least privilege*.

Authentication uses credentials to establish the identity of the requestor and determine whether they have permissions to access your resources. You can manage identities by using the AWS Identity and Access Management (IAM) service, which integrates with most AWS services, or through other centralized management services. After an entity is authenticated, policies and permissions will determine what actions they can perform on the requested resource. Best practices such as using groups and attribute-based access control (ABAC), role-based access control (RBAC), temporary credentials, and proper secrets storage (by using services such as AWS Secrets Manager) will further assist you to

implement fine-grained access control in your pipeline.

AWS Identity and Access Management (IAM)

- Helps you to securely share and control access to your AWS resources for individuals and groups
- Integrates with most AWS services
- Supports federated identity management
- Supports granular permissions
- Supports MFA
- Provides identity information for information assurance and compliance audits



| Slide number 10

| Instructor notes

|

| Student notes

IAM is a web service that helps you to securely control access to AWS resources. Use IAM to control who is *authenticated* (signed in) and *authorized* (has permissions) to use resources.

IAM is integrated into most AWS services, and you can implement IAM to effectively control access to the services that make up your data analytics or ML pipeline. You can define access controls from one place in the AWS Management Console, and they will take effect throughout your AWS environment.

You can use IAM to grant granular access to users, groups, and applications to the console and to AWS service APIs by using existing identity systems. AWS supports federation from corporate systems such as Microsoft Active Directory and standards-based identity providers. For example, you can use IAM policies to control access to each of the services that make up your ML pipeline, enforcing least privilege access to secure your pipeline from beginning to end.

IAM also supports *multi-factor authentication* (MFA). If MFA is activated and an IAM user attempts to log in, then the user is prompted for an authentication code. The authentication code is delivered to a specially configured AWS MFA hardware device or to a software-based device, such as Google's Authenticator application.

When used in conjunction with AWS CloudTrail, IAM can also support information assurance by providing log records that include the identity information of users that request resources in your account.

Data security

Data at rest

Any data that persists in nonvolatile storage for any duration

Protections

- Implement secure key management.
- Enforce encryption at rest.
- Enforce access control.
- Audit the use of encryption keys.
- Use mechanisms to keep people away from data.
- Automate data-at-rest protection.
- Audit data access logs.

Data in transit

Any data that is sent from one system to another

Protections

- Implement secure key and certificate management.
- Enforce encryption in transit.
- Authenticate network communications.
- Automate detection of unintended data access.
- Secure data from between VPC or on-premises locations.



| Slide number 11

| Instructor notes

| Discuss with students how data at rest and data in transit are protected in AWS. The topic of encryption on this slide leads into the next slide, which discusses the AWS Key Management Service (AWS KMS).

|

| Student notes

The term *data at rest* describes any data that you persist in nonvolatile storage for any duration in your workload. The term typically refers to data that is currently being stored and not accessed. *Data in transit* describes data that is being sent from one system or service to another—for example, a file in an Amazon Simple Storage Service (Amazon S3) bucket that is being accessed by an EC2 instance. Protect both forms of data in accordance with their assigned sensitivity levels by using protective mechanisms such as encryption and access control. Use protective measures that are commensurate with the compliance requirements that your organization must adhere to, which will generally be determined by factors such as location and industry.

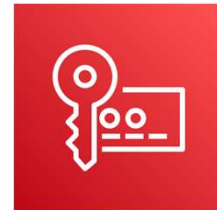
Protecting data at rest relies on encryption and tokenization. You can enforce encryption of data at rest in AWS by setting default encryption in services such as Amazon S3 and Amazon EC2. Access measures are also useful to protect data at rest and enable you to implement

least privilege access.

To protect data in transit, enforce encryption in transit. AWS services provide HTTPS endpoints that use TLS for communication to provide encryption in transit when communicating with the AWS API. You can further enforce encryption in transit by using security groups at the VPC level to block insecure protocols to ensure that any communications that originate outside of your pipeline are secure.

AWS Key Management Service (AWS KMS)

- Provides the ability to create and manage cryptographic keys
- Uses hardware security modules (HSMs) to protect your keys
- Is integrated with other AWS services
- Provides the ability to set usage policies to determine which users can use which keys



| Slide number 12

| Instructor notes

|

| Student notes

AWS Key Management Service (AWS KMS) is a managed service that provides the ability to create and control the keys that are used to encrypt your data. You can create data keys with unique aliases and descriptions for better management, automatically rotate your keys on a scheduled basis, and disable or delete keys so that no one can use them. You can also import your own keys instead of using AWS generated keys.

The service uses FIPS 140-2 validated hardware security modules (HSMs) to protect keys. AWS KMS is integrated with other AWS services to help you protect the data that you store with these services. Integrated AWS services use envelope encryption to help protect your encryption keys.

With AWS KMS, you can centrally manage and securely store your keys. You can use the keys within your applications and supported AWS Cloud services to protect your data, but the keys never leave AWS KMS. This reduces the risk of your data key being compromised. You submit data to AWS KMS to be encrypted or decrypted under keys that you control.

You can set usage policies on these keys to determine which users can use them. All requests to use these keys are logged in CloudTrail so that you can understand who used which key and when. CloudTrail logs all AWS KMS operations, including read-only operations, operations that manage AWS KMS keys, and cryptographic operations.

Logging and monitoring

Logging

- Logging is the collection and recording of activity and event data.
- The information logged varies based on the service.
- Common log elements include date and time of event, origin of event, and identity of resources that were accessed.

Monitoring

- Monitoring is the continuous verification of the security and performance of your resources, applications, and data.
- AWS provides several services that give you the visibility to spot issues before they impact operations.



| Slide number 13

| Instructor notes

| This slide discusses the basics of logging and monitoring. Slides 14 (AWS CloudTrail) and 15 (Amazon CloudWatch) will go into further detail about two key monitoring services that will be discussed throughout the module.

|

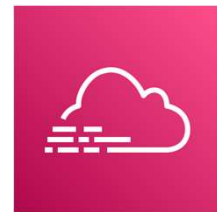
| Student notes

Logging is the collection and recording of activity and event data. A service itself can provide logging capabilities, as with Amazon Virtual Private Cloud (Amazon VPC) Flow Logs and Amazon S3 server access logs. Or a secondary service, such as CloudTrail, might provide logging. The type of information logged will vary based on the service that conducts the logging, but some common elements are logged, such as the date and time of an event, the originating location of request, and the identity of resources accessed.

Monitoring is the practice of continuously verifying the security and performance of your resources and data. AWS provides several services that you can use to monitor who accesses your resources as well as monitor the resources themselves. Amazon CloudWatch is one such service that you can use to monitor your AWS resources. CloudWatch collects monitoring and operational data to provide you with a unified view of your resources, both in the cloud and on premises.

AWS CloudTrail

- Is the primary AWS solution for logging
- Assists you to enable governance and compliance as well as operational and risk auditing of your AWS account
- Records actions taken by a user, role, or AWS service as events
- Can be used to view, search, download, archive, analyze, and respond to account activity across your AWS infrastructure



| Slide number 14

| Instructor notes

|

| Student notes

The CloudTrail service helps you enable governance and compliance, as well as operational and risk auditing, of your AWS account. Actions taken by a user, role, or AWS service are recorded as *events* in CloudTrail. Events include actions taken in the console, AWS Command Line Interface (AWS CLI), and AWS SDKs and APIs. You can view, search, download, archive, analyze, and respond to these recorded events in the CloudTrail console. These events can be used to audit and troubleshoot issues that might arise within your data pipeline and can alert you to possible inconsistencies and unauthorized access or actions.

You can integrate CloudTrail into applications by using the API, automate trail creation for your organization, check the status of trails you create, and control how users view CloudTrail events.

If the data that you use is sensitive, such as personal medical information that is used in an ML pipeline, CloudTrail can help you ensure that your organization maintains compliance with the applicable laws and regulations governing the data type.

Amazon CloudWatch

- Is a monitoring and observability service
- Provides a unified view of the operational health of your AWS resources, applications, and services
- Collects metrics in the AWS Cloud and on premises
- Can be used to monitor and troubleshoot infrastructure
- Provides the ability to customize logs and events



| Slide number 15

| Instructor notes

|

| Student notes

CloudWatch is a monitoring and observability service built for DevOps engineers, developers, site reliability engineers (SREs), IT managers, and product owners. CloudWatch collects monitoring and operational data as logs, metrics, and events. You can use CloudWatch to detect anomalous behavior in your environments, set alarms, visualize logs and metrics side by side, take automated actions, troubleshoot issues, and discover insights to keep your applications running smoothly. With CloudWatch, you can collect, access, and correlate information for a unified view of all of your AWS resources, applications, and services running on AWS and on premises. You can use CloudWatch Events to deliver a near real-time stream of system events that describe changes in AWS resources. CloudWatch Events becomes aware of operational changes as they occur and responds to them. CloudWatch Events takes corrective action as necessary by sending messages to respond to the environment, activating functions, making changes, and capturing state information. Note that CloudWatch Events is currently being replaced by Amazon EventBridge.

With CloudWatch Logs, you can monitor, store, and access your log files from EC2 instances, CloudTrail, Amazon Route 53, and other sources. You can centralize the logs from all of your

systems, applications, and AWS services in a single, highly scalable service. You can then easily view the logs, search them for specific error codes or patterns, filter them based on specific fields, or archive them securely for future analysis. With CloudWatch Logs, you can see all of your logs, regardless of their source, as a single and consistent flow of events ordered by time. You can query the logs and sort them based on other dimensions, group them by specific fields, create custom computations with a powerful query language, and visualize log data in dashboards.

By implementing CloudWatch, you can use AWS Auto Scaling for your EC2 instances at no additional charge.

Key takeaways: Cloud security review



- Access management consists of authentication and authorization; adhere to the principle of least privilege with both.
- IAM integrates with most AWS services and helps you to securely share and control individual and group access to your AWS resources.
- A key aspect of a data security plan is to secure data at rest and data in transit.
- Logging and monitoring can assist your organization to maintain compliance with local laws and regulations.

| Slide number 16

| Instructor notes

|

| Student notes

Here are a few key points to summarize this section.

- Access management consists of authentication and authorization; adhere to the principle of least privilege with both.
- IAM integrates with most AWS services and helps you to securely share and control individual and group access to your AWS resources.
- A key aspect of a data security plan is to secure data at rest and data in transit.
- Logging and monitoring can assist your organization to maintain compliance with local laws and regulations.

Security of analytics workloads

Securing and Scaling the Data Pipeline



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| Slide number 17

| Instructor notes

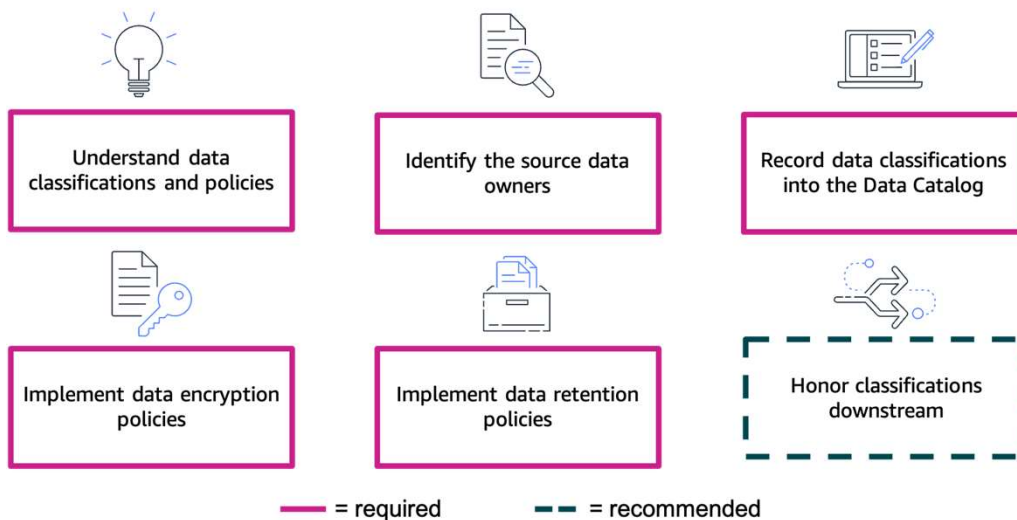
| This section discusses the security elements of the AWS Well-Architected Data Analytics Lens.

|

| Student notes

This section introduces best practices related to the security of analytics workloads.

Classify and protect data



| Slide number 18

| Instructor notes

| Slides 19 (Classify and protect data) through 21 (Control the access to workload infrastructure) discuss the best practices that are outlined in the AWS Well-Architected Framework Data Analytics Lens.

|

| Student notes

The first group of best practices falls under the category of *classify and protect data*. This category contains six best practices that focus on classifying and protecting data in analytics workloads. Of the six best practices, five are given a priority of *required*, and one has a priority of *recommended*.

As the analytics workload owner, honor the data classifications and protection policies that the data owners assigned to the source data that you're ingesting. Share these classifications and policies with the downstream data consumers. This sharing of classifications will allow for the data to be properly protected and retained based on organizational policies.

Understand data classifications and their protection policies.

To properly classify data, you first need to understand the data classifications and the

affiliated protection policies of your organization. The levels and their titles will vary based on industry and organizational compliance requirements, so ensure that you're using the correct classifications for your organization. If needed, use the Data Classification: Secure Cloud Adoption whitepaper to assist you to identify the possible classification levels that you are working with. After you determine the possible classifications, allow the data owners to classify the data and define the access rules based on the sensitivity and criticality of the data. Next, identify security zone models to isolate data based on classification by designing security zone models, beginning at the AWS account level and extending down to the AWS resource level. Finally, identify sensitive information and define your protection policies. Amazon Macie can help you identify sensitive data by using custom data identifiers. Review the sensitivity and criticality level of the data, and implement the appropriate data protection policies.

Identify the source data owners and have them set the data classifications.

Identify the owners of the source data to confirm the classification of their data to ensure that the appropriate protections are applied within the analytics workload. As the data moves through the analytics workflow, these classifications will ensure that the data is appropriately protected from unauthorized access by personnel or systems. Assign ownership of datasets into the *Data Catalog*, a collection of metadata that helps centralize share, search, and manage permissions. Knowing the identity of the dataset owners can help you to more efficiently troubleshoot any issues with the dataset.

Record data classifications into the Data Catalog so that analytics workloads can understand.

The Data Catalog is only useful if it is up to date. One approach to ensure that the catalog is up to date is by allowing processes to update it. An up-to-date Data Catalog will provide an accurate and reliable record of data locations and classifications, which allows for more effective protection of potentially sensitive data. Two methods are recommended to keep an up-to-date Data Catalog: 1. Use tags to indicate the data classifications and 2. Record the lineage of data to track changes in the Data Catalog.

Using tags to indicate data classifications will allow for the discoverability of data sensitivity without exposing the actual data. Data *lineage*, which is a relation among data and the processing systems, should be discoverable, recordable, and visualizable from source to downstream systems to understand the data's source, what data derived from it, and which systems are reading it downstream.

Implement encryption policies for each class of data in the analytics workload.

Encrypt workload data in accordance with the policies that govern the classification level. Implement encryption policies for both data at rest and data in transit. You can find AWS service-specific implementation guides on the AWS Documentation website. AWS KMS provides you with a simple and robust mechanism to manage your encryption keys.

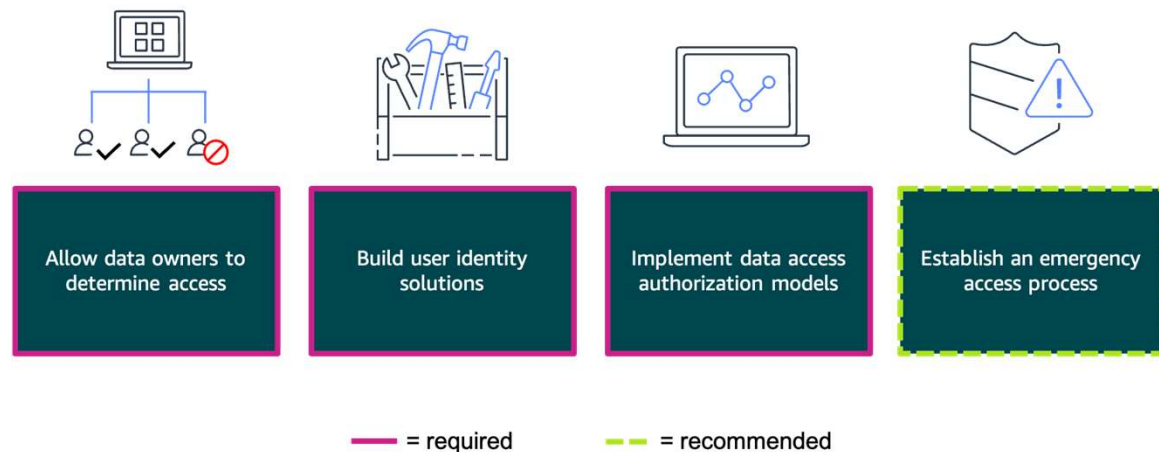
Implement data retention policies for each class of data in the analytics workload.

While your organization might have a standard data retention policy, you should also implement classification-based retention policies. Create backup requirements and policies based on data classifications. Ensure that you align them with the appropriate recovery point objective (RPO), recovery time objective (RTO), data classification, and any compliance and audit requirements that your organization must adhere to. Create data retention requirement policies based on the data classifications, and tailor your policies to individual assets based on retention requirements—avoid blanket retention policies. Use analytics systems to implement rules that retain in-scoped data that aligns with your organization's retention policies.

Require downstream systems to honor the classifications.

Remember that other data-consuming systems will access the data that your analytics workload shares. Because of this, you should require those downstream systems to implement the appropriate data classification policies. Have a centralized, shareable catalog with cross-account access to ensure that data owners manage permissions for downstream systems. Having appropriately configured cross-account access will allow downstream systems to use the data from your analytics workload. Another recommendation is to monitor the downstream systems' eligibility to access classified data from the analytics workload. While cross-account access will allow downstream systems to use your analytics data, you want to ensure that those systems are *eligible* to access that data. Periodically verify that any downstream systems are eligible to process data at the appropriate level of classification.

Control the data access



| Slide number 19

| Instructor notes

|

| Student notes

The next group of best practices is categorized under the title of *control the data access*. As the owner of the analytics workload, you should honor the access management policies of the source systems, secure access to the data in the analytics workload, and ensure that any downstream sharing of data is done in compliance with the source system's classification policies.

Allow data owners to determine which people or systems can access data in analytics and downstream workloads.

In general, an analytics workload will consolidate multiple datasets into a central location. Each of those consolidated datasets will likely be owned by different teams or individuals, so it's important to identify which person or group owns which dataset in order to coordinate data access permissions and ensure those permissions adhere to the principle of least privilege. Another best practice is to use an Access Control Matrix to document which users, roles, or systems have access to which datasets and what actions each user, role, or system can perform.

Build user identity solutions that uniquely identify people and systems.

To federate with IAM by using AWS IAM Identity Center (successor to AWS Single Sign-On) or another identity provider, a best practice is to centralize your workforce identities. For example, you can map IAM roles to AWS Glue Data Catalog resource policies to effectively manage access.

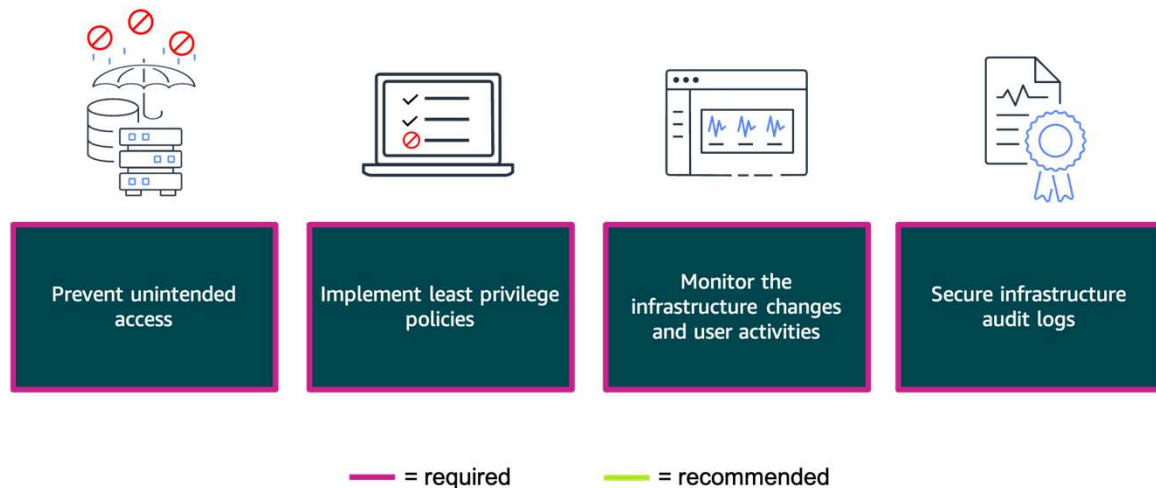
Implement the required data access authorization models.

As much as possible, limit access to data stores that contain sensitive data. One way that you can effectively manage this access is by using control mechanisms such as role-based access control (RBAC). Another best practice is to implement dataset-level access controls by building analytics workloads to have the dataset owners control the data access per dataset level. Finally, implement column-level data access controls and enforce the masking of sensitive data through the use of column-level restrictions. This practice will help ensure that end users of analytics applications are not inadvertently exposed to sensitive data.

Establish an emergency access process to the source, analytics, and downstream systems.

Though unlikely, automated processes and pipelines can experience occasional issues. You can remediate such issues more efficiently by expediting access through the use of emergency access. Ensure that risk analysis is done on your analytics workload by identifying emergency situations and a procedure to allow such emergency access. Conducting such risk analysis can help you identify potential threats from source systems, analytics workloads, and downstream systems. As you identify risks, prioritize each event based on likelihood and overall business impact level. Discuss these findings with your source and downstream system owners, and determine how you could continue to allow analytics workloads to access those systems.

Control the access to workload infrastructure



| Slide number 20

| Instructor notes

|

| Student notes

The final group of data analytics best practices are categorized as *control the access to workload infrastructure*. Through the use of services such as IAM, you can ensure that your environment is accessible while adhering to the principle of least privilege.

Prevent unintended access to the infrastructure.

Prevent unintended or unauthorized access to your infrastructure by granting the least privilege possible. IAM provides the AWS Identity and Access Management Access Analyzer tool for all AWS accounts that are centrally managed through AWS Organizations. IAM Access Analyzer aids you to set and verify permissions, and assists you to refine those permissions by removing unused access. Infrastructure boundaries are yet another way that you can prevent unauthorized access. You can create network boundaries by using VPC private subnets to isolate your analytics resources and allowing only the minimal amount of connections that are required to support your analytics workload.

Implement least privilege policies for source and downstream systems.

Identify your source and downstream users and systems, and then identify the minimum

privileges that each would require and implement only those permissions necessary. For example, if a downstream Amazon QuickSight service needs access to your Amazon OpenSearch Service data, ensure that the QuickSight users have the minimal amount of access that is required. You can further secure access by implementing the two-person rule. This practice can help you prevent unauthorized access, altering, or even deletion of critical or classified information by requiring the presence of two separate users to perform tasks that you deem critical.

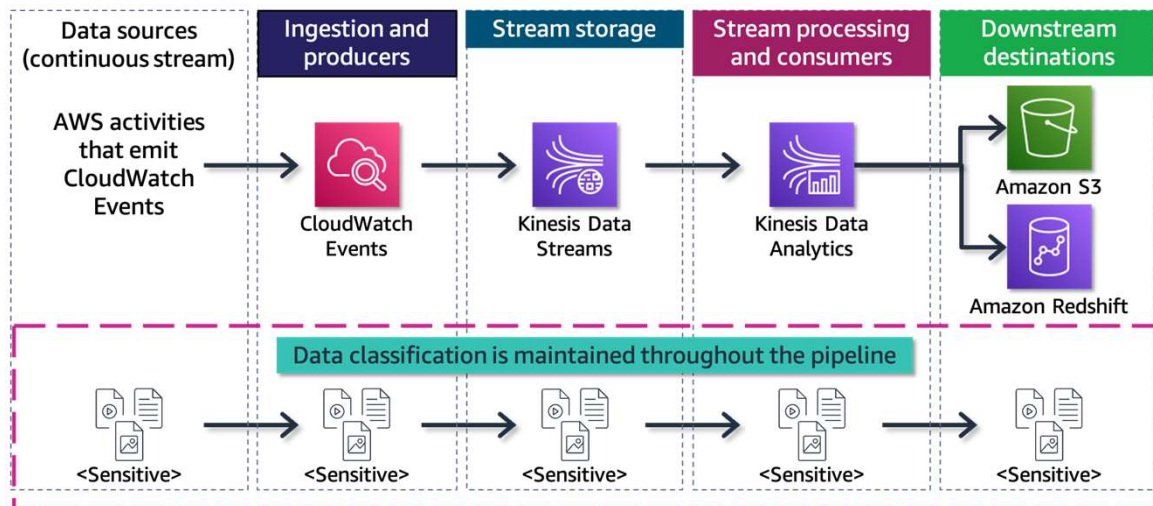
Monitor the infrastructure changes and the user activities against the infrastructure.

Monitor infrastructure changes closely. Practices such as implementing a change control board or change advisory board can be helpful to prevent potentially disruptive or harmful changes. AWS services such as AWS Config, Amazon Inspector, and Amazon GuardDuty can assist you to monitor and log those changes. For security and accounting purposes, you should also monitor and log user activities against your infrastructure. CloudTrail can provide you with reviewable audit logs of user actions against your resources and can be paired with CloudWatch for near real-time monitoring.

Secure the audit logs that record every data or resource access in analytics infrastructure.

While services such as AWS Config and CloudTrail can provide you with a wealth of audit logs, these records are useless unless they are also securely handled and maintained. Use fault-tolerant storage for your logs, and restrict access to the logs to privileged users. Regularly monitor, log, and audit access to audit logs as well. Review the audit log features of your analytics systems, and ensure that auditing is active and delivered to the fault-tolerant storage that you have selected.

Securing the stream processing pipeline



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

21

| Slide number 21

| Instructor notes

| Use the operational analytics security scenario to tie together the best practices discussed in this section. This slide discusses how the best practices from the preceding slides could be applied to a stream processing pipeline.

|

| Student notes

Let's look at how to apply some of the security best practices from the AWS Well-Architected Data Analytics Lens to a typical stream processing pipeline.

- **Understand data classifications and their protection policies:** This best practice will govern how you handle all of the data that flows through your data pipeline. Review your organization's policies for classifying sensitive data, and know what steps you will need to take to ensure adherence to those policies.
- **Identify the source data owners and have them set the data classifications:** The source data types that are listed in the operational data section of your pipeline will likely have various teams or individuals as owners. Identify who the dataset owners are, and request that they properly classify their datasets if they haven't already done so.
- **Record data classifications into the Data Catalog so that analytics workloads can understand:** Keep your Data Catalog up to date so that you have accurate and reliable records of data locations and classifications.

- **Implement encryption policies for each class of data in the analytics workload:** After you identify the source data that you will work with and have established the classification level of each dataset, you will need to implement the applicable encryption policies. For data at rest, use one of the multiple encryption options available in Amazon S3. Secure your Amazon Kinesis data streams with server-side encryption by using AWS KMS.
- **Implement data retention policies for each class of data in the analytics workload:** Use classification-based retention policies for your datasets. Back up and retain analytics datasets based on your organization's policies for classified data.
- **Require downstream systems to honor the classifications:** Ensure that downstream services (such as Amazon Redshift), storage services, and downstream workloads honor your data classifications. If confidential source data enters your pipeline, handle the output of your pipeline as confidential data.
- **Allow data owners to determine which people or systems can access data in analytics and downstream workloads:** This applies to the owners of the operational data shown in the diagram as well as any downstream workloads that could be using the data that your workload produces.
- **Build user identity solutions that uniquely identify people and systems:** Implement IAM or another centralized identity management solution to control which users, roles, or services can access your resources. Every service shown in the diagram integrates with IAM and should be correctly configured to do so.
- **Implement the required data access authorization models:** The operational data in this diagram is classified as *sensitive*, which means that appropriate control measures must be put in place to adequately secure the data.
- **Establish an emergency access process to the source, analytics, and downstream systems:** Each service and resource in the pipeline should have access controls implemented. Ensure that you implement emergency access capabilities that would enable expedited access to your workload in the event of an issue with your pipeline.
- **Prevent unintended access to the infrastructure:** Implement IAM to control user, role, and system access to the data and services within your workload. Isolate your network as much as possible while maintaining only the network connections that are necessary to support your analytics workload.
- **Implement least privilege policies for source and downstream systems:** After you have identified your source and downstream users and systems, identify the minimum privileges that each would require and implement only the necessary permissions. For example, if the downstream Amazon RedShift service needs access to your Amazon Kinesis Data Analytics service data, ensure that the Amazon Redshift users have the minimal amount of access that is required.
- **Monitor the infrastructure changes and the user activities against the infrastructure:** After you secure access to your operational analytics pipeline, it's vital that you monitor it for infrastructure changes and malicious activity. Use additional AWS services, such as AWS Config and GuardDuty, to monitor your infrastructure. Use CloudTrail in conjunction with IAM to log user activities against your resources.
- **Secure the audit logs that record every data or resource access in analytics**

infrastructure: Finally, secure the audit logs for all of the services in your operational analytics pipeline and the services that support it. Ensure that you maintain adequate log security over a fault-tolerant storage solution, such as Amazon S3.

Key takeaways: Security of analytics workloads



- Honor the data classifications and protection policies that the owners of the source data assigned.
- Secure access to the data in the analytics workload.
- Share data downstream in compliance with the source system's classification policies.
- Ensure the environment is accessible with the least permissions necessary; automate auditing of environment changes, and alert in case of abnormal environment access.

| Slide number 22

| Instructor notes

|

| Student notes

Here are a few key points to summarize this section.

- Honor the data classifications and protection policies that the owners of the source data assigned.
- Secure access to the data in the analytics workload.
- Share data downstream in compliance with the source system's classification policies.
- Ensure the environment is accessible with the least permissions necessary; automate auditing of environment changes, and alert in case of abnormal environment access.

ML security

Securing and Scaling the Data Pipeline



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| Slide number 23

| Instructor notes

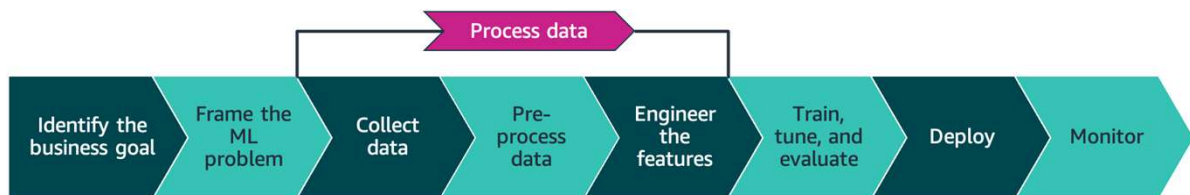
| This section of the module focuses on the security best practices from the AWS Well-Architected Machine Learning Lens.

|

| Student notes

This section introduces best practices related to ML security.

AWS Well-Architected Framework: ML Lens



ML lifecycle



| Slide number 24

| Instructor notes

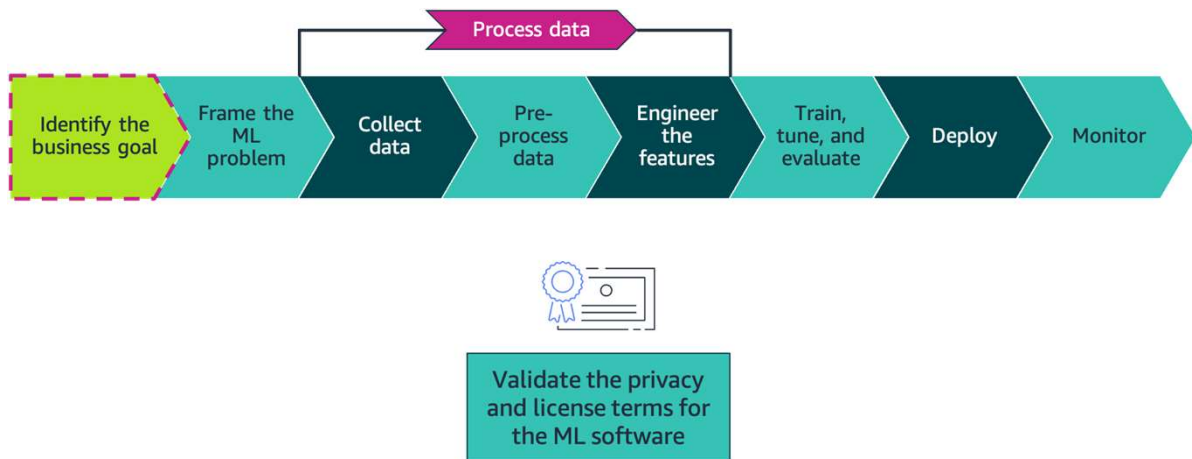
| The student notes point to six phases of the ML lifecycle, but the next several slides display what appears to be eight phases. The *collect data*, *pre-process data*, and *engineer the features* steps are grouped together as a single element that is referred to as the *process data* phase.

|

| Student notes

The ML lifecycle consists of six phases, which are portrayed here in a linear fashion. Note that *collect data*, *pre-process data*, and *engineer the features* are grouped together in the *process data* phase. The AWS Well-Architected Machine Learning Lens provides best practices from all five pillars, but this module focuses on the applicable security best practices.

Phase: Identify the business goal



| Slide number 25

| Instructor notes

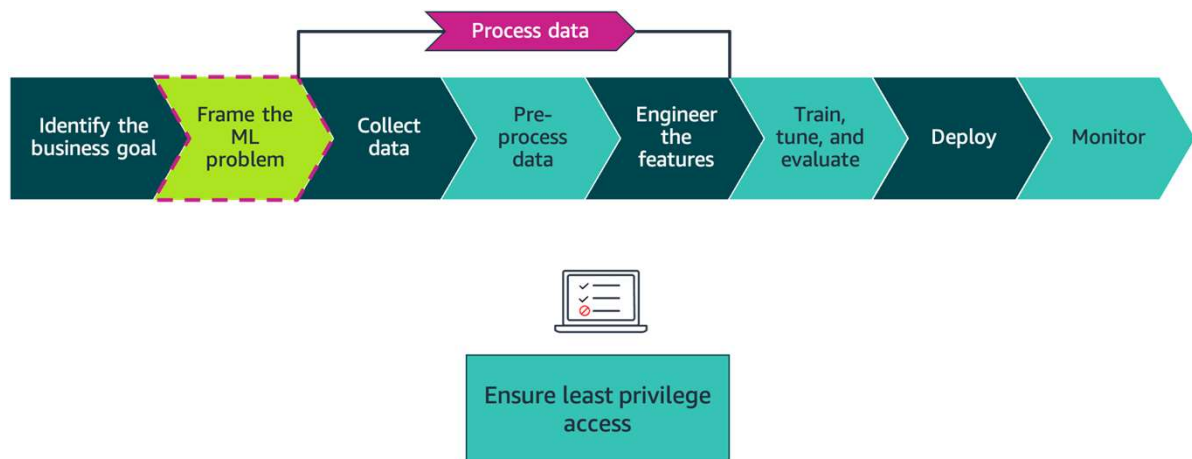
| Slides 26 (Phase: Identify the business goal) through 31 (Phase: Monitor) discuss the security best practices that are outlined in the AWS Well-Architected Machine Learning Lens.

|

| Student notes

Begin by identifying your business goal. ML libraries and packages handle all of the data processing, model development, training, and hosting. You will need to review the privacy and license agreements for all of the software and ML libraries that your ML lifecycle depends on to ensure that they comply with your organization's legal, privacy, and security terms and conditions. Review these items regularly, and establish a process to do so. You can control much of the software through the use of private repositories, or repos.

Phase: Frame the ML problem



| Slide number 26

| Instructor notes

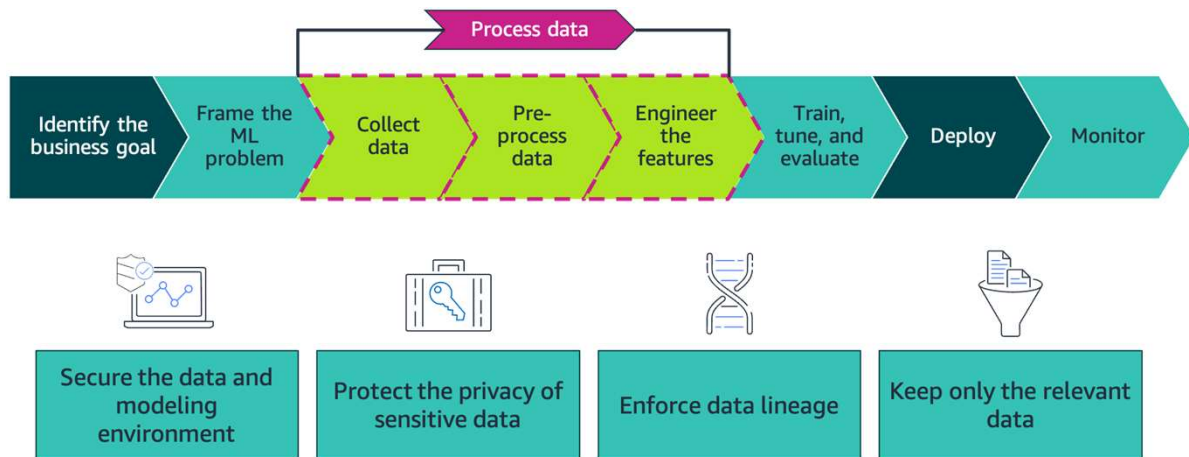
|

| Student notes

Apply the principle of least privilege throughout the ML lifecycle. Provide your project with dedicated network environments that have dedicated resources and services. This will assist you to achieve least privilege access to data, assets, and services.

Restrict access to data based on individual roles. Identify which roles will need data access to build models, features, and algorithms, and map those roles to access patterns by using role-based authentication.

Phase: Process data



| Slide number 27

| Instructor notes

|

| Student notes

Secure any system or environment that hosts data or enables model development. Store training data on secured storage and repositories. Run data preparation model development in a secure cloud. Tightly control access to the destination compute instances as data moves from the data repositories to the instances. Encrypt data in transit to—and at rest on—the compute and storage infrastructure.

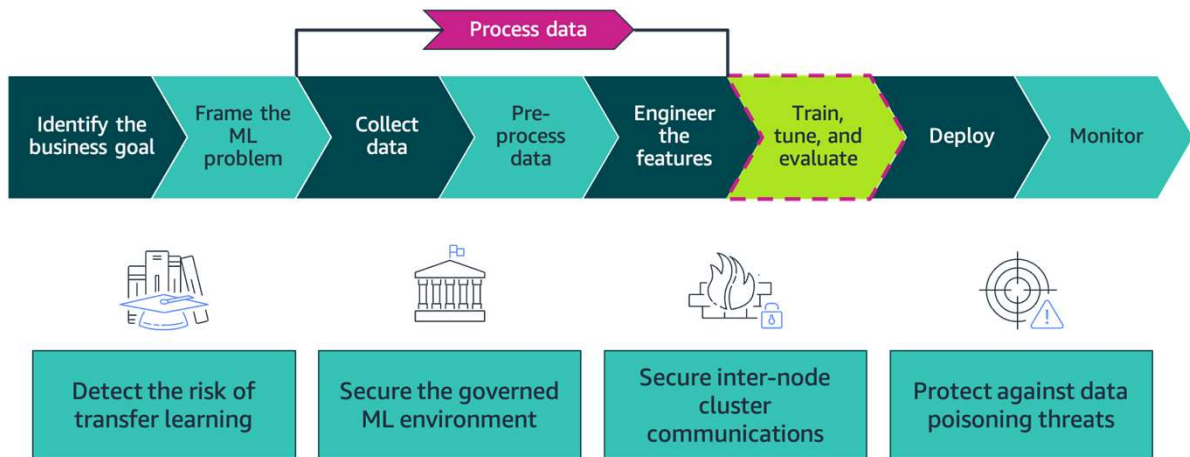
Protect sensitive training data against unintended disclosure. Identify and classify the sensitive data. Handle the sensitive data by using strategies such as removing, masking, tokenizing, and principal component analysis (PCA). Document best governance practices for future reuse and reference.

Monitor and track data origins and transformations over time. Strictly control who can access the data and what they can do with it. Perform preventative controls, auditing, and monitoring to demonstrate how data has been controlled during its lifetime. Implement integrity checks against training data to detect any unexpected deviances caused by loss, corruption, or manipulation. Data lineage enables visibility and helps you trace data

processing errors back to the root cause.

Preserve data across computing environments (such as development and staging), and only store the data that has business need to reduce data exposure risks. Implement mechanisms to enforce a lifecycle management process across the data. Decide when to remove data automatically based on age.

Phase: Train, tune, and evaluate



| Slide number 28

| Instructor notes

|

| Student notes

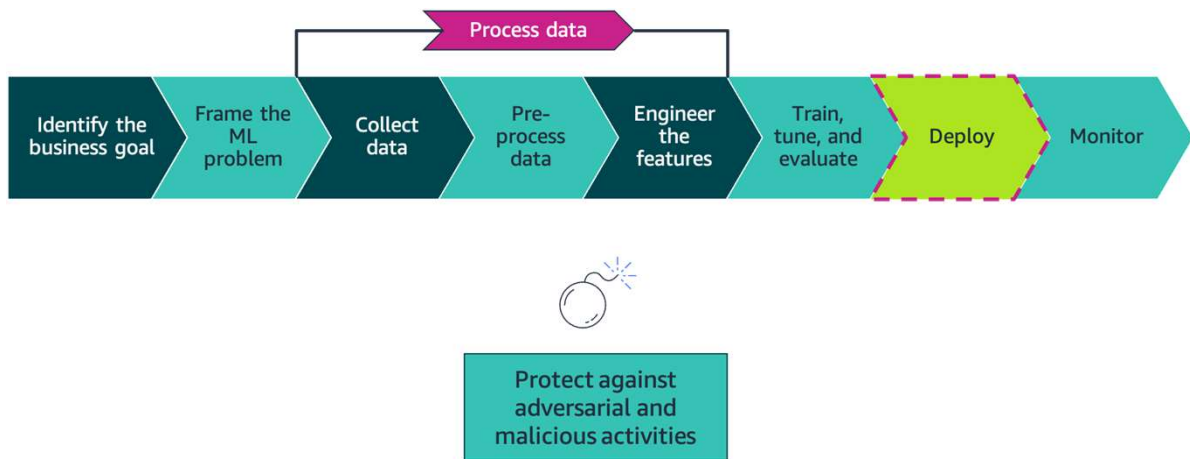
Monitor and ensure that the inherited prediction weights from a transferred model lead to the correct results. This helps minimize the risk of weak learning and incorrect outputs by using pretrained models. *Transfer learning* is an ML technique where a model that was pretrained on one task is fine-tuned on a new task. When using the transfer learning approach, use Amazon SageMaker Debugger to detect hidden problems that might have serious consequences. Examine model predictions to see what mistakes were made. Validate the robustness of your model, and consider how much of this robustness is from the inherited capabilities. Validate input and preprocesses to the model for realistic expectations.

Protect ML operations environments by using managed services with best practices, including detective and preventive guardrails, monitoring, security, and incident management. Explore data in a managed and secure notebook environment. Centrally manage the configuration of Jupyter environments, and enable self-service provisioning for the users.

For frameworks such as TensorFlow, it's common to share information such as coefficients as part of the internode cluster communications. The algorithms require that exchanged information stay synchronized across nodes. Secure this information through encryption in transit. Enable internode encryption in SageMaker, and enable encryption in transit in Amazon EMR.

Protect against data injection and data manipulation that pollutes the training dataset. Data injections add corrupt training data that will result in incorrect models and outputs. Data manipulations change existing data (for example, labels), which can result in inaccurate and weak predictive models. Identify and address corrupt data and inaccurate models by using security methods and anomaly detection algorithms.

Phase: Deploy



| Slide number 29

| Instructor notes

|

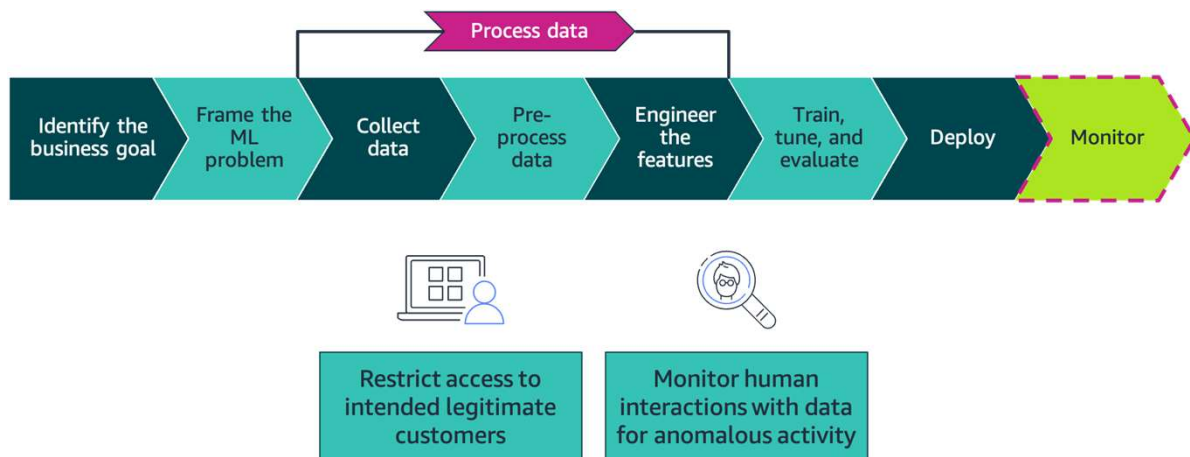
| Student notes

Add protection inside and outside of the deployed code to detect malicious inputs that might result in incorrect predictions. Automatically detect unauthorized changes by examining the inputs in detail. Repair and validate the inputs before they are added back to the pool. Evaluate your use case and determine bad predictions or classifications. Evaluate the robustness of the algorithm against increasingly perturbed inputs to understand susceptibility to manipulated inputs. Select diverse features to improve the algorithm's ability to handle outliers.

Consider pairing models in groups for increased diversity in decisions. Consider using ensembles to build in robustness around decision points. Detect similar repeated inputs to the model to indicate possible threats to the decision boundaries. This can take the form of model brute forcing, where threats iterate only a limited set of variables to determine what influences decision points and derive feature importance. If retraining on untrusted or invalidated inputs, make sure that any model skew is traced back to the data and pruned before retraining a replacement model. Host the model so that a consumer of the model can perform inference against it securely. Enable consumers using the API to define the

relationship, restrict access to the base model, and provide monitoring of model interactions.

Phase: Monitor



| Slide number 30

| Instructor notes

|

| Student notes

Use least-privileged permissions to invoke the deployed model endpoint. For consumers who are external to the workload environment, provide access through a secure API. Only authorized parties should make inferences against the ML model. Treat inference endpoints as you would any other HTTPS API. Ensure that you follow guidance from the AWS Well-Architected Framework to provide network controls, such as restricting access to specific IP ranges, and bot control. The HTTPS requests for these API calls should be signed so that the requester identity can be verified and the requested data is protected in transit.

Ensure that data access logging is enabled. Audit for anomalous data access events, such as access events from abnormal locations, or activity exceeding the baseline for that entity. Use services and tools that support anomalous activity alerting, and combine their use with data classification to assess risk. Evaluate services that can be used to aid in monitoring data access events. Verify that you have data access logging for all human create, read, update, and delete (CRUD) operations, including the details of who accessed what elements, what action they took, and at what time.

Use Amazon Macie to protect and classify training and inference data in Amazon S3. Macie is a fully managed security service that uses ML to automatically discover, classify, and protect sensitive data in AWS. The service recognizes sensitive data, such as personally identifiable information (PII) or intellectual property. Macie provides visibility into how this data is being accessed or moved. Macie also continually monitors data access activity for anomalies. The service generates detailed alerts when it detects a risk of unauthorized access or inadvertent data leaks.

Use Amazon GuardDuty to monitor for malicious and unauthorized activities. This will enable you to protect AWS accounts, workloads, and data stored in Amazon S3.

Key takeaways: ML security



- Apply the principle of least privilege throughout the ML lifecycle.
- Encrypt data in transit to and at rest in the compute and storage infrastructure.
- To reduce data exposure risks, only store the data that has business need.
- To detect malicious inputs that might result in incorrect predictions, add protection inside and outside of the deployed code.
- Ensure that data access logging is enabled, and audit for anomalous data access events.

~dev notes: [Fatemeh] I had to read the third bullet again. 'has business need' sounded awkward.

| Slide number 31

| Instructor notes

|

| Student notes

Here are a few key points to summarize this section.

- Apply the principle of least privilege throughout the ML lifecycle.
- Encrypt data in transit to and at rest in the compute and storage infrastructure.
- To reduce data exposure risks, only store the data that has business need.
- To detect malicious inputs that might result in incorrect predictions, add protection inside and outside of the deployed code.
- Ensure that data access logging is enabled, and audit for anomalous data access events.

Scaling: An overview

Securing and Scaling the Data Pipeline



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| Slide number 32

| Instructor notes

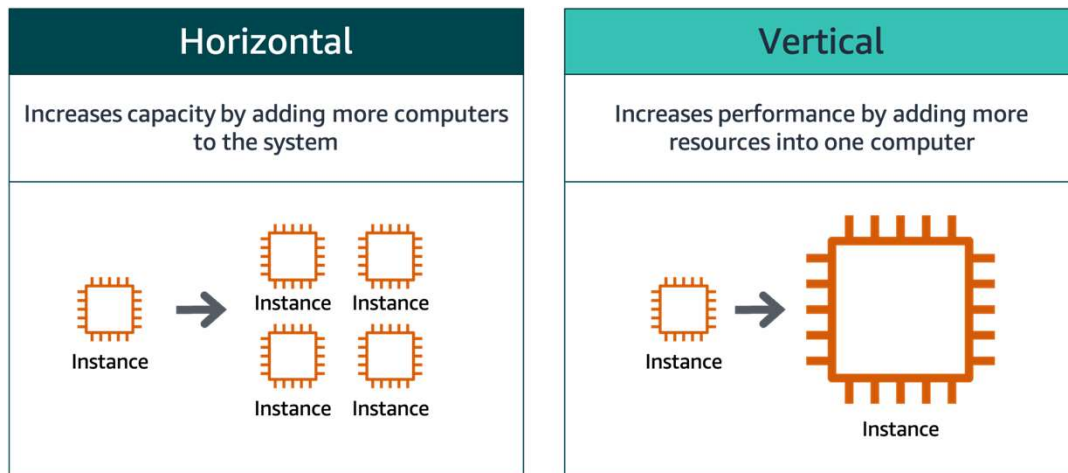
| This section discusses the concept of scaling and how it is applied to pipelines.

|

| Student notes

This section provides an overview of the concept of scaling.

Types of scaling



| Slide number 33

| Instructor notes

| Ensure that students understand the differences between horizontal and vertical scaling. Discuss the different methods that are used to accomplish each type of scaling.

|

| Student notes

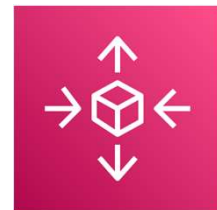
Scalability is a measurement of a system's ability to grow to accommodate an increase in demand. There are two types of scaling: horizontal scaling and vertical scaling.

Horizontal scaling meets the needs of increased demand by adding additional computers—in this case, EC2 instances. An example of horizontal scaling would be adding additional EC2 instances to a high-traffic resource pool to alleviate traffic congestion. You can use horizontal scaling to provide resilience and fault tolerance, but this type of scaling might require you to rework service implementations.

Vertical scaling meets the needs of increased demand by adding additional resources to a computer. Adding additional memory or processing power to an existing EC2 instance can improve analytics workload and ML processing times, but it doesn't improve resilience and fault tolerance.

AWS Auto Scaling

- Automatically adjusts capacity
- Set target resource utilization levels
- Create scaling plans
- Continuous performance monitoring
- Cost efficient
- Predictive scaling



| Slide number 34

| Instructor notes

|

| Student notes

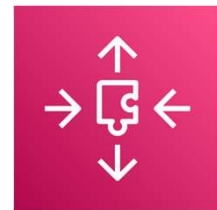
AWS Auto Scaling is a service that will automatically adjust your workload capacity, which gives you the ability to optimize both performance and costs. You can implement AWS Auto Scaling for a number of services, such as Amazon EC2, Amazon Elastic Container Service (Amazon ECS), Amazon DynamoDB, and Amazon Aurora. Scaling plans are available for you to configure automatic scaling for related or associated scalable resources. If you have workloads that have regularly occurring traffic spikes, predictive scaling can use the power of ML to automatically scale your resources in anticipation of your needs.

Because AWS Auto Scaling works in conjunction with CloudWatch, you can monitor metrics for your Auto Scaling groups and instances. AWS Auto Scaling publishes metrics about your Auto Scaling groups to CloudWatch at a 1-minute interval, which allows you to view your metrics in near real time.

Note that AWS Auto Scaling only accomplishes horizontal scaling. To automatically scale vertically, use the Ops Automator solution. Ops Automator works in conjunction with CloudWatch and AWS Lambda to provide vertical automatic scaling.

AWS Application Auto Scaling

- Automatically scale scalable services and resources
- Conditions-based
- Target tracking scaling
- Step scaling
- Scheduled scaling



| Slide number 35

| Instructor notes

|

| Student notes

Similar to AWS Auto Scaling, AWS Application Auto Scaling provides you with the ability to automatically adjust your workload capacity, which gives you the ability to optimize both performance and costs. The difference is that with Application Auto Scaling you can configure automatic scaling for resources such as Amazon EMR clusters, Lambda functions, and SageMaker endpoint variants. What exactly does this mean for you? It means that you can use many of the familiar automatic scaling features from Amazon EC2 on your data analytics and ML workloads.

With Application Auto Scaling, you can set conditions to automatically scale your scalable resources. You can scale a resource based on a target value for a specific CloudWatch metric by using *target tracking scaling*. You can scale a resource based on a set of scaling adjustments that vary based on the size of the alarm breach by using *step scaling*. Finally, by using *scheduled scaling*, you can scale a resource one time only or on a recurring schedule.

Scaling and cost management

- Identify the key performance goals for your workload.
- Design your workload to allow for components to scale independently
- Use metrics to find and address bottlenecks and issues.
- Make trade-offs that align to your key performance goals.



| Slide number 38

| Instructor notes

|

| Student notes

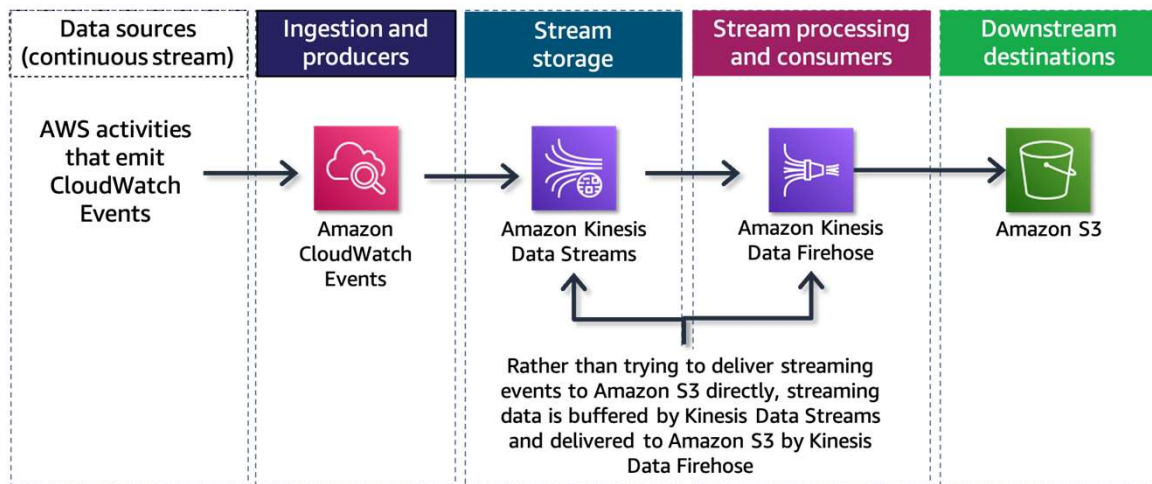
When you look at applying scaling to your pipeline, you need to start by thinking about the types of performance goals you should consider for each pipeline and each layer in the pipeline. You should also consider what you need to monitor to determine if you are meeting your performance requirements and that your pipeline is handling the scale of the data you are ingesting and processing.

For example, your primary goal might be to complete processing within a certain time frame of data arriving, or by a deadline. Or you might be more focused on making sure that the resource usage of the pipeline is within a specified budget, so you might focus on resource selection that reduces costs of processing. When you find steps that are failing, you need to determine if the cause of failure is related to scaling. For example, is a resource getting overwhelmed with requests, or is a process timing out waiting for input from a prior step?

Knowing how much data, how often it is updated, and how many different types of data you are going to process is important in considering how to scale your solution. You also

need to understand the limitation and scaling options for the services you are using, and design decoupled components wherever possible. For example, for many AWS services there are options that function like the AWS Auto Scaling feature just described. For example, Amazon Kinesis Data Streams can automatically add or reduce capacity to a stream based on traffic. Some services provide throttling options so that you can limit how much traffic is passed downstream. You can also use services like Amazon SNS, Amazon SQS or Amazon Kinesis Data Streams to buffer messaging between two other parts of the system.

Example: Scaling in a stream processing pipeline



| Slide number 38

| Instructor notes

Let's look at a variation of the stream processing pipeline example from earlier in this module and focus on how scaling is addressed. In this example, the CloudWatch events data is destined for storage in Amazon S3. But you wouldn't want to try to directly write this very high volume of streaming data to Amazon S3. Instead, you can configure an Amazon Kinesis Data Stream stream which provides the temporary buffer for the data. A Kinesis Data Firehose Delivery stream reads data off of the stream in batches and delivers it to Amazon S3. If the number of events increases, Kinesis Data Streams can automatically scale processing up without impacting the downstream systems. However, there is a cost to increasing the capacity on the stream.

For each layer and each set of services in your pipeline, you will need to make decisions and trade-offs about handling the scale of data passing through your pipeline.

Key takeaways: Scaling: An overview



- Horizontal scaling adds additional instances; vertical scaling adds additional resources to an instance.
- AWS Auto Scaling automatically scales Amazon EC2 capacity horizontally or vertically based on scaling plans and predictive scaling.
- Application Auto Scaling automatically scales resources for individual AWS services beyond Amazon EC2.

| Slide number 38

| Instructor notes

|

| Student notes

Here are a few key points to summarize this section.

- Horizontal scaling adds additional instances; vertical scaling adds additional resources to an instance.
- AWS Auto Scaling automatically scales Amazon EC2 capacity horizontally or vertically based on scaling plans and predictive scaling.
- Application Auto Scaling automatically scales resources for individual AWS services beyond Amazon EC2.

Creating a scalable infrastructure

Securing and Scaling the Data Pipeline



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| **Slide number 39**

| **Instructor notes**

|

| **Student notes**

This section introduces concepts related to creating a scalable infrastructure.

Issues with traditional infrastructure deployments

- Is time-consuming
- Has tasks that can be repetitive and prone to error
- Is a combination of scripts and manual processes
- Inadequate version control of scripts and processes
- Can lead to nonrepeatable, unreliable, or inconsistent deployments



| Slide number 40

| Instructor notes

| Discuss with students the hazards of traditional infrastructure deployments. Use the issues noted here to discuss the benefits of infrastructure as code (IaC) on the next slide. Refer to the stream processing pipeline example from earlier in the module, and discuss the process to set up the individual services manually.

|

| Student notes

Infrastructure was traditionally provisioned using a combination of scripts and manual processes. Sometimes these scripts were stored in version control systems or documented step by step in text files or runbooks. Often, the person writing the runbooks is not the same person running these scripts or following through the runbooks. If these scripts or runbooks are not updated frequently, they can potentially become showstoppers in deployments. This results in the creation of new environments not always being repeatable, reliable, or consistent.

Infrastructure as code

- Automates the process to create, update, and delete AWS infrastructure
- Stands up identical dev/test environments on demand
- Uses the same code to create your production environment that you used to create your other environments
- Provides the ability to create, deploy, and maintain infrastructure in a programmatic, descriptive, and declarative manner
- Benefits from the rigor, clarity, and reliability provided by CloudFormation and the AWS Cloud Development Kit (AWS CDK)



| Slide number 41

| Instructor notes

| Explain the benefits of infrastructure as code (IaC) over traditional infrastructure deployment. Discuss methods of traditional deployment, and align them with the issues referenced in the previous slide.

|

| Student notes

A fundamental principle of DevOps is to treat infrastructure the same way that developers treat code. Application code has a defined format and syntax. If the code is not written according to the rules of the programming language, applications cannot be created. Code is stored in a version management or source control system, which logs a history of code development, changes, and bug fixes. When code is compiled or built into applications, we expect a consistent application to be created, and we expect the build to be repeatable and reliable.

Practicing *infrastructure as code* (IaC) means applying the same rigor of application code development to infrastructure provisioning. All configurations should be defined in a declarative way and stored in a source control system, such as AWS CodeCommit—the same as application code. Infrastructure provisioning, orchestration, and deployment should also support the use of infrastructure as code.

AWS provides a DevOps-focused way to create and maintain infrastructure. Similar to the way that software developers write application code, AWS provides services that help you create, deploy, and maintain infrastructure in a programmatic, descriptive, and declarative way. These services provide rigor, clarity, and reliability. The AWS services are core to a DevOps methodology and form the underpinnings of numerous higher level AWS DevOps principles and practices.

Infrastructure as code methods

Infrastructure as code can use a *declarative* or *imperative* programming approach

| Declarative | Imperative |
|---|--|
| <ul style="list-style-type: none">• Defines the resources• Defines the resource properties• Defines the system state• Defines the tools that are used to configure your environment <p>Defines what, not how</p> | <ul style="list-style-type: none">• Provides the equivalent of step-by-step instructions• Focuses on the desired end state <p>Defines what <i>and</i> how, step by step</p> |



| Slide number 42

| Instructor notes

| Explain the key differences between declarative and imperative programming. Focus on the issues of scalability that imperative programming can be susceptible to.

|

| Student notes

Infrastructure as code can use a declarative or imperative programming approach. A *declarative* programming approach defines resources and their properties, the system state, and the tools to be used to configure your environment. However, an *imperative* programming approach provides the equivalent of step-by-step instructions to achieve the desired end state. The choice of declarative and imperative will depend on factors like coding language selection and performance considerations.

AWS CloudFormation

- Is a fully managed service
- Creates, updates, and deletes resources in stacks
- Automates AWS resource provisioning
- Simplifies the task of repeatedly and predictably creating groups of related resources that power your applications



| Slide number 43

| Instructor notes

|

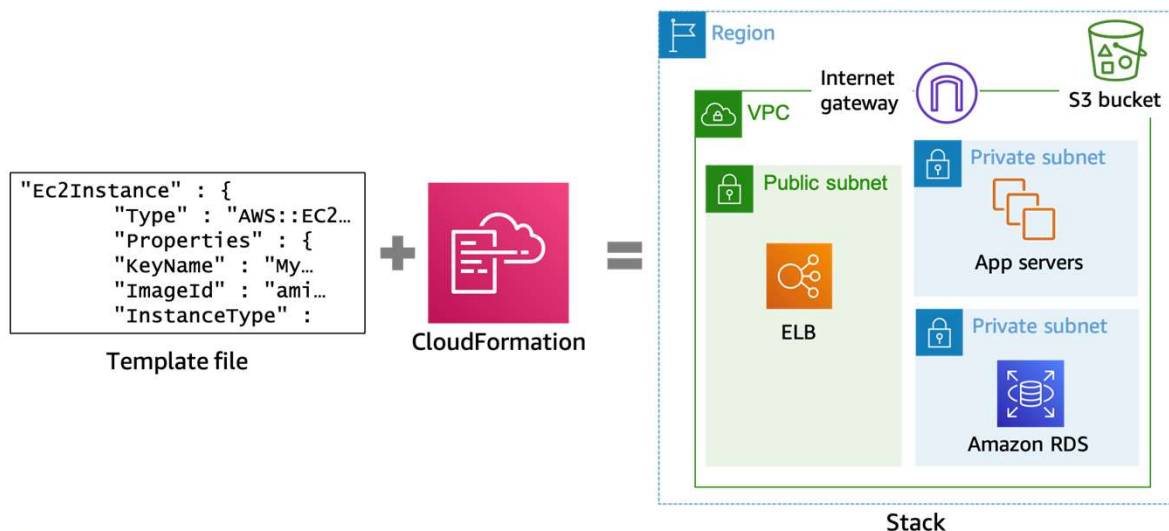
| Student notes

AWS CloudFormation is a fully managed service that provides a common language for you to describe and provision all of the infrastructure resources in your cloud environment. CloudFormation creates, updates, and deletes the resources for your applications in environments called *stacks*. A stack is a collection of AWS resources that are managed as a single unit.

CloudFormation is all about automated resource provisioning—it simplifies the task of repeatedly and predictably creating groups of related resources that power your applications. Resources are written in text files by using JSON or YAML format.

CloudFormation supports the infrastructure needs of many types of applications, such as existing enterprise applications, legacy applications, applications built using a variety of AWS resources, and container-based solutions (including those built with AWS Elastic Beanstalk).

How CloudFormation works



| Slide number 44

| Instructor notes

|

| Student notes

You can automate the provisioning of these AWS resources by using CloudFormation. CloudFormation reads *template files*, which provide instructions for what resources need to be provisioned. CloudFormation constructs the resources listed in the template file. The output of this process is your environment, or *stack*. You can create a template that creates a single resource stack or a stack with hundreds of resources.

You can interact with CloudFormation by using the console, AWS CLI, and the AWS SDKs or APIs directly.

CloudFormation template structure

```
{
  "AWSTemplateFormatVersion" : "version date",
  "Description" : "JSON string",
  "Metadata" : {template metadata},
  "Parameters" : {set of parameters},
  "Mappings" : {set of mappings},
  "Conditions" : {set of conditions},
  "Transform" : {set of transforms},
  "Resources" : {set of resources},
  "Outputs" : {set of outputs}
}
```



| Slide number 45

| Instructor notes

|

| Student notes

This slide illustrates the CloudFormation template structure and its sections. Templates include the following major sections:

- **Format version:** The CloudFormation template version that the template conforms to, which identifies the capabilities of the template. The latest template format version is 2010-09-09 and is currently the only valid value.
- **Description:** A text string that describes the template.
- **Metadata:** Objects that provide additional information about the template.
- **Parameters:** Values to pass to your template at runtime when you create or update a stack.
- **Mappings:** A mapping of keys and associated values, which you can use to specify conditional parameter values, similar to a lookup table.
- **Conditions:** Conditions control whether certain resources are created or whether certain resource properties are assigned a value during stack creation or update. For example, you could conditionally create a resource that depends on whether the stack is for a production or test environment.
- **Transform:** For serverless (that is, Lambda based) applications, this specifies the version

of the AWS Serverless Application Model (AWS SAM) to use. When you specify a transform, you can use AWS SAM syntax to declare resources in your template. The model defines the syntax that you can use and how it is processed. You will learn about AWS SAM in another section of this module.

- **Resources:** Specifies the stack resources and their properties, such as an EC2 instance or S3 bucket. *This is the only required section.*
- **Outputs:** Describes the values that are returned whenever you view your stack's properties.


You don't need to list your resources in the template in the exact order of creation. You can use the DependsOn attribute to specify the order in which CloudFormation will create the resources. In this way, you can build a sequence of events, such as creating a database server before a web server.

CloudFormation template example: Description

```
{
  "AWSTemplateFormatVersion": "2010-09-09",

  "Description": "AWS CloudFormation Sample Template
Amazon S3 Pipeline: Configure and launch an
Infrastructure that includes a pipeline that connects
to an Amazon S3 source bucket."

  "Parameters": {
    "SourceObjectKey": {
      "Description": "S3 source artifact",
      "Type": "String",
      "Default": "SampleApp_Linux.zip"
    },
  },
}
```



| Slide number 46

| Instructor notes

|

| Student notes

The description section is a text string that describes the template. This example gives a description of a CloudFormation sample template that can be used to configure and launch an infrastructure that includes a pipeline that connects to an S3 source bucket.

CloudFormation template example: Parameters

```
{
  "AWSTemplateFormatVersion": "2010-09-09",

  "Description": "AWS CloudFormation Sample Template
Amazon S3 Pipeline: Configure and launch an
Infrastructure that includes a pipeline that connects
to an Amazon S3 source bucket."

  "Parameters": {
    "SourceObjectKey": {
      "Description": "S3 source artifact",
      "Type": "String",
      "Default": "SampleApp_Linux.zip"
    },
  },
```

Parameters
section



| Slide number 47

| Instructor notes

|

| Student notes

Parameters are values to pass to your template at runtime when you create or update a stack. In this example, the SourceObjectKey parameter has a default of SampleApp_Linux.zip.

CloudFormation template example: Resources

```
"AWSTemplateFormatVersion": "2010-09-09",
"Resources": {
  "AppPipeline": {
    "Type": "AWS::CodePipeline::Pipeline",
    "Properties": {
      "Name": "s3-events-pipeline",
      "RoleArn": {
        "Fn::GetAtt": [
          "CodePipelineServiceRole",
          "Arn"
        ]
      }
    }
  },
}
```

Resources
section



| Slide number 48

| Instructor notes

|

| Student notes

The resources section is the only required section of the CloudFormation template. This section specifies the stack resources and their properties, such as an EC2 instance or S3 bucket. This example describes a pipeline for Amazon S3 events.

AWS Cloud Development Kit (AWS CDK)

- AWS CDK apps are written in TypeScript, JavaScript, Python, Java, C#, or Go.
- AWS CDK apps use the AWS CDK to define AWS infrastructure.
- An app defines one or more stacks that each contain constructs that define concrete AWS resources.
- The AWS CDK includes the CDK Toolkit—a command line tool to work with AWS CDK apps and stacks.



| Slide number 49

| Instructor notes

|

| Student notes

An AWS Cloud Development Kit (AWS CDK) *app* is an application written in TypeScript, JavaScript, Python, Java, C#, or Go that uses the AWS CDK to define AWS infrastructure. An app defines one or more *stacks*. Stacks contain *constructs*, which define one or more concrete AWS resources, such as S3 buckets, Lambda functions, or DynamoDB tables.

Constructs (as well as stacks and apps) are represented as classes (types) in your programming language of choice. You instantiate constructs within a stack to declare them to AWS and connect them to each other using well-defined interfaces.

The AWS CDK includes the CDK Toolkit (also called the CLI), which is a command line tool to work with your AWS CDK apps and stacks. Among other functions, the Toolkit provides the ability to convert one or more AWS CDK stacks to CloudFormation templates and related assets (a process called *synthesis*) and to deploy your stacks to an AWS account.

The AWS CDK includes a library of AWS constructs called the AWS Construct Library, which is organized into various modules. The library contains constructs for each AWS service. The

main CDK package is called `aws-cdk-lib`, and it contains the majority of the AWS Construct Library, along with base classes such as `Stack` and `App`, which are used in most CDK applications.

Key takeaways: Creating a scalable infrastructure



- Traditional infrastructure deployments use manual processes that are time-consuming and prone to error.
- IaC automates the process to create, update, and delete AWS infrastructure.
- IaC can use a declarative or imperative programming approach.
- You can use CloudFormation to automate AWS resource and pipeline provisioning.

| Slide number 50

| Instructor notes

|

| Student notes

Here are a few key points to summarize this section.

- Traditional infrastructure deployments use manual processes that are time-consuming and prone to error.
- IaC automates the process to create, update, and delete AWS infrastructure.
- IaC can use a declarative or imperative programming approach.
- You can use CloudFormation to automate AWS resource and pipeline provisioning.

Creating scalable components

Securing and Scaling the Data Pipeline



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| **Slide number 51**

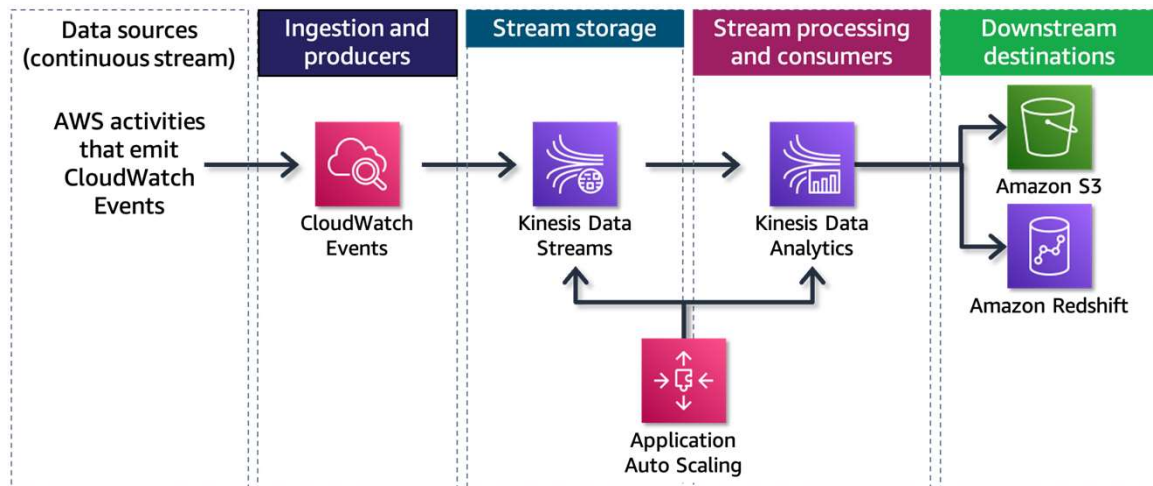
| **Instructor notes**

|

| **Student notes**

This section introduces concepts related to creating scalable components.

Stream processing pipeline



| Slide number 52

| Instructor notes

|

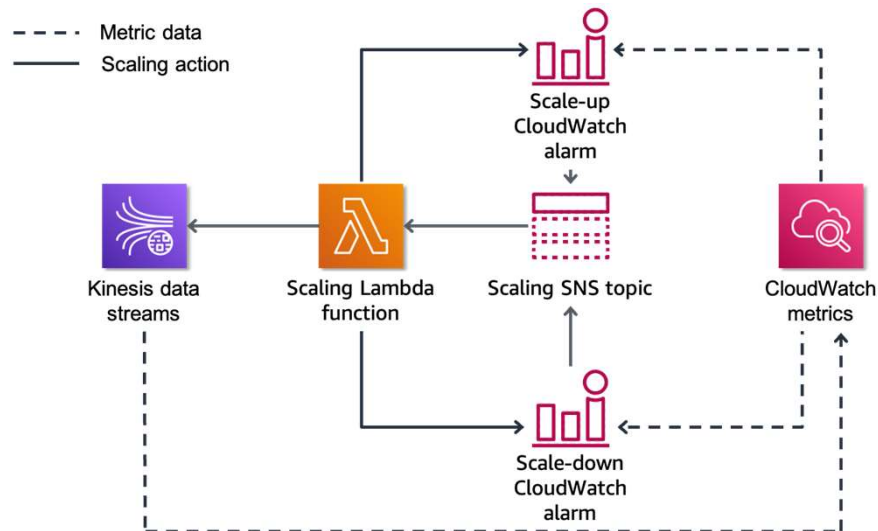
| Student notes

Let's look at the stream processing pipeline example from earlier in this module and focus on the services used in the example. In this example, automatic scaling can be configured for Amazon Kinesis Data Streams by using CloudWatch and Lambda, as you will see on the next slide. Kinesis Data Streams offers an on-demand mode, which you can use to automatically scale your Kinesis data streams without needing to manage provisioning or capacity for your streaming data. Kinesis Data Streams on-demand mode will automatically scale your capacity in response to data traffic. This capability doesn't require you to write new APIs to write or read data and can be helpful when your workload is variable or unknown.

If your stream processing needs to be scaled in conjunction with your stream storage, you can configure the parallel implementation of tasks and the allocation of resources for Kinesis Data Analytics to achieve that scaling. After you have configured the parallel execution for your Kinesis Data Analytics application tasks, Kinesis Data Analytics will automatically scale your application's parallelism to accommodate the data throughput. Automatic scaling is enabled by default, and you can disable or enable this behavior.

The next slide provides a deeper dive into how you can scale Kinesis Data Streams by using Lambda and CloudWatch.

Scaling Kinesis data streams



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

53

| Slide number 53

| Instructor notes

|

| Student notes

This slide shows a solution to automatically scale one or more Amazon Kinesis data streams by using a lightweight serverless architecture. The solution uses CloudWatch, Amazon Simple Notification Service (Amazon SNS), and Lambda. The streams are processed using a single SNS topic and Lambda function. Each stream requires two CloudWatch alarms: one to scale up and one to scale down.

In this workflow, metrics flow from the Kinesis data stream into CloudWatch, where they are evaluated by the scale-up and scale-down alarms. These alarms decide when to scale. When one of the alarms initiates, it sends a message to the scaling SNS topic. Amazon SNS then sends a message to Lambda, and the scaling Lambda function processes the message. The Lambda function then scales the data stream up or down based on the message by using Kinesis shards, either doubling them or halving them. Finally, the metric math on the scale-up and scale-down alarms is updated to reflect the new shard count.

While this method is an effective way of automatically scaling Kinesis data streams, the Kinesis Data Streams on-demand mode is now the recommended way to automatically

scale your Kinesis data streams.

Key takeaways: Creating scalable components



- Configure scalable components in your pipeline to ensure that it is optimized.
- Use CloudWatch and Lambda to provide automatic scaling for Kinesis data streams.
- Kinesis Data Streams on-demand mode is the recommended method for automatic scaling.

| Slide number 54

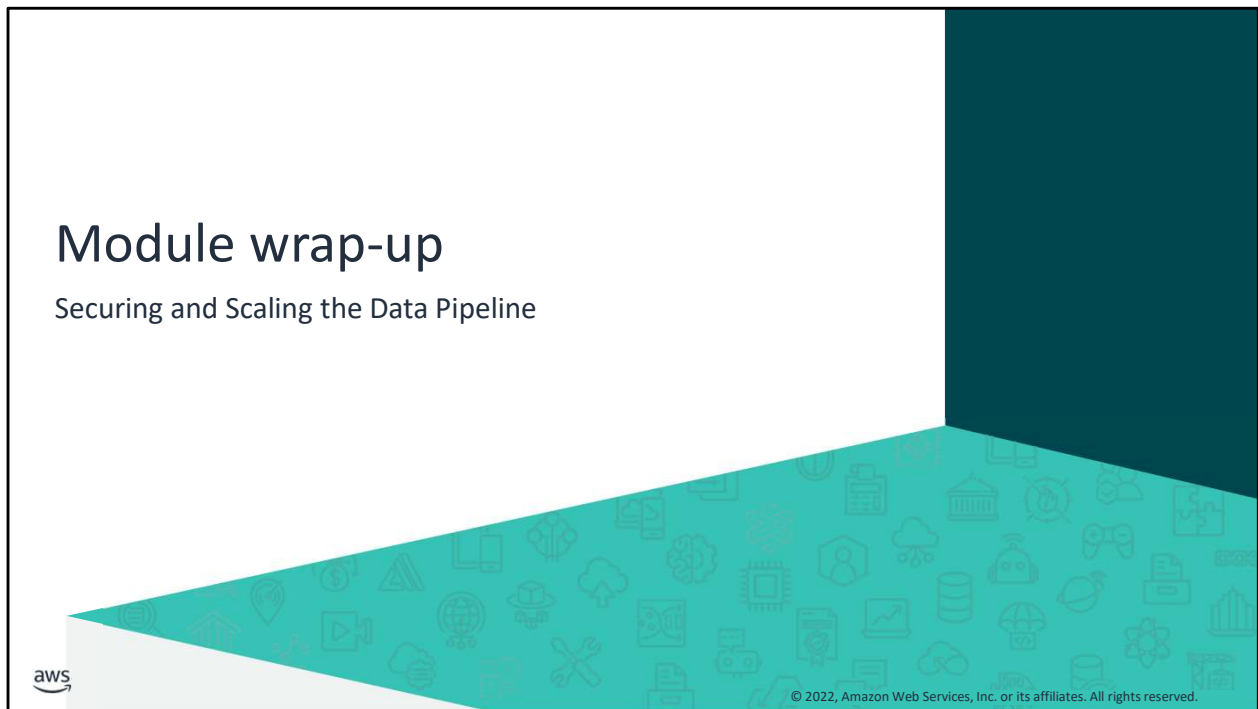
| Instructor notes

|

| Student notes

Here are a few key points to summarize this section.

- Configure scalable components in your pipeline to ensure that it is optimized.
- Use CloudWatch and Lambda to provide automatic scaling for Kinesis data streams.
- Kinesis Data Streams on-demand mode is the recommended method for automatic scaling.



| Slide number 55

| Instructor notes

|

| Student notes

This section summarizes what you have learned and brings the Securing and Scaling the Data Pipeline module to a close.

Module summary

This module prepared you to do the following:

- Highlight how cloud security best practices apply to analytics and ML data pipelines.
- List AWS services that play key roles in securing a data pipeline.
- Describe how IaC supports the security and scalability of a data pipeline infrastructure.
- Identify the function of common CloudFormation template sections.



| Slide number 56

| Instructor notes

This is a good opportunity to use an online group or discussion board to ask students to reflect on what they have learned. You might ask the students to recall a point from the module that aligns to one of the listed objectives. This provides a good segue to the knowledge check and sample exam question.

|

| Student notes

This module prepared you to do the following:

- Highlight how cloud security best practices apply to analytics and ML data pipelines.
- List AWS services that play key roles in securing a data pipeline.
- Cite factors that drive performance and scaling decisions across each layer of a data pipeline.
- Describe how IaC supports the security and scalability of a data pipeline infrastructure.
- Identify the function of common CloudFormation template sections.

Module knowledge check



- The knowledge check is delivered online within your course.
- The knowledge check includes 10 questions based on material presented on the slides and in the slide notes.
- You can retake the knowledge check as many times as you like.

| Slide number 57

| Instructor notes

|

| Student notes

Use your online course to access the knowledge check for this module.

Sample exam question

A data engineer is preparing to deploy a data lake that users and programs throughout the organization will access. The engineer wants to ensure that the Amazon EMR clusters that will support the architecture can maintain a consistent performance standard regardless of the demand.

Which service could the data engineer use to accomplish this task?

Identify the key words and phrases before continuing.

The following are the key words and phrases:

- EMR clusters
- Consistent performance



| Slide number 58

| Instructor notes

| The "AWS Application Auto Scaling" slide includes information on the services that are supported by AWS Application Auto Scaling, specifically mentioning its use with Amazon EMR clusters.

| Student notes

The question notes that the engineer wants consistent performance out of their Amazon EMR clusters regardless of the demand. Increases in demand will require increases in performance, and there are specific services that provide such capabilities.

Sample exam question: Response choices

A data engineer is preparing to deploy a data lake that users and programs throughout the organization will access. The engineer wants to ensure that the Amazon **EMR clusters** that will support the architecture can maintain a **consistent performance** standard regardless of the demand.

Which service could the data engineer use to accomplish this task?

| Choice | Response |
|--------|--|
| A | AWS EC2 Auto Scaling |
| B | AWS Identity and Access Management (IAM) |
| C | AWS Application Auto Scaling |
| D | AWS CloudTrail |



| Slide number 59

| Instructor notes

|

| Student notes

Use the key words that you identified on the previous slide, and review each of the possible responses to determine which one best addresses the question.

Sample exam question: Answer

The correct answer is C.

| Choice | Response |
|--------|------------------------------|
| C | AWS Application Auto Scaling |



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

60

| Slide number 60

| Instructor notes

|

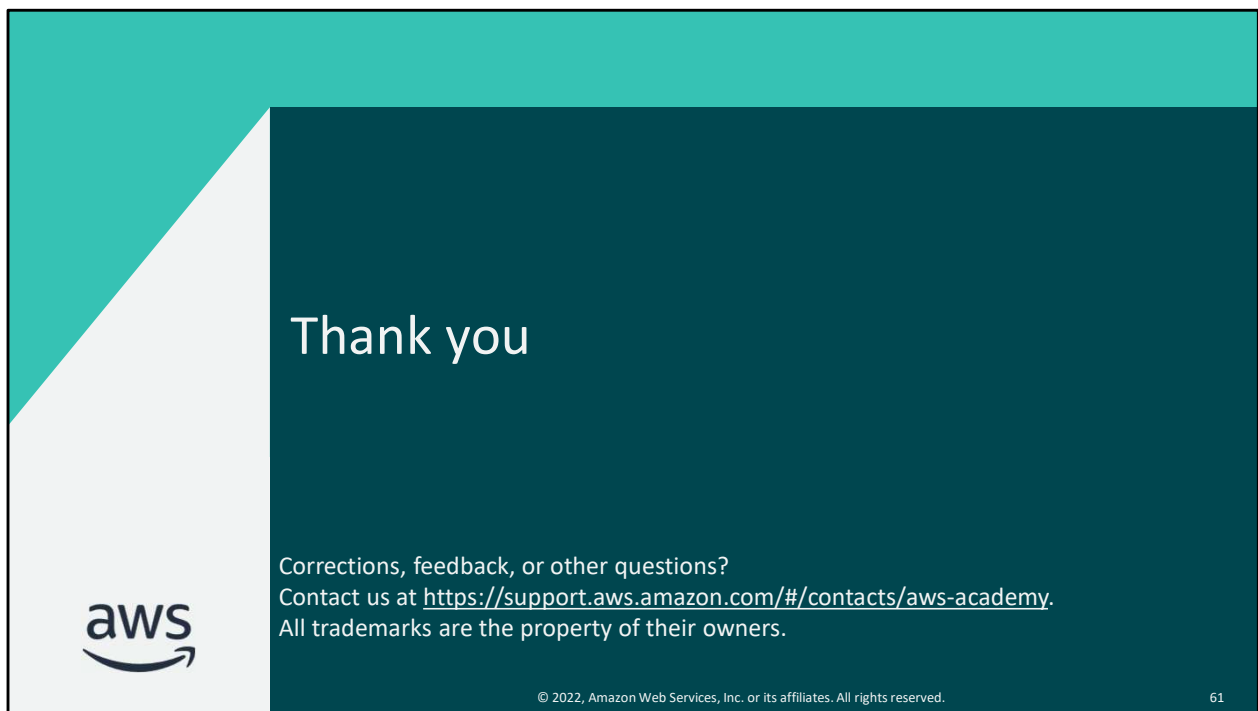
| Student notes

Choice A (AWS EC2 Auto Scaling) would be an appropriate answer if the data engineer needed to scale EC2 resources, but in this instance, a service needs to be scaled.

Choice B (IAM) does not provide the data engineer with the ability to scale an Amazon EMR cluster.

Choice D (CloudTrail) would provide logging of the events but does not provide any scaling capability.

Choice C (Application Auto Scaling) is the correct answer. The data engineer can implement Application Auto Scaling to ensure that the Amazon EMR clusters scale with respect to increased or decreased demand to provide consistent performance for workloads.



~Script note: Thanks for watching

| Slide number 61

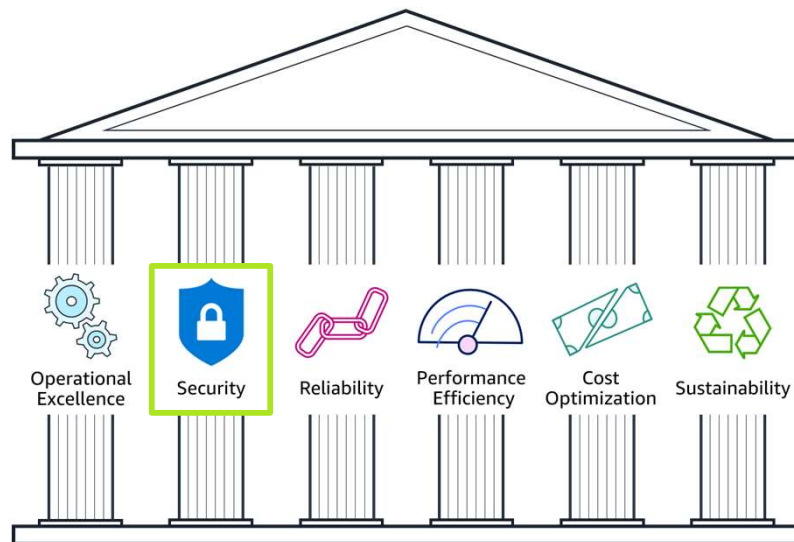
| Instructor notes

|

| Student notes

That concludes this module. The Content Resources page of your course includes links to additional resources that are related to this module.

AWS Well-Architected Framework: Security



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

62

| Slide number 62

| Instructor notes

| Suggestion: Show students where to find the security information for AWS Well-Architected Framework:

1. | Go to the main AWS Well-Architected Framework website, and go to the pillars section of the page to show the six pillars.
2. | Open the Framework Overview page.
3. | To show the security design principles, go to **The Pillars of the Framework > Security > Design Principles**.
4. | To show the questions and best practices of the security pillar, go to **Appendix: Questions and Best Practices > Security**.

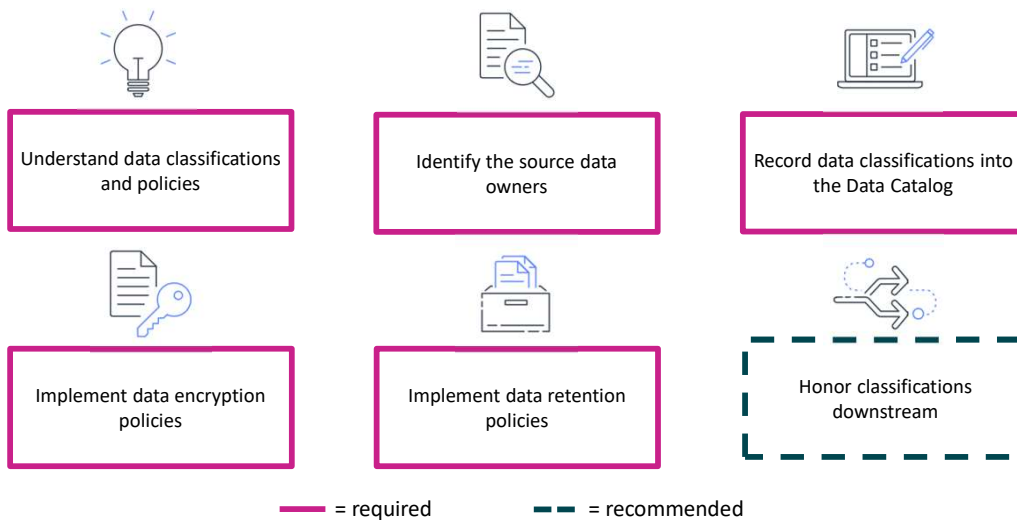
| Student notes

The AWS Academy Cloud Foundations course introduced the AWS Well-Architected Framework. The framework is available on the AWS website, and you can find the link on the Content Resources page in your online course. You might find it helpful to open the website now to look at the way that the pillars are organized.

The framework comprises six pillars: Operational Excellence, Security, Reliability,

Performance Efficiency, Cost Optimization, and Sustainability. The framework provides best practices and design guidance across each pillar to help you make choices and review existing architectures. This module focuses on using the best practices from the security pillar to secure your data analytics and ML workloads by using the corresponding AWS Well-Architected Lenses.

Classify and protect data



| Slide number 63

| Instructor notes

| Slides 19 (Classify and protect data) through 21 (Control the access to workload infrastructure) discuss the best practices that are outlined in the AWS Well-Architected Framework Data Analytics Lens.

|

| Student notes

The first group of best practices falls under the category of *classify and protect data*. This category contains six best practices that focus on classifying and protecting data in analytics workloads. Of the six best practices, five are given a priority of *required*, and one has a priority of *recommended*.

As the analytics workload owner, honor the data classifications and protection policies that the data owners assigned to the source data that you're ingesting. Share these classifications and policies with the downstream data consumers. This sharing of classifications will allow for the data to be properly protected and retained based on organizational policies.

Understand data classifications and their protection policies.

To properly classify data, you first need to understand the data classifications and the

affiliated protection policies of your organization. The levels and their titles will vary based on industry and organizational compliance requirements, so ensure that you're using the correct classifications for your organization. If needed, use the Data Classification: Secure Cloud Adoption whitepaper to assist you to identify the possible classification levels that you are working with. After you determine the possible classifications, allow the data owners to classify the data and define the access rules based on the sensitivity and criticality of the data. Next, identify security zone models to isolate data based on classification by designing security zone models, beginning at the AWS account level and extending down to the AWS resource level. Finally, identify sensitive information and define your protection policies. Amazon Macie can help you identify sensitive data by using custom data identifiers. Review the sensitivity and criticality level of the data, and implement the appropriate data protection policies.

Identify the source data owners and have them set the data classifications.

Identify the owners of the source data to confirm the classification of their data to ensure that the appropriate protections are applied within the analytics workload. As the data moves through the analytics workflow, these classifications will ensure that the data is appropriately protected from unauthorized access by personnel or systems. Assign ownership of datasets into the *Data Catalog*, a collection of metadata that helps centralize share, search, and manage permissions. Knowing the identity of the dataset owners can help you to more efficiently troubleshoot any issues with the dataset.

Record data classifications into the Data Catalog so that analytics workloads can understand.

The Data Catalog is only useful if it is up to date. One approach to ensure that the catalog is up to date is by allowing processes to update it. An up-to-date Data Catalog will provide an accurate and reliable record of data locations and classifications, which allows for more effective protection of potentially sensitive data. Two methods are recommended to keep an up-to-date Data Catalog: 1. Use tags to indicate the data classifications and 2. Record the lineage of data to track changes in the Data Catalog.

Using tags to indicate data classifications will allow for the discoverability of data sensitivity without exposing the actual data. Data *lineage*, which is a relation among data and the processing systems, should be discoverable, recordable, and visualizable from source to downstream systems to understand the data's source, what data derived from it, and which systems are reading it downstream.

Implement encryption policies for each class of data in the analytics workload.

Encrypt workload data in accordance with the policies that govern the classification level. Implement encryption policies for both data at rest and data in transit. You can find AWS service-specific implementation guides on the AWS Documentation website. AWS KMS provides you with a simple and robust mechanism to manage your encryption keys.

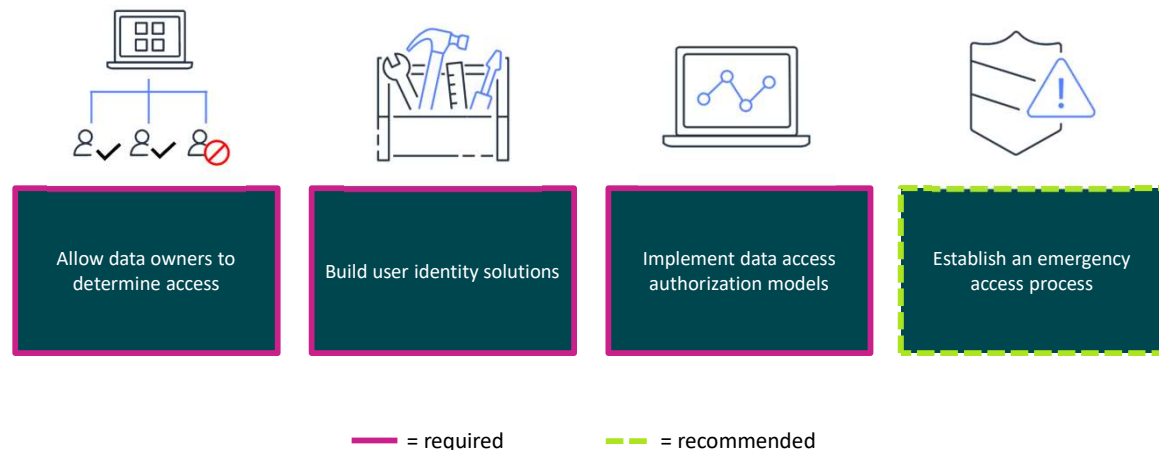
Implement data retention policies for each class of data in the analytics workload.

While your organization might have a standard data retention policy, you should also implement classification-based retention policies. Create backup requirements and policies based on data classifications. Ensure that you align them with the appropriate recovery point objective (RPO), recovery time objective (RTO), data classification, and any compliance and audit requirements that your organization must adhere to. Create data retention requirement policies based on the data classifications, and tailor your policies to individual assets based on retention requirements—avoid blanket retention policies. Use analytics systems to implement rules that retain in-scoped data that aligns with your organization's retention policies.

Require downstream systems to honor the classifications.

Remember that other data-consuming systems will access the data that your analytics workload shares. Because of this, you should require those downstream systems to implement the appropriate data classification policies. Have a centralized, shareable catalog with cross-account access to ensure that data owners manage permissions for downstream systems. Having appropriately configured cross-account access will allow downstream systems to use the data from your analytics workload. Another recommendation is to monitor the downstream systems' eligibility to access classified data from the analytics workload. While cross-account access will allow downstream systems to use your analytics data, you want to ensure that those systems are *eligible* to access that data. Periodically verify that any downstream systems are eligible to process data at the appropriate level of classification.

Control the data access



| Slide number 64

| Instructor notes

|

| Student notes

The next group of best practices is categorized under the title of *control the data access*. As the owner of the analytics workload, you should honor the access management policies of the source systems, secure access to the data in the analytics workload, and ensure that any downstream sharing of data is done in compliance with the source system's classification policies.

Allow data owners to determine which people or systems can access data in analytics and downstream workloads.

In general, an analytics workload will consolidate multiple datasets into a central location. Each of those consolidated datasets will likely be owned by different teams or individuals, so it's important to identify which person or group owns which dataset in order to coordinate data access permissions and ensure those permissions adhere to the principle of least privilege. Another best practice is to use an Access Control Matrix to document which users, roles, or systems have access to which datasets and what actions each user, role, or system can perform.

Build user identity solutions that uniquely identify people and systems.

To federate with IAM by using AWS IAM Identity Center (successor to AWS Single Sign-On) or another identity provider, a best practice is to centralize your workforce identities. For example, you can map IAM roles to AWS Glue Data Catalog resource policies to effectively manage access.

Implement the required data access authorization models.

As much as possible, limit access to data stores that contain sensitive data. One way that you can effectively manage this access is by using control mechanisms such as role-based access control (RBAC). Another best practice is to implement dataset-level access controls by building analytics workloads to have the dataset owners control the data access per dataset level. Finally, implement column-level data access controls and enforce the masking of sensitive data through the use of column-level restrictions. This practice will help ensure that end users of analytics applications are not inadvertently exposed to sensitive data.

Establish an emergency access process to the source, analytics, and downstream systems.

Though unlikely, automated processes and pipelines can experience occasional issues. You can remediate such issues more efficiently by expediting access through the use of emergency access. Ensure that risk analysis is done on your analytics workload by identifying emergency situations and a procedure to allow such emergency access. Conducting such risk analysis can help you identify potential threats from source systems, analytics workloads, and downstream systems. As you identify risks, prioritize each event based on likelihood and overall business impact level. Discuss these findings with your source and downstream system owners, and determine how you could continue to allow analytics workloads to access those systems.

Control the access to workload infrastructure



Prevent unintended access



Implement least privilege policies



Monitor the infrastructure changes and user activities



Secure infrastructure audit logs

— = required

— = recommended



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

65

| Slide number 65

| Instructor notes

|

| Student notes

The final group of data analytics best practices are categorized as *control the access to workload infrastructure*. Through the use of services such as IAM, you can ensure that your environment is accessible while adhering to the principle of least privilege.

Prevent unintended access to the infrastructure.

Prevent unintended or unauthorized access to your infrastructure by granting the least privilege possible. IAM provides the AWS Identity and Access Management Access Analyzer tool for all AWS accounts that are centrally managed through AWS Organizations. IAM Access Analyzer aids you to set and verify permissions, and assists you to refine those permissions by removing unused access. Infrastructure boundaries are yet another way that you can prevent unauthorized access. You can create network boundaries by using VPC private subnets to isolate your analytics resources and allowing only the minimal amount of connections that are required to support your analytics workload.

Implement least privilege policies for source and downstream systems.

Identify your source and downstream users and systems, and then identify the minimum

privileges that each would require and implement only those permissions necessary. For example, if a downstream Amazon QuickSight service needs access to your Amazon OpenSearch Service data, ensure that the QuickSight users have the minimal amount of access that is required. You can further secure access by implementing the two-person rule. This practice can help you prevent unauthorized access, altering, or even deletion of critical or classified information by requiring the presence of two separate users to perform tasks that you deem critical.

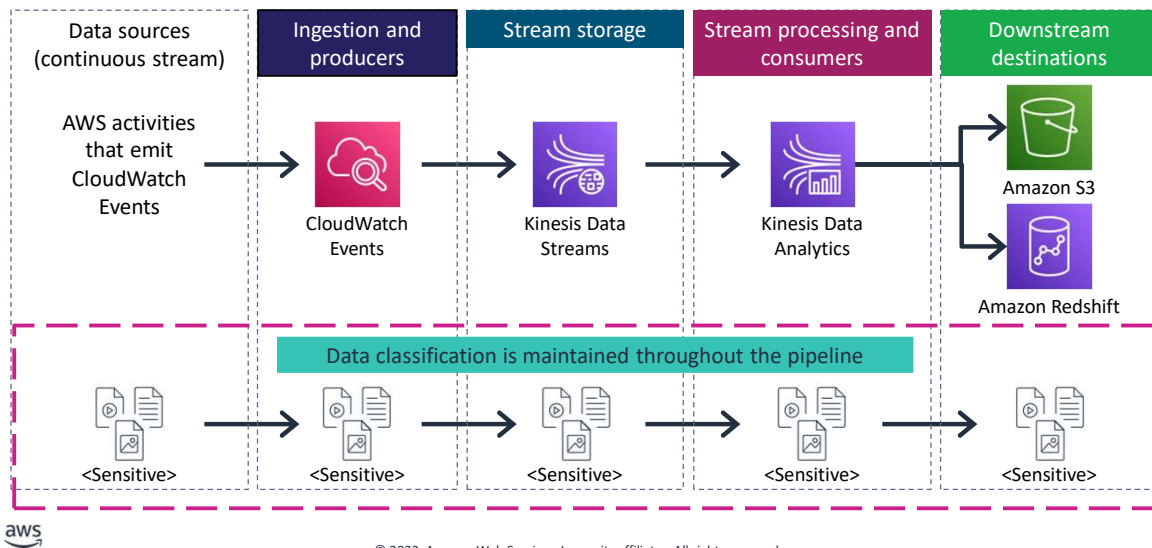
Monitor the infrastructure changes and the user activities against the infrastructure.

Monitor infrastructure changes closely. Practices such as implementing a change control board or change advisory board can be helpful to prevent potentially disruptive or harmful changes. AWS services such as AWS Config, Amazon Inspector, and Amazon GuardDuty can assist you to monitor and log those changes. For security and accounting purposes, you should also monitor and log user activities against your infrastructure. CloudTrail can provide you with reviewable audit logs of user actions against your resources and can be paired with CloudWatch for near real-time monitoring.

Secure the audit logs that record every data or resource access in analytics infrastructure.

While services such as AWS Config and CloudTrail can provide you with a wealth of audit logs, these records are useless unless they are also securely handled and maintained. Use fault-tolerant storage for your logs, and restrict access to the logs to privileged users. Regularly monitor, log, and audit access to audit logs as well. Review the audit log features of your analytics systems, and ensure that auditing is active and delivered to the fault-tolerant storage that you have selected.

Securing the stream processing pipeline



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

66

| Slide number 66

| Instructor notes

| Use the operational analytics security scenario to tie together the best practices discussed in this section. This slide discusses how the best practices from the preceding slides could be applied to a stream processing pipeline.

| Student notes

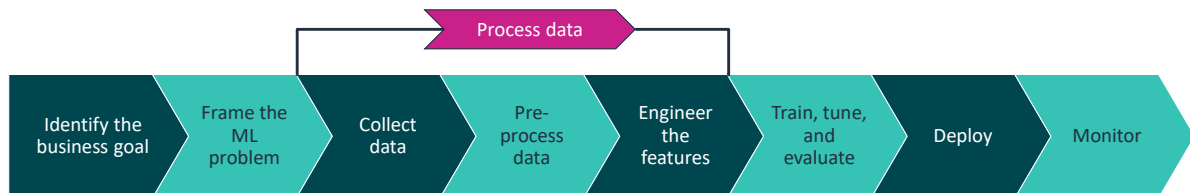
Let's look at how to apply some of the security best practices from the AWS Well-Architected Data Analytics Lens to a typical stream processing pipeline.

- **Understand data classifications and their protection policies:** This best practice will govern how you handle all of the data that flows through your data pipeline. Review your organization's policies for classifying sensitive data, and know what steps you will need to take to ensure adherence to those policies.
- **Identify the source data owners and have them set the data classifications:** The source data types that are listed in the operational data section of your pipeline will likely have various teams or individuals as owners. Identify who the dataset owners are, and request that they properly classify their datasets if they haven't already done so.
- **Record data classifications into the Data Catalog so that analytics workloads can understand:** Keep your Data Catalog up to date so that you have accurate and reliable records of data locations and classifications.

- **Implement encryption policies for each class of data in the analytics workload:** After you identify the source data that you will work with and have established the classification level of each dataset, you will need to implement the applicable encryption policies. For data at rest, use one of the multiple encryption options available in Amazon S3. Secure your Amazon Kinesis data streams with server-side encryption by using AWS KMS.
- **Implement data retention policies for each class of data in the analytics workload:** Use classification-based retention policies for your datasets. Back up and retain analytics datasets based on your organization's policies for classified data.
- **Require downstream systems to honor the classifications:** Ensure that downstream services (such as Amazon Redshift), storage services, and downstream workloads honor your data classifications. If confidential source data enters your pipeline, handle the output of your pipeline as confidential data.
- **Allow data owners to determine which people or systems can access data in analytics and downstream workloads:** This applies to the owners of the operational data shown in the diagram as well as any downstream workloads that could be using the data that your workload produces.
- **Build user identity solutions that uniquely identify people and systems:** Implement IAM or another centralized identity management solution to control which users, roles, or services can access your resources. Every service shown in the diagram integrates with IAM and should be correctly configured to do so.
- **Implement the required data access authorization models:** The operational data in this diagram is classified as *sensitive*, which means that appropriate control measures must be put in place to adequately secure the data.
- **Establish an emergency access process to the source, analytics, and downstream systems:** Each service and resource in the pipeline should have access controls implemented. Ensure that you implement emergency access capabilities that would enable expedited access to your workload in the event of an issue with your pipeline.
- **Prevent unintended access to the infrastructure:** Implement IAM to control user, role, and system access to the data and services within your workload. Isolate your network as much as possible while maintaining only the network connections that are necessary to support your analytics workload.
- **Implement least privilege policies for source and downstream systems:** After you have identified your source and downstream users and systems, identify the minimum privileges that each would require and implement only the necessary permissions. For example, if the downstream Amazon RedShift service needs access to your Amazon Kinesis Data Analytics service data, ensure that the Amazon Redshift users have the minimal amount of access that is required.
- **Monitor the infrastructure changes and the user activities against the infrastructure:** After you secure access to your operational analytics pipeline, it's vital that you monitor it for infrastructure changes and malicious activity. Use additional AWS services, such as AWS Config and GuardDuty, to monitor your infrastructure. Use CloudTrail in conjunction with IAM to log user activities against your resources.
- **Secure the audit logs that record every data or resource access in analytics**

infrastructure: Finally, secure the audit logs for all of the services in your operational analytics pipeline and the services that support it. Ensure that you maintain adequate log security over a fault-tolerant storage solution, such as Amazon S3.

AWS Well-Architected Framework: ML Lens



ML lifecycle



| Slide number 67

| Instructor notes

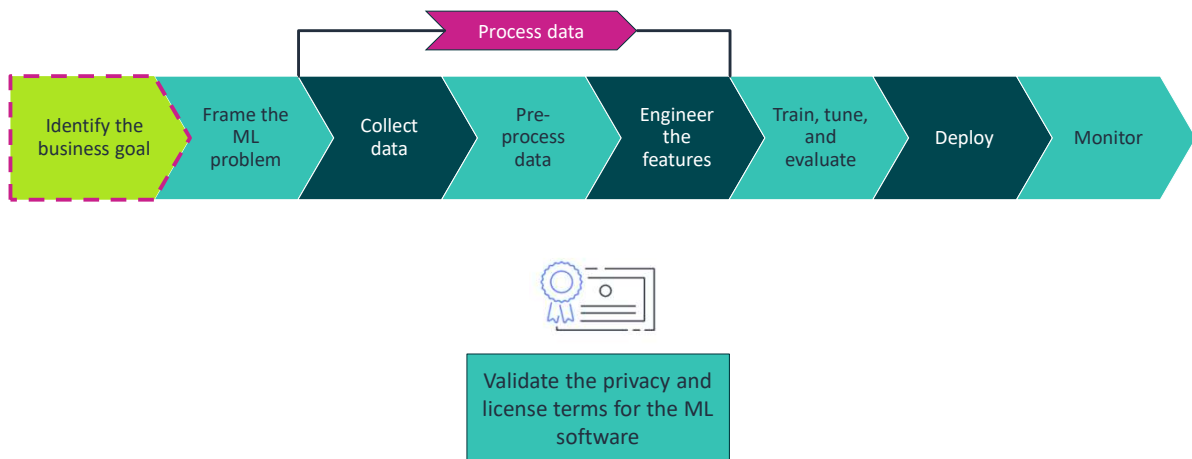
| The student notes point to six phases of the ML lifecycle, but the next several slides display what appears to be eight phases. The *collect data*, *pre-process data*, and *engineer the features* steps are grouped together as a single element that is referred to as the *process data* phase.

|

| Student notes

The ML lifecycle consists of six phases, which are portrayed here in a linear fashion. Note that *collect data*, *pre-process data*, and *engineer the features* are grouped together in the *process data* phase. The AWS Well-Architected Machine Learning Lens provides best practices from all five pillars, but this module focuses on the applicable security best practices.

Phase: Identify the business goal



| Slide number 68

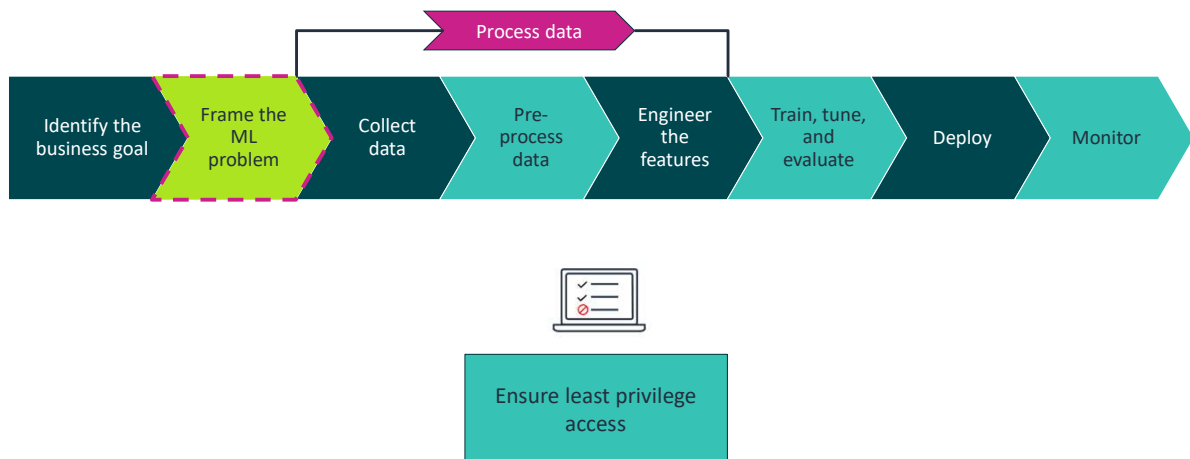
| Instructor notes

| Slides 26 (Phase: Identify the business goal) through 31 (Phase: Monitor) discuss the security best practices that are outlined in the AWS Well-Architected Machine Learning Lens.

| Student notes

Begin by identifying your business goal. ML libraries and packages handle all of the data processing, model development, training, and hosting. You will need to review the privacy and license agreements for all of the software and ML libraries that your ML lifecycle depends on to ensure that they comply with your organization's legal, privacy, and security terms and conditions. Review these items regularly, and establish a process to do so. You can control much of the software through the use of private repositories, or repos.

Phase: Frame the ML problem



| Slide number 69

| Instructor notes

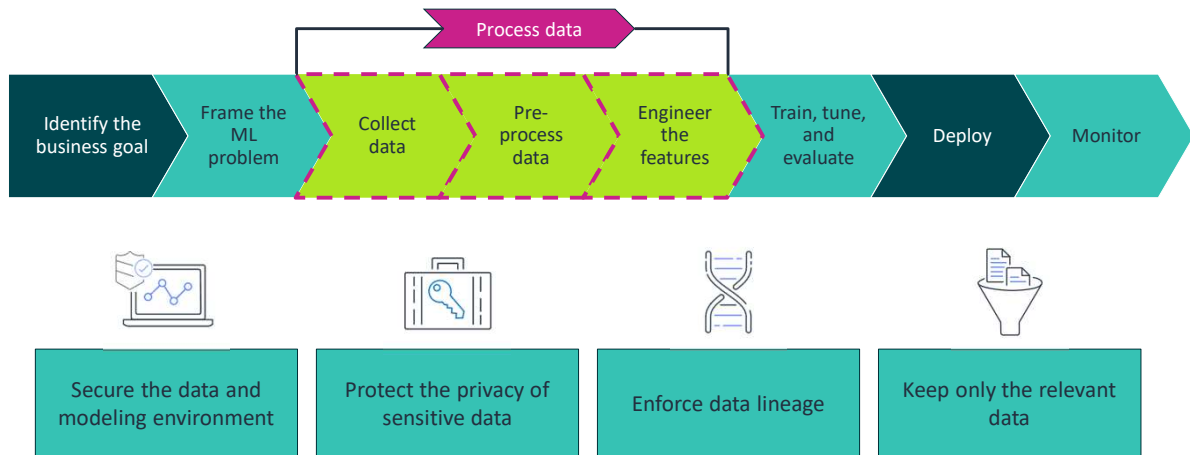
|

| Student notes

Apply the principle of least privilege throughout the ML lifecycle. Provide your project with dedicated network environments that have dedicated resources and services. This will assist you to achieve least privilege access to data, assets, and services.

Restrict access to data based on individual roles. Identify which roles will need data access to build models, features, and algorithms, and map those roles to access patterns by using role-based authentication.

Phase: Process data



| Slide number 70

| Instructor notes

|

| Student notes

Secure any system or environment that hosts data or enables model development. Store training data on secured storage and repositories. Run data preparation model development in a secure cloud. Tightly control access to the destination compute instances as data moves from the data repositories to the instances. Encrypt data in transit to—and at rest on—the compute and storage infrastructure.

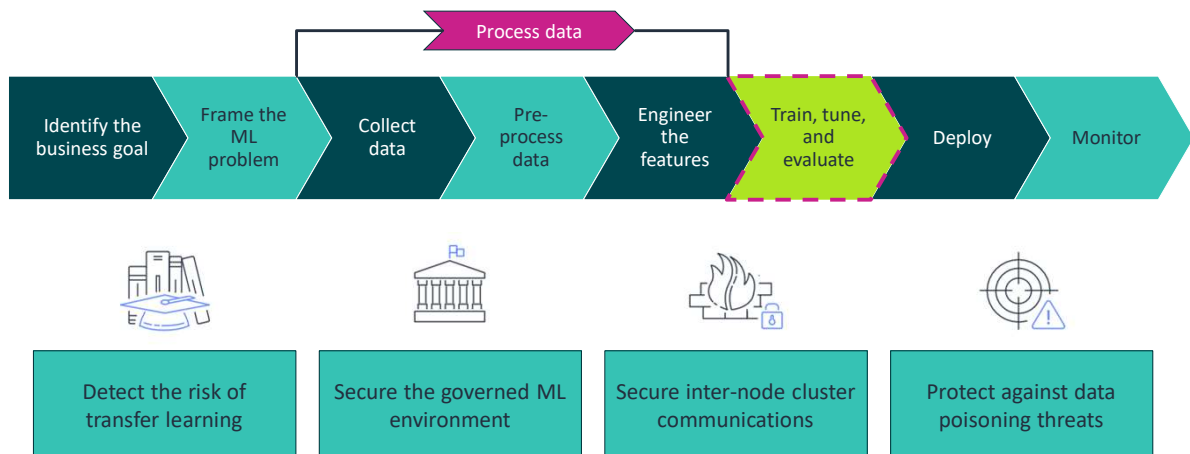
Protect sensitive training data against unintended disclosure. Identify and classify the sensitive data. Handle the sensitive data by using strategies such as removing, masking, tokenizing, and principal component analysis (PCA). Document best governance practices for future reuse and reference.

Monitor and track data origins and transformations over time. Strictly control who can access the data and what they can do with it. Perform preventative controls, auditing, and monitoring to demonstrate how data has been controlled during its lifetime. Implement integrity checks against training data to detect any unexpected deviances caused by loss, corruption, or manipulation. Data lineage enables visibility and helps you trace data

processing errors back to the root cause.

Preserve data across computing environments (such as development and staging), and only store the data that has business need to reduce data exposure risks. Implement mechanisms to enforce a lifecycle management process across the data. Decide when to remove data automatically based on age.

Phase: Train, tune, and evaluate



| Slide number 71

| Instructor notes

|

| Student notes

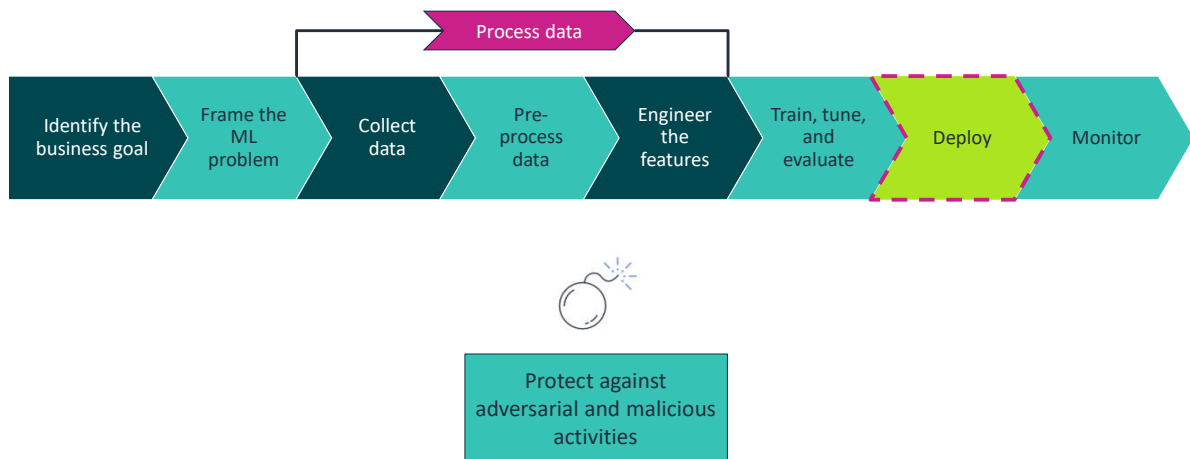
Monitor and ensure that the inherited prediction weights from a transferred model lead to the correct results. This helps minimize the risk of weak learning and incorrect outputs by using pretrained models. *Transfer learning* is an ML technique where a model that was pretrained on one task is fine-tuned on a new task. When using the transfer learning approach, use Amazon SageMaker Debugger to detect hidden problems that might have serious consequences. Examine model predictions to see what mistakes were made. Validate the robustness of your model, and consider how much of this robustness is from the inherited capabilities. Validate input and preprocesses to the model for realistic expectations.

Protect ML operations environments by using managed services with best practices, including detective and preventive guardrails, monitoring, security, and incident management. Explore data in a managed and secure notebook environment. Centrally manage the configuration of Jupyter environments, and enable self-service provisioning for the users.

For frameworks such as TensorFlow, it's common to share information such as coefficients as part of the internode cluster communications. The algorithms require that exchanged information stay synchronized across nodes. Secure this information through encryption in transit. Enable internode encryption in SageMaker, and enable encryption in transit in Amazon EMR.

Protect against data injection and data manipulation that pollutes the training dataset. Data injections add corrupt training data that will result in incorrect models and outputs. Data manipulations change existing data (for example, labels), which can result in inaccurate and weak predictive models. Identify and address corrupt data and inaccurate models by using security methods and anomaly detection algorithms.

Phase: Deploy



| Slide number 72

| Instructor notes

|

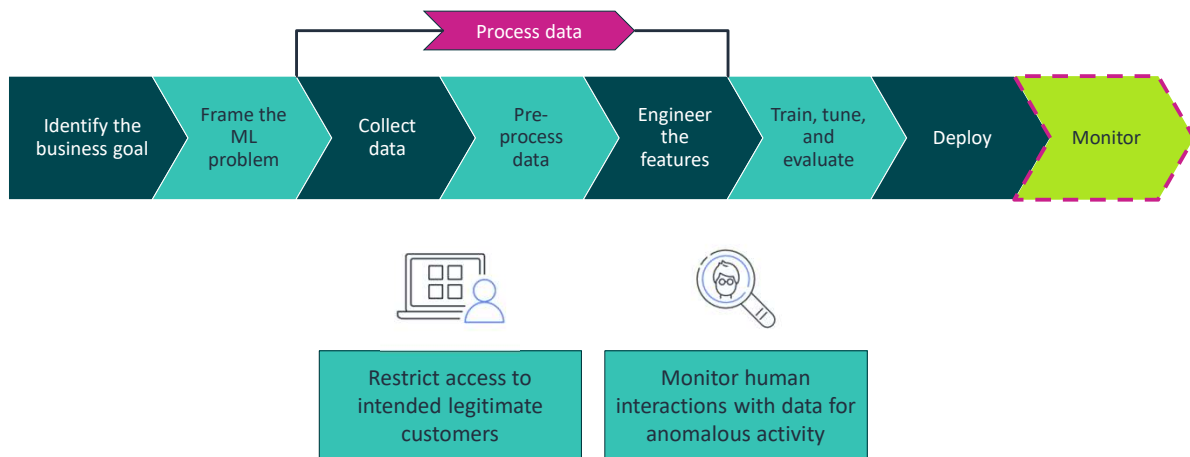
| Student notes

Add protection inside and outside of the deployed code to detect malicious inputs that might result in incorrect predictions. Automatically detect unauthorized changes by examining the inputs in detail. Repair and validate the inputs before they are added back to the pool. Evaluate your use case and determine bad predictions or classifications. Evaluate the robustness of the algorithm against increasingly perturbed inputs to understand susceptibility to manipulated inputs. Select diverse features to improve the algorithm's ability to handle outliers.

Consider pairing models in groups for increased diversity in decisions. Consider using ensembles to build in robustness around decision points. Detect similar repeated inputs to the model to indicate possible threats to the decision boundaries. This can take the form of model brute forcing, where threats iterate only a limited set of variables to determine what influences decision points and derive feature importance. If retraining on untrusted or invalidated inputs, make sure that any model skew is traced back to the data and pruned before retraining a replacement model. Host the model so that a consumer of the model can perform inference against it securely. Enable consumers using the API to define the

relationship, restrict access to the base model, and provide monitoring of model interactions.

Phase: Monitor



| Slide number 73

| Instructor notes

|

| Student notes

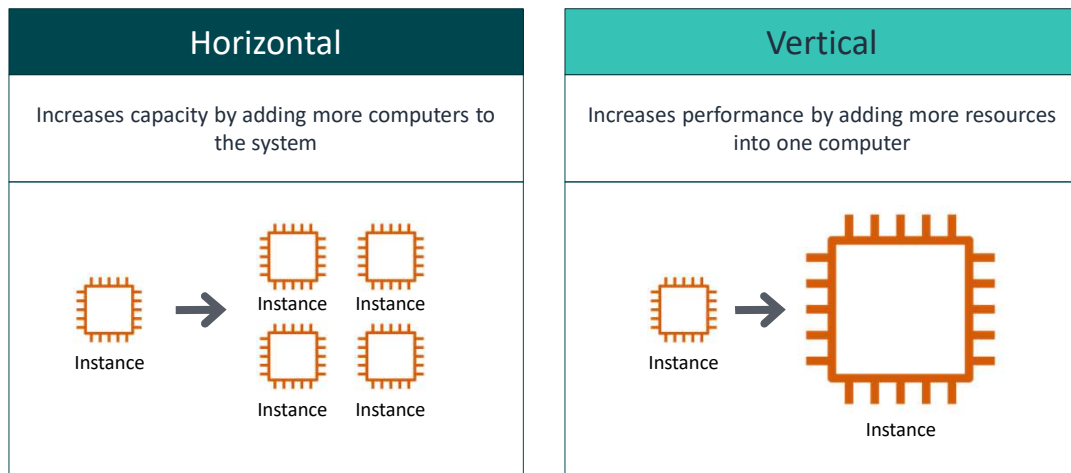
Use least-privileged permissions to invoke the deployed model endpoint. For consumers who are external to the workload environment, provide access through a secure API. Only authorized parties should make inferences against the ML model. Treat inference endpoints as you would any other HTTPS API. Ensure that you follow guidance from the AWS Well-Architected Framework to provide network controls, such as restricting access to specific IP ranges, and bot control. The HTTPS requests for these API calls should be signed so that the requester identity can be verified and the requested data is protected in transit.

Ensure that data access logging is enabled. Audit for anomalous data access events, such as access events from abnormal locations, or activity exceeding the baseline for that entity. Use services and tools that support anomalous activity alerting, and combine their use with data classification to assess risk. Evaluate services that can be used to aid in monitoring data access events. Verify that you have data access logging for all human create, read, update, and delete (CRUD) operations, including the details of who accessed what elements, what action they took, and at what time.

Use Amazon Macie to protect and classify training and inference data in Amazon S3. Macie is a fully managed security service that uses ML to automatically discover, classify, and protect sensitive data in AWS. The service recognizes sensitive data, such as personally identifiable information (PII) or intellectual property. Macie provides visibility into how this data is being accessed or moved. Macie also continually monitors data access activity for anomalies. The service generates detailed alerts when it detects a risk of unauthorized access or inadvertent data leaks.

Use Amazon GuardDuty to monitor for malicious and unauthorized activities. This will enable you to protect AWS accounts, workloads, and data stored in Amazon S3.

Types of scaling



| Slide number 74

| Instructor notes

| Ensure that students understand the differences between horizontal and vertical scaling. Discuss the different methods that are used to accomplish each type of scaling.

|

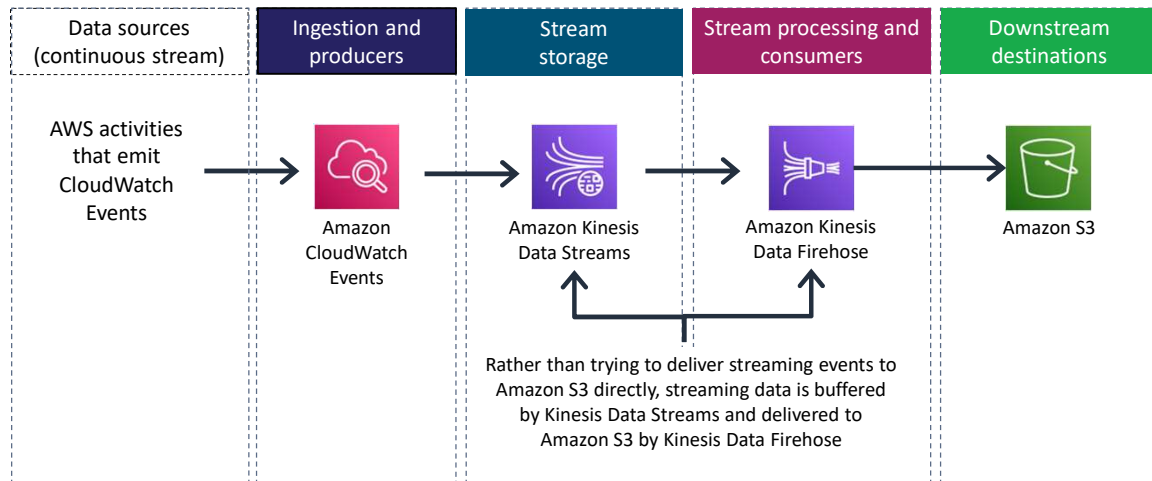
| Student notes

Scalability is a measurement of a system's ability to grow to accommodate an increase in demand. There are two types of scaling: horizontal scaling and vertical scaling.

Horizontal scaling meets the needs of increased demand by adding additional computers—in this case, EC2 instances. An example of horizontal scaling would be adding additional EC2 instances to a high-traffic resource pool to alleviate traffic congestion. You can use horizontal scaling to provide resilience and fault tolerance, but this type of scaling might require you to rework service implementations.

Vertical scaling meets the needs of increased demand by adding additional resources to a computer. Adding additional memory or processing power to an existing EC2 instance can improve analytics workload and ML processing times, but it doesn't improve resilience and fault tolerance.

Example: Scaling in a stream processing pipeline



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

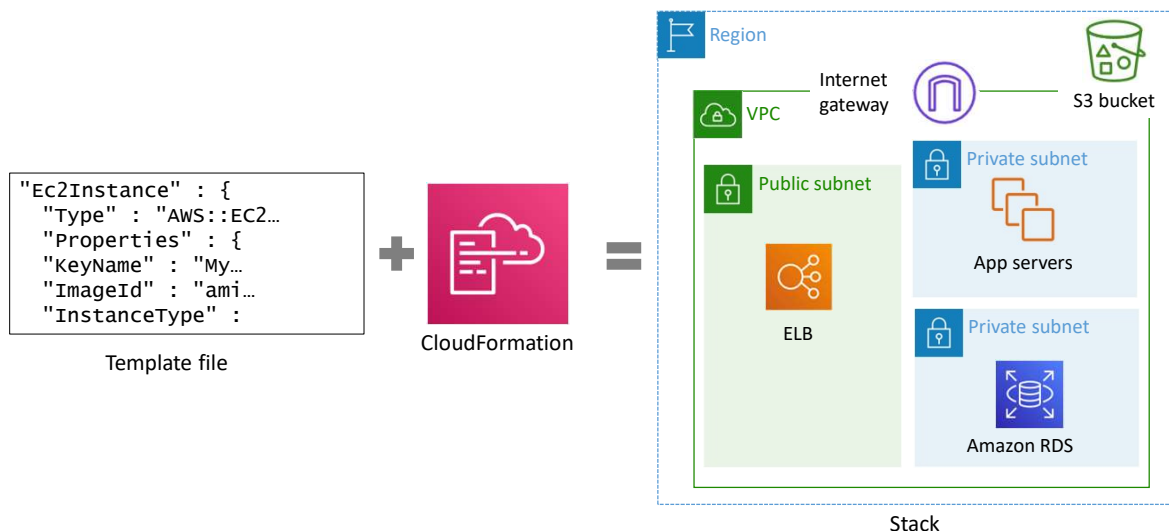
| Slide number 38

| Instructor notes

Let's look at a variation of the stream processing pipeline example from earlier in this module and focus on how scaling is addressed. In this example, the CloudWatch events data is destined for storage in Amazon S3. But you wouldn't want to try to directly write this very high volume of streaming data to Amazon S3. Instead, you can configure an Amazon Kinesis Data Stream stream which provides the temporary buffer for the data. A Kinesis Data Firehose Delivery stream reads data off of the stream in batches and delivers it to Amazon S3. If the number of events increases, Kinesis Data Streams can automatically scale processing up without impacting the downstream systems. However, there is a cost to increasing the capacity on the stream.

For each layer and each set of services in your pipeline, you will need to make decisions and trade-offs about handling the scale of data passing through your pipeline.

How CloudFormation works



| Slide number 76

| Instructor notes

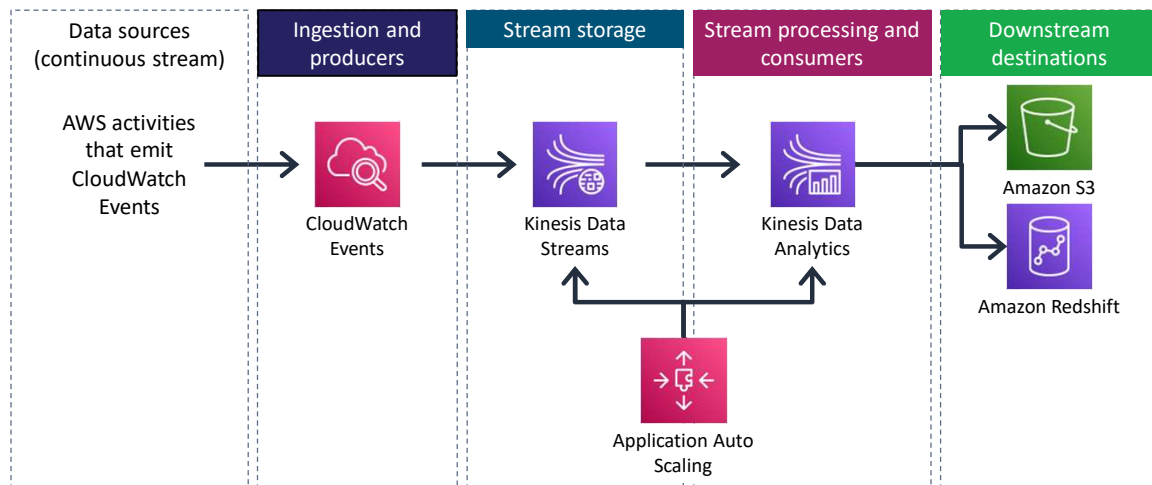
|

| Student notes

You can automate the provisioning of these AWS resources by using CloudFormation. CloudFormation reads *template files*, which provide instructions for what resources need to be provisioned. CloudFormation constructs the resources listed in the template file. The output of this process is your environment, or *stack*. You can create a template that creates a single resource stack or a stack with hundreds of resources.

You can interact with CloudFormation by using the console, AWS CLI, and the AWS SDKs or APIs directly.

Stream processing pipeline



| Slide number 77

| Instructor notes

|

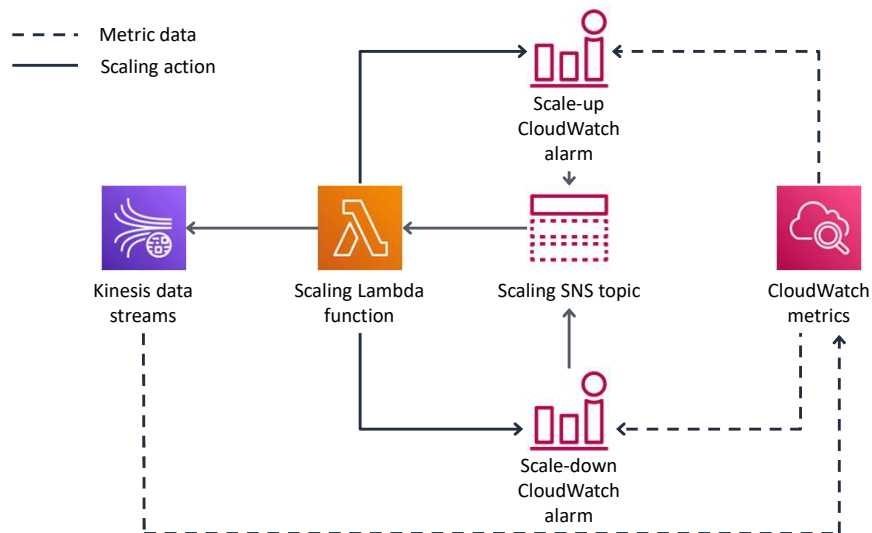
| Student notes

Let's look at the stream processing pipeline example from earlier in this module and focus on the services used in the example. In this example, automatic scaling can be configured for Amazon Kinesis Data Streams by using CloudWatch and Lambda, as you will see on the next slide. Kinesis Data Streams offers an on-demand mode, which you can use to automatically scale your Kinesis data streams without needing to manage provisioning or capacity for your streaming data. Kinesis Data Streams on-demand mode will automatically scale your capacity in response to data traffic. This capability doesn't require you to write new APIs to write or read data and can be helpful when your workload is variable or unknown.

If your stream processing needs to be scaled in conjunction with your stream storage, you can configure the parallel implementation of tasks and the allocation of resources for Kinesis Data Analytics to achieve that scaling. After you have configured the parallel execution for your Kinesis Data Analytics application tasks, Kinesis Data Analytics will automatically scale your application's parallelism to accommodate the data throughput. Automatic scaling is enabled by default, and you can disable or enable this behavior.

The next slide provides a deeper dive into how you can scale Kinesis Data Streams by using Lambda and CloudWatch.

Scaling Kinesis data streams



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

78

| Slide number 78

| Instructor notes

|

| Student notes

This slide shows a solution to automatically scale one or more Amazon Kinesis data streams by using a lightweight serverless architecture. The solution uses CloudWatch, Amazon Simple Notification Service (Amazon SNS), and Lambda. The streams are processed using a single SNS topic and Lambda function. Each stream requires two CloudWatch alarms: one to scale up and one to scale down.

In this workflow, metrics flow from the Kinesis data stream into CloudWatch, where they are evaluated by the scale-up and scale-down alarms. These alarms decide when to scale. When one of the alarms initiates, it sends a message to the scaling SNS topic. Amazon SNS then sends a message to Lambda, and the scaling Lambda function processes the message. The Lambda function then scales the data stream up or down based on the message by using Kinesis shards, either doubling them or halving them. Finally, the metric math on the scale-up and scale-down alarms is updated to reflect the new shard count.

While this method is an effective way of automatically scaling Kinesis data streams, the Kinesis Data Streams on-demand mode is now the recommended way to automatically

scale your Kinesis data streams.