Data Lake Solution

AWS Implementation Guide

November 2016

Last updated September 2018 (see revisions)

Contents

Overview ... 3
Cost ... 4
Architecture Overview ... 4
Solution Features ... 6
AWS CloudFormation Template ... 6
Automated Deployment ... 7
What We'll Cover ... 7
Step 1. Launch the Stack ... 7
Step 2. Log in to the Data Lake Console ... 9
Security ... 9
User Authorization ... 9
Additional Resources ... 10
Appendix A: Federated Template ... 10
AWS CloudFormation Template ... 10
Automated Deployment ... 11
What We'll Cover ... 11
Step 1. Launch the Stack ... 11
Step 2. Complete AD Federation ... 12
Add Amazon Cognito as a relying party in AD FS: ... 13
Enable Sign Out Flow ... 15
Custom Claim Rules ... 16
Appendix B: Solution Components ... 19
AWS KMS Key ... 19
Amazon CloudFront ... 19
Amazon S3 ... 19
Amazon Athena with AWS Glue ... 19
Amazon Cognito User Pool ... 19

Data Lake API and Microservices 20
Admin Microservice 20
Cart Microservice 21
Manifest Microservice 21
Package Microservice 21
Search Microservice 21
Profile Microservice 21
Logging Microservice 21
Amazon DynamoDB Tables 21
Amazon Elasticsearch Service Cluster 22
Appendix C: Collection of Anonymous Data 22
Send Us Feedback 23
Document Revisions 23

About This Guide

This implementation guide discusses architectural considerations and configuration steps for deploying the data lake solution on the Amazon Web Services (AWS) Cloud. It includes links to AWS CloudFormation templates that launch, configure, and run the AWS compute, network, storage, and other services required to deploy this solution on AWS, using AWS best practices for security and availability.

The guide is intended for IT infrastructure architects, administrators, and DevOps professionals who have practical experience architecting on the AWS Cloud.

Overview

Many Amazon Web Services (AWS) customers require a data storage and analytics solution that offers more agility and flexibility than traditional data management systems. A *data lake* is a new and increasingly popular way to store and analyze data because it allows companies to store all of their data, structured and unstructured, in a centralized repository. An effective data lake should provide low-cost, scalable, and secure storage, and support search and analysis capabilities on a variety of data types.

The AWS Cloud provides many of the building blocks required to help customers implement a secure, flexible, and cost-effective data lake. To support our customers as they build data lakes, AWS offers the data lake solution, which is an automated reference implementation that deploys a highly available, cost-effective data lake architecture on the AWS Cloud along with a user-friendly console for searching and requesting datasets. The solution is intended to address common customer pain points around conceptualizing data lake architectures and transforming and analyzing data. The solution automatically configures the core AWS services necessary to easily tag, search, share, and govern specific subsets of data across a company or with other external users. This solution allows users to catalog new datasets, upload datasets with searchable metadata, and to create data profiles for existing datasets in Amazon Simple Storage Service (Amazon S3) with minimal effort.

The data lake solution stores and registers datasets of any size in their native form in the secure, durable, highly-scalable Amazon S3. Customers can upload datasets with searchable metadata and integrate with AWS Glue and Amazon Athena to transform and analyze the data. The solution automatically crawls your data sources, identifies data formats, and then suggests schemas and transformations, so you don’t have to spend time hand-coding data flows. Additionally, user-defined tags are stored in Amazon DynamoDB to add business-relevant context to each dataset. The solution enables companies to create simple governance policies to require specific tags when datasets are registered with the data lake. Users can browse available datasets or search on dataset attributes and tags to quickly find and access data relevant to their business needs.

Additionally, the data lake solution includes a federated template that allows you to launch a version of the solution that is ready to integrate with your existing SAML identity provider such as Microsoft Active Directory. For more information, see Appendix A.

Cost

You are responsible for the cost of the AWS services used while running the data lake solution. The total cost for running this solution depends on the amount of data being loaded, requested, stored, processed, and presented. For full details, see the pricing webpage for each AWS service you will be using in this solution.

Architecture Overview

Deploying this solution with the **default parameters** builds the following environment in the AWS Cloud.

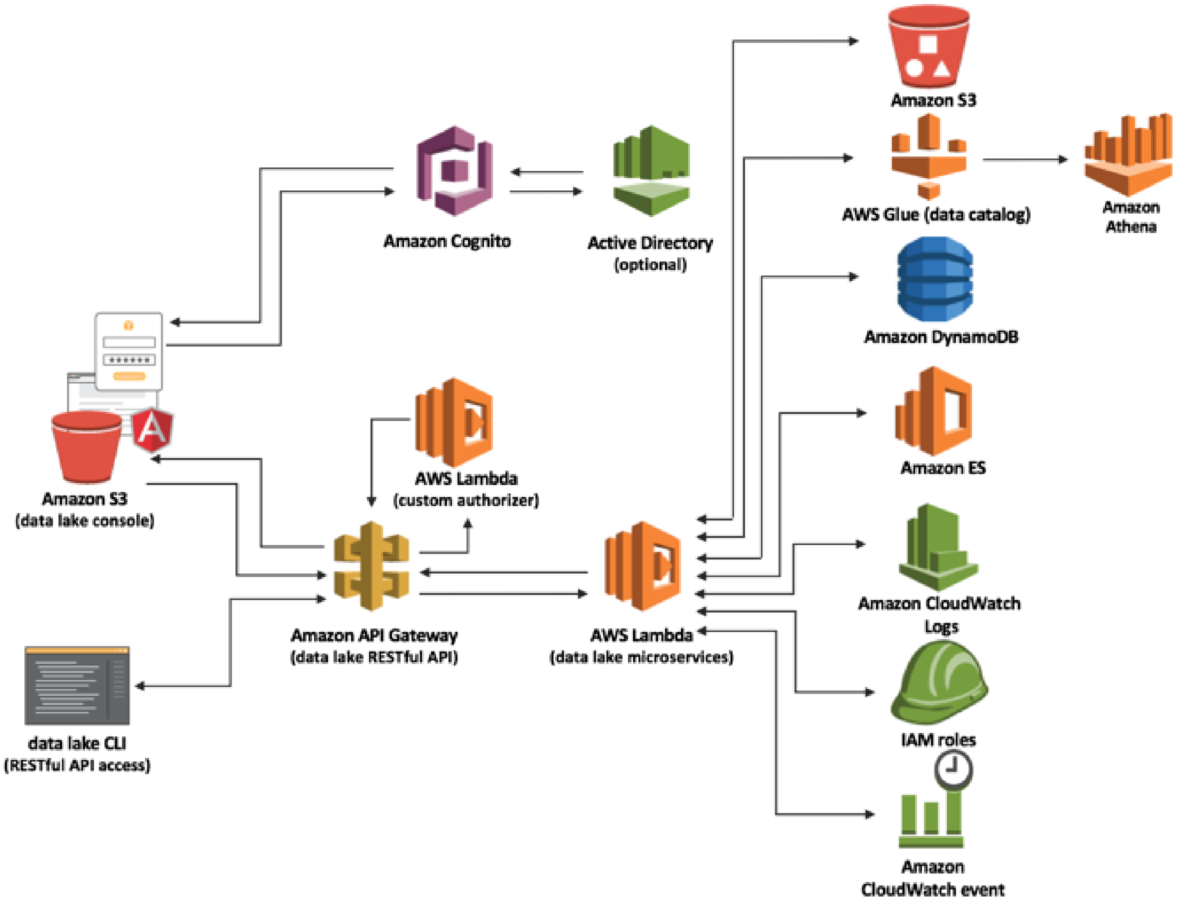

Amazon S3
AWS Glue (data catalog)
Amazon Athena
Amazon Cognito
Active Directory (optional)
Amazon DynamoDB
Amazon ES
AWS Lambda (custom authorizer)
Amazon S3 (data lake console)
Amazon API Gateway (data lake RESTful API)
AWS Lambda (data lake microservices)
Amazon CloudWatch Logs
IAM roles
data lake CLI (RESTful API access)
Amazon CloudWatch event

Figure 1: Data lake solution architecture on AWS

The solution uses AWS CloudFormation to deploy the infrastructure components supporting this data lake reference implementation. At its core, this solution implements a data lake API, which leverages Amazon API Gateway to provide access to data lake microservices, (AWS Lambda functions). These microservices provide the business logic to create data packages, upload data, search for existing packages, add interesting data to a cart, generate data manifests, and perform administrative functions. These microservices interact with Amazon S3, AWS Glue, Amazon Athena, Amazon DynamoDB, Amazon ES, and Amazon CloudWatch Logs to provide data storage, management, and audit functions.

The solution creates a data lake console and deploys it into an Amazon S3 bucket configured for static website hosting, and configures an Amazon CloudFront distribution to be used as the solution's console entrypoint. During initial configuration, the solution also creates a default administrator role and sends an access invite to a customer-specified email address. Note that if you deploy the federated stack, you must manually create user and admin groups. For more information, see Appendix A.

The solution uses an Amazon Cognito user pool to manage user access to the console and the data lake API. See Appendix B for detailed information on each of the solutions components.

Solution Features

This data lake solution provides the following features:

- **Data lake reference implementation:** Leverage this data lake solution out-of-the-box, or as a reference implementation that you can customize to meet unique data management, search, and processing needs.
- **User interface:** The solution automatically creates an intuitive, web-based console UI hosted on Amazon S3 and delivered by Amazon CloudFront. Access the console to easily manage data lake users, data lake policies, add or remove data packages, search data packages, and create manifests of datasets for additional analysis.
- **Command line interface:** Use the provided CLI or API to easily automate data lake activities or integrate this solution into existing data automation for dataset ingress, egress, and analysis.
- **Managed storage layer:** Secure and manage the storage and retrieval of data in a managed Amazon S3 bucket and use a solution-specific AWS Key Management Service (AWS KMS) key to encrypt data at rest.
- **Data access flexibility:** Leverage pre-signed Amazon S3 URLs or use an appropriate AWS Identity and Access Management (IAM) role for controlled yet direct access to datasets in Amazon S3.
- **Data transformation and analysis:** Upload datasets with searchable metadata that integrates with AWS Glue and Amazon Athena to transform and analyze the data.
- **Federation sign in:** Optionally, you can enable users to sign in through a SAML identity provider (IdP) such as Microsoft Active Directory Federation Services (AD FS).

AWS CloudFormation Template

This solution uses AWS CloudFormation to automate the deployment of the data lake solution on the AWS Cloud. It includes the following AWS CloudFormation template, which you can download before deployment:

View template

data-lake-deploy.template: Use this template to launch the data lake solution and all associated components. The default configuration deploys built-in authentication, authorization and user/group management. You can also customize the template based on your specific needs. This template, in turn, launches the following nested stacks:

- **data-lake-storage.template:** This template deploys the Amazon S3, Amazon Elasticsearch Service, and Amazon DynamoDB components of the solution.
- **data-lake-services.template:** This template deploys the AWS Lambda microservices and the necessary IAM roles and policies. In addition, it deploys the AWS KMS resources for the solution.
- **data-lake-api.template:** This template deploys the Amazon API Gateway resources.

Automated Deployment

Before you launch the automated deployment, please review the architecture, configuration, network security, and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy the data lake solution into your account.

Time to deploy: Approximately 30 minutes

What We'll Cover

The procedure for deploying this architecture on AWS consists of the following steps. For detailed instructions, follow the links for each step.

Step 1. Launch the stack

- Launch the AWS CloudFormation template into your AWS account.
- Enter values for required parameters: **Stack Name, Administrator Name**, **Administrator Email**, **Cognito Domain**
- Review the other template parameters and adjust if necessary.

Step 2. Log in to the Data Lake Console

- Log in with the URL and temporary password sent to the Administrator email.
- Review the solution's online guide.

Step 1. Launch the Stack

The AWS CloudFormation template automatically deploys the data lake solution on the AWS Cloud.

> **Note**: You are responsible for the cost of the AWS services used while running this solution. See the Cost section for more details. For full details, see the pricing webpage for each AWS service you will be using in this solution.

1. Log in to the AWS Management Console and click the button to the right to launch the `data-lake-deploy` AWS CloudFormation template.

Launch Solution

You can also download the template as a starting point for your own implementation.

2. The template is launched in the US East (N. Virginia) Region by default. To launch the data lake solution in a different AWS Region, use the region selector in the console navigation bar.

> **Note**: This solution uses Amazon Cognito, Amazon Athena, and AWS Glue which are currently available in specific AWS Regions only. Therefore, you must launch this solution in an AWS Region where these services are available.[1]

3. On the **Select Template** page, verify that you selected the correct template and choose **Next**.
4. On the **Specify Details** page, assign a name to your data lake solution stack.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following default values.

Parameter	Default	Description
Administrator Name	*<Requires input>*	The user name for the initial solution Administrator. After the solution is deployed, this Administrator can create and manage other users, including additional Administrators.
Administrator Email	*<Requires input>*	A valid email associated with the Administrator user
Cognito Domain	*<Requires input>*	Choose an available domain prefix for your Amazon Cognito hosted domain. The solution uses Amazon Cognito to offer user name and password protection for solution's Kibana. Defining a domain name for the user pool is a pre-requirement for that.

6. Choose **Next**.
7. On the **Options** page, you can specify tags (key-value pairs) for resources in your stack and set additional options, and then choose **Next**.
8. On the **Review** page, review and confirm the settings. Be sure to check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources with custom names.
9. Choose **Create** to deploy the stack.

   You can view the status of the stack in the AWS CloudFormation console in the **Status** column. After the stack launches, the three nested stacks will be launched in the same AWS Region. Once all of the stacks and stack resources have successfully launched, you will see the message CREATE_COMPLETE. This can take 30 minutes or longer.

[1] For the most current service availability by AWS Region, see https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services/

Step 2. Log in to the Data Lake Console

After the data lake stack launch completes the Administrator will receive an email that contains the URL to the data lake console and a temporary password.

> **Note:** This email will be sent from *no-reply@verificationemail.com*. Check your email configuration to make sure you do not block or filter emails from this domain.

1. Click the link in the email to open the solution console, and then log in with your email address and the temporary password.
2. You will be prompted to set a new password, and then you will be signed in to the console.
3. In the top navigation bar, choose **Support** to open the online guide.

   Explore the guide subsections (**User Guide**, **Admin Guide**, and **CLI**) for specific instructions and examples.

Security

The AWS Cloud provides a scalable, highly reliable platform that helps customers deploy applications and data quickly and securely. When you build systems on AWS infrastructure, security responsibilities are shared between you and AWS, which can reduce your operational burden. For more information about security on AWS, visit the AWS Security Center.

User Authorization

Authorized users access the data lake using the solution-generated console, the data lake CLI, or direct calls to the data lake APIs. Users log in to the data lake console with their user name (by default, their email) and password. Authentication to the console is managed in an Amazon Cognito user pool.

Requests to the data lake API are HTTPS based and must be signed with an access key (access key and secret access key combination) to confirm the user's identity. Administrators can grant API access on an individual user basis. If a user is granted API access, an access key is generated to identify that user's calls to the data lake API. Each user has the ability to generate their own secret access keys to allow them to work with the data lake CLI or make direct API calls.

See Appendix B for additional component-level security information.

Additional Resources

AWS service documentation

- AWS CloudFormation
- AWS Lambda
- Amazon S3
- Amazon CloudFront
- Amazon DynamoDB
- Amazon API Gateway
- Amazon Athena
- Amazon Cognito
- Amazon Elasticsearch Service
- AWS Key Management Service
- AWS Glue
- Amazon CloudWatch

AWS webpages

- What is a Data Lake?
- Big Data on AWS
- AWS Answers: Data Lakes on AWS

Appendix A: Federated Template

For customers who want to integrate with their existing SAML identity provider such Microsoft Active Directory, this data lake solution includes another AWS CloudFormation template that deploys the same workflow with Active Directory (AD) Federation configuration.

AWS CloudFormation Template

This solution includes the following AWS CloudFormation template, which you can download before deployment:

View template

data-lake-deploy-federated.template: Use this template to launch a version of the solution that is ready to integrate with your existing SAML identity provider such Microsoft Active Directory.

This template, in turn, launches the following nested stacks:

- **data-lake-storage.template:** This template deploys the Amazon S3, Amazon Elasticsearch Service, and Amazon DynamoDB components of the solution.

- **data-lake-services.template:** This template deploys the AWS Lambda microservices and the necessary IAM roles and policies. In addition, it deploys the AWS KMS resources for the solution.
- **data-lake-api.template:** This template deploys the Amazon API Gateway resources.

Automated Deployment

Before you launch the automated deployment, please review the architecture, configuration, network security, and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy the data lake solution with AD Federation into your account.

Time to deploy: Approximately 30 minutes

What We'll Cover

The procedure for deploying this architecture on AWS consists of the following steps. For detailed instructions, follow the links for each step.

Step 1. Launch the Stack

- Launch the AWS CloudFormation template into your AWS account.
- Enter values for required parameters: **Stack Name, Cognito Domain, AD FS Hostname**
- Review the other template parameters and adjust if necessary.

Step 2. Complete AD Federation

- Manually configure the AD Federation.

Step 1. Launch the Stack

> **Note**: You are responsible for the cost of the AWS services used while running this solution. See the Cost section for more details. For full details, see the pricing webpage for each AWS service you will be using in this solution.

1. Log in to the AWS Management Console and click the button to the right to launch the `data-lake-deploy-federated` AWS CloudFormation template.
   You can also download the template as a starting point for your own implementation.

   Launch Solution

2. The template is launched in the US East (N. Virginia) Region by default. To launch the data lake solution in a different AWS Region, use the region selector in the console navigation bar.

> **Note**: This solution uses Amazon Cognito Amazon Athena, and AWS Glue which are currently available in specific AWS Regions only. Therefore, you must launch this solution in an AWS Region where these services are available. [2]

3. On the **Select Template** page, verify that you selected the correct template and choose **Next**.
4. On the **Specify Details** page, assign a name to your data lake solution stack.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following default values.

Parameter	Default	Description
Cognito Domain	*<Requires input>*	Choose an available domain prefix for your Amazon Cognito hosted domain. The solution uses Amazon Cognito to offer user name and password protection for solution's Kibana. Defining a domain name for the user pool is a pre-requirement for that.
AD FS Hostname	*<Requires input>*	Insert the hostname of your AD FS endpoint.

6. Choose **Next**.
7. On the **Options** page, you can specify tags (key-value pairs) for resources in your stack and set additional options, and then choose **Next**.
8. On the **Review** page, review and confirm the settings. Be sure to check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources with custom names.
9. Choose **Create** to deploy the stack.

   You can view the status of the stack in the AWS CloudFormation console in the **Status** column. After the stack launches, the three nested stacks will be launched in the same AWS Region. Once all of the stacks and stack resources have successfully launched, you will see the message CREATE_COMPLETE. This can take 30 minutes or longer.

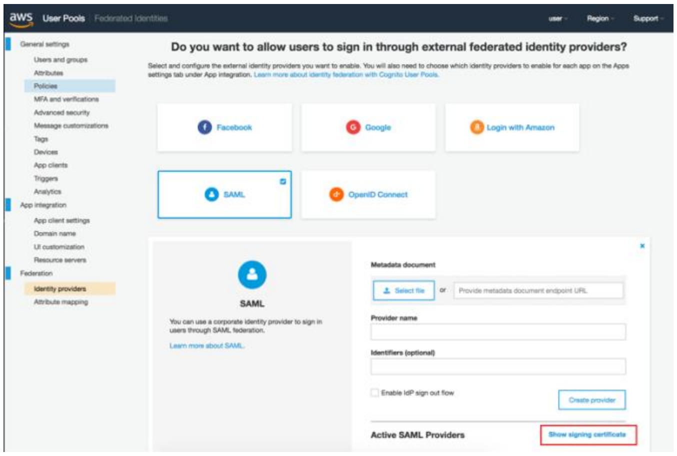
Step 2. Complete AD Federation

After the data lake stack launch completes, you must complete the AD Federation configuration. Note that this step is required only if you deployed the federated template.

1. Log into the AWS Management Console and navigate to the stack **Outputs** tab.
2. Note the values of the **RelyingPartyURL**, **RelyingPartyTrustedIdentifier**, and **LogoutTrustedURL** keys.
3. Navigate to the **IdentityProvidersUrl** link.

[2] For the most current service availability by AWS Region, see https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services/

4. On the **Federation** console, in the **Identity providers** section under **Active SAML Providers**, select **Show signing certificate**.



Do you want to allow users to sign in through external federated identity providers?

Select and configure the external identity providers you want to enable. You will also need to choose which identity providers to enable for each app on the Apps settings tab under App integration. Learn more about identity federation with Cognito User Pools.

You can use a corporate identity provider to sign in users through SAML federation.

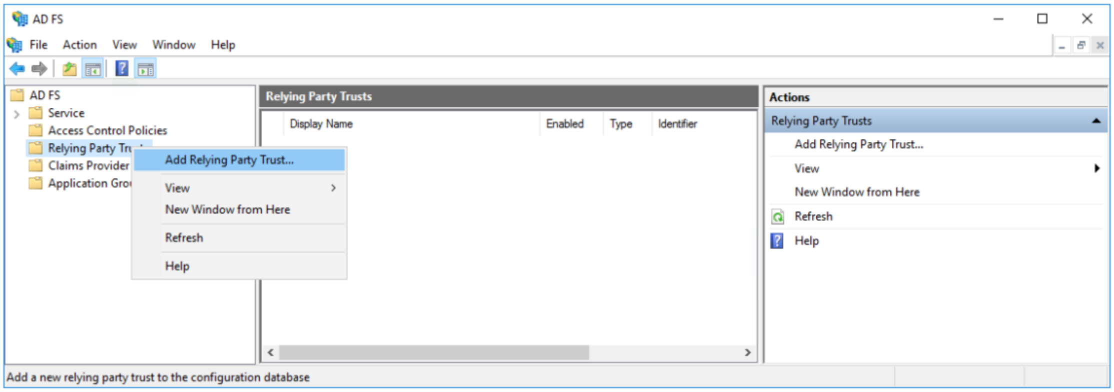
Learn more about SAML.

5. Copy the certificate containing the public key. This key will be used by the identity provider to verify the signed logout request to a .cer file (for example, `datalake.cer`) on your AD FS server.

Add Amazon Cognito as a relying party in AD FS:

AD FS federation occurs with the participation of two parties; the identity or claims provider (Active Directory) and the relying party (Cognito). The relying party is a federation partner that is represented by a claims provider trust in the federation service. Use the following procedure to configure a new relying party in Active Directory Federation Services:

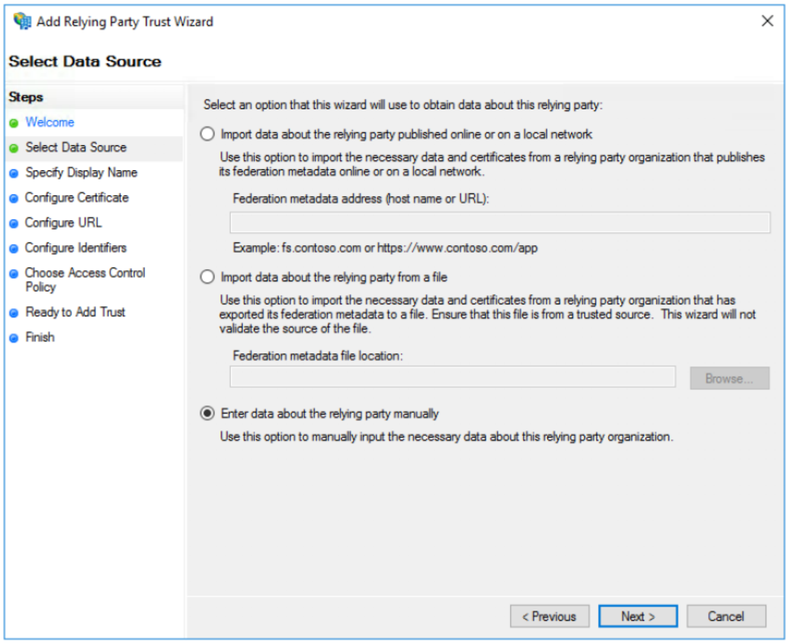
1. In the AD FS Management Console, right-click **AD FS**, and select **Add Relying Party Trust**.



Add a new relying party trust to the configuration database

2. In the Add Relying Party Trust wizard, select **Start**.

3. Select **Select Data Source**. Then, select the **Enter data about the relying party manually** radio button, and choose **Next**.



Select Data Source

Select an option that this wizard will use to obtain data about this relying party:

○ Import data about the relying party published online or on a local network

Use this option to import the necessary data and certificates from a relying party organization that publishes its federation metadata online or on a local network.

Federation metadata address (host name or URL):

Example: fs.contoso.com or https://www.contoso.com/app

○ Import data about the relying party from a file

Use this option to import the necessary data and certificates from a relying party organization that has exported its federation metadata to a file. Ensure that this file is from a trusted source. This wizard will not validate the source of the file.

Federation metadata file location:

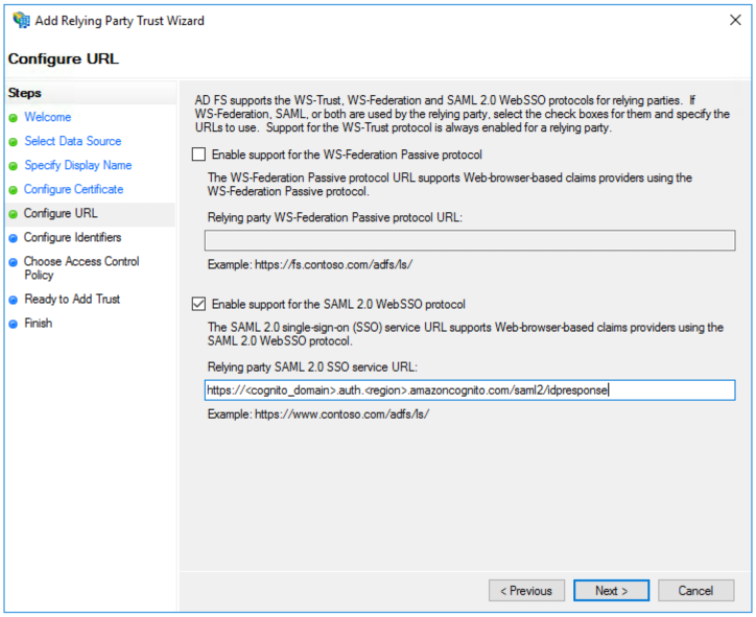
● Enter data about the relying party manually

Use this option to manually input the necessary data about this relying party organization.

4. Select **Specify Display Name** and set a Display Name (For example, Data Lake Solution on AWS).

5. Select **Configure Certificate** and select **Next** to accept the default values.

6. Select **Configure URL**. Then, select **Enable support for the SAML 2.0 WebSSO protocol** and set the URL replying party (use the value you noted from **RelyingPartyURL** output parameter). Then, select **Next**.



Configure URL

AD FS supports the WS-Trust, WS-Federation and SAML 2.0 WebSSO protocols for relying parties. If WS-Federation, SAML, or both are used by the relying party, select the check boxes for them and specify the URLs to use. Support for the WS-Trust protocol is always enabled for a relying party.

□ Enable support for the WS-Federation Passive protocol

The WS-Federation Passive protocol URL supports Web-browser-based claims providers using the WS-Federation Passive protocol.

Relying party WS-Federation Passive protocol URL:

Example: https://fs.contoso.com/adfs/ls/

☑ Enable support for the SAML 2.0 WebSSO protocol

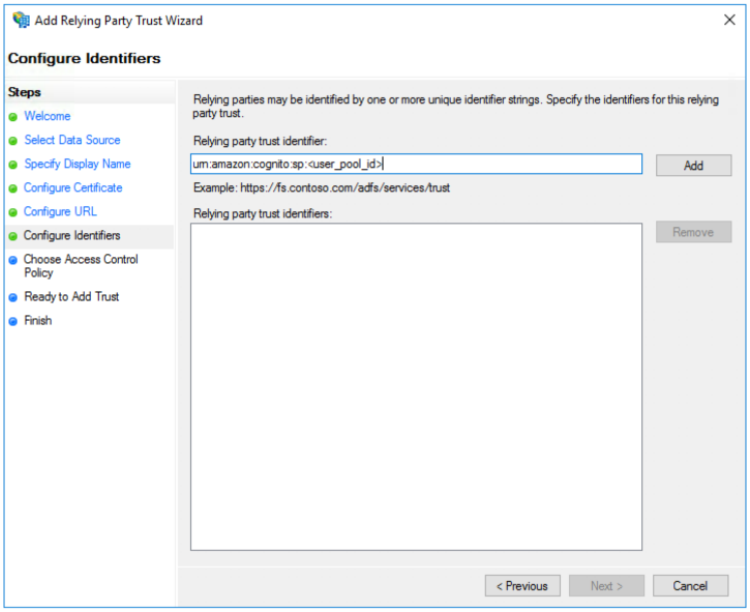
The SAML 2.0 single-sign-on (SSO) service URL supports Web-browser-based claims providers using the SAML 2.0 WebSSO protocol.

Relying party SAML 2.0 SSO service URL:

https://<cognito_domain>.auth.<region>.amazoncognito.com/saml2/idpresponse

Example: https://www.contoso.com/adfs/ls/

7. Select **Configure Identifiers** and set the **Relying party trusted identifier** (use the value you noted from **RelyingPartyTrustedIdentifier** output parameter). Then, select **Next**.



Configure Identifiers

Relying parties may be identified by one or more unique identifier strings. Specify the identifiers for this relying party trust.

Relying party trust identifier: urn:amazon:cognito:sp:<user_pool_id>

Example: https://fs.contoso.com/adfs/services/trust

Relying party trust identifiers:

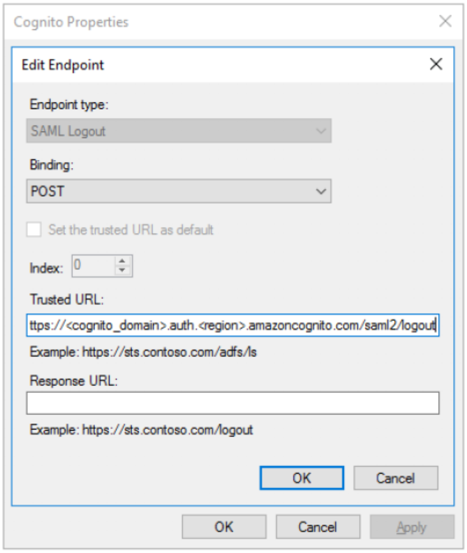
8. Select **Next** until you reach the end of the wizard.

Enable Sign Out Flow

Enabling this flow sends a signed logout request to the Active Directory when logout is called. The AD will process the signed logout request and logout your user from the Amazon Cognito session. Note that the AD FS server expects a signed logout request, you must configure the signing certificate provided by Amazon Cognito with your AD FS.

Use the following procedure to configure this endpoint for consuming logout responses from your Active Directory Federation Services:

1. In the AD FS Management Console, double-click on the **relying party**, select the **Endpoints** tab, and select **Add SAML**.
2. Set the **Endpoint type** to **SAML Logout**; **Binding** to **POST**, and set the **Trusted URL** value (use the value you noted from **LogoutTrustedURL** output parameter). Then, select **OK**.



Cognito Properties

Edit Endpoint

Endpoint type: SAML Logout

Binding: POST

□ Set the trusted URL as default

Index: 0

Trusted URL: ttps://<cognito_domain>.auth.<region>.amazoncognito.com/saml2/logout

Example: https://sts.contoso.com/adfs/ls

Response URL:

Example: https://sts.contoso.com/logout

3. Select the **Signature** tab, and add the certificate you copied from **Federation** console (`datalake.cer`), and select **OK**.

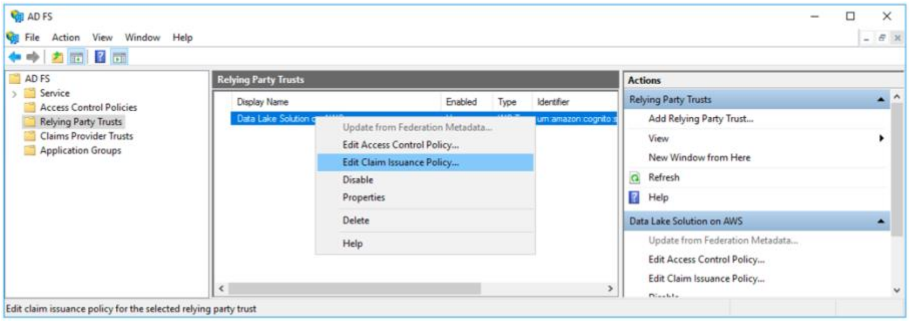
Custom Claim Rules

Microsoft AD FS uses Claims Rule Language to issue and transform claims between claims providers and relying parties. A claim is information about a user from a trusted source. The trusted source is asserting that the information is true, and that source has authenticated the user. The claims provider is the source of the claim. This can be information pulled from an attribute store such as Active Directory (AD). Amazon Cognito user pools support SAML 2.0 federation with post-binding endpoints. This eliminates the need for your app to retrieve or parse SAML assertion responses, because the user pool directly receives the SAML response from your identity provider via a user agent. Your user pool acts as a service provider on behalf of your application.

Use the following procedure to configure a new relying party in Active Directory Federation Services:

Note that this procedure configures all members of the **DataLake Admins** groups **Role** outgoing claim type as **Admin**.

1. In the AD FS Management Console, right-click on the **relying party**, and select **Edit Claim Issuance Policy**.


AD FS
File Action View Window Help
AD FS
Service
Access Control Policies
Relying Party Trusts
Claims Provider Trusts
Application Groups
Relying Party Trusts
Display Name
Enabled
Type
Identifier
Data Lake Solution on AWS
urn:amazon:cognito
Update from Federation Metadata...
Edit Access Control Policy...
Edit Claim Issuance Policy...
Disable
Properties
Delete
Help
Actions
Relying Party Trusts
Add Relying Party Trust...
View
New Window from Here
Refresh
Help
Data Lake Solution on AWS
Update from Federation Metadata...
Edit Access Control Policy...
Edit Claim Issuance Policy...

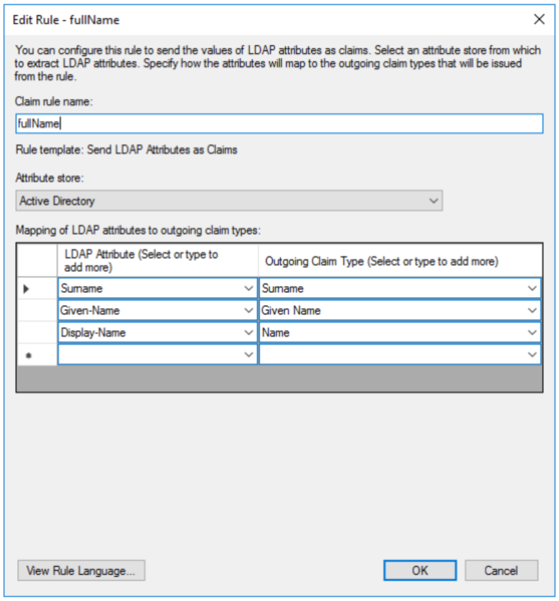
Edit claim issuance policy for the selected relying party trust

2. Specify a **claim rule name**.

3. Select **Attribute store**. Note that this can be Active Directory if your users are in Active Directory.

4. Map an LDAP Attribute (For example, `E-Mail-Address`) to Outgoing Claim Type (For example, `E-Mail Address`).

   Make sure that your AD FS populates the following required attributes for your user pool in the SAML assertion: `fullName`, `email`, `nameId`, `groups`, and `isAdmin`.


Edit Rule - fullName

You can configure this rule to send the values of LDAP attributes as claims. Select an attribute store from which to extract LDAP attributes. Specify how the attributes will map to the outgoing claim types that will be issued from the rule.

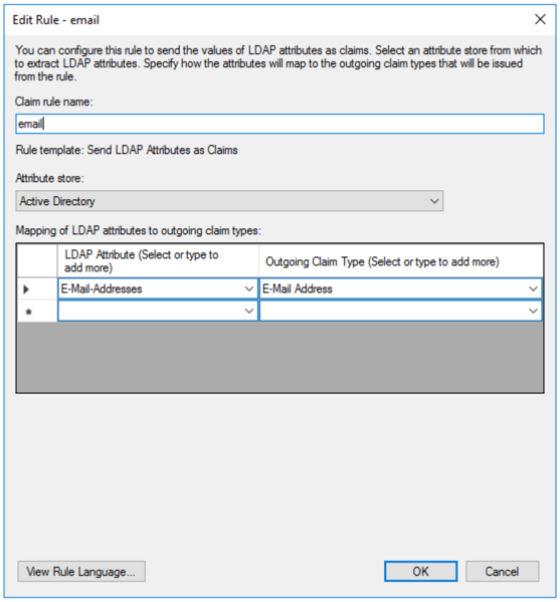
Claim rule name: fullName

Rule template: Send LDAP Attributes as Claims

Attribute store: Active Directory

Mapping of LDAP attributes to outgoing claim types:

LDAP Attribute (Select or type to add more)	Outgoing Claim Type (Select or type to add more)
Surname	Surname
Given-Name	Given Name
Display-Name	Name


Edit Rule - email

You can configure this rule to send the values of LDAP attributes as claims. Select an attribute store from which to extract LDAP attributes. Specify how the attributes will map to the outgoing claim types that will be issued from the rule.

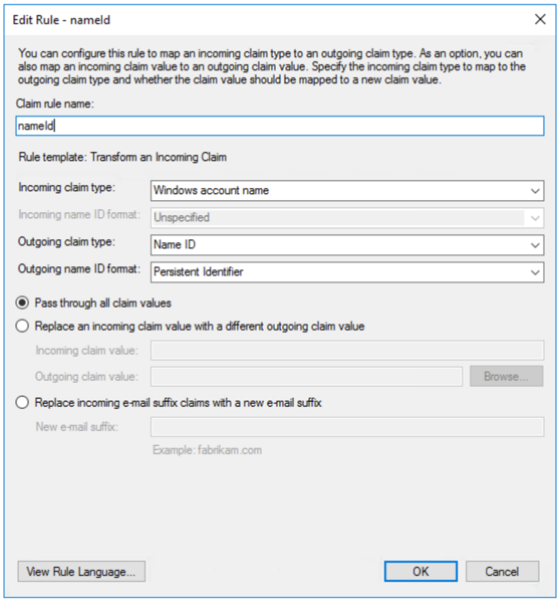
Claim rule name: email

Rule template: Send LDAP Attributes as Claims

Attribute store: Active Directory

Mapping of LDAP attributes to outgoing claim types:

LDAP Attribute (Select or type to add more)	Outgoing Claim Type (Select or type to add more)
E-Mail-Addresses	E-Mail Address



Edit Rule - nameId

You can configure this rule to map an incoming claim type to an outgoing claim type. As an option, you can also map an incoming claim value to an outgoing claim value. Specify the incoming claim type to map to the outgoing claim type and whether the claim value should be mapped to a new claim value.

Claim rule name: nameId

Rule template: Transform an Incoming Claim

Incoming claim type: Windows account name

Incoming name ID format: Unspecified

Outgoing claim type: Name ID

Outgoing name ID format: Persistent Identifier

● Pass through all claim values

○ Replace an incoming claim value with a different outgoing claim value

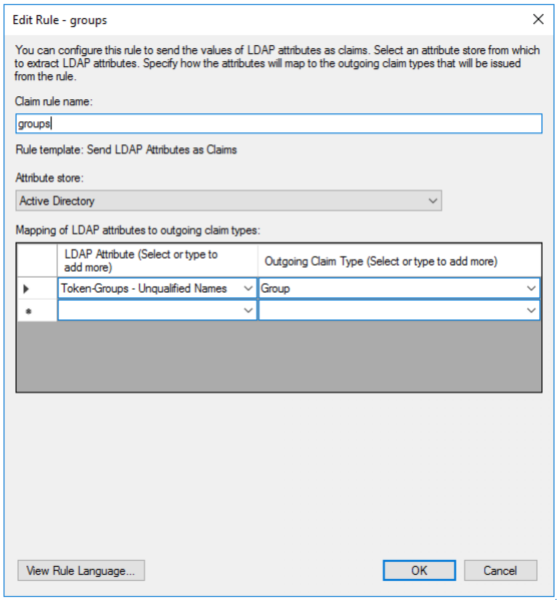
Incoming claim value:

Outgoing claim value:

○ Replace incoming e-mail suffix claims with a new e-mail suffix

New e-mail suffix:

Example: fabrikam.com



Edit Rule - groups

You can configure this rule to send the values of LDAP attributes as claims. Select an attribute store from which to extract LDAP attributes. Specify how the attributes will map to the outgoing claim types that will be issued from the rule.

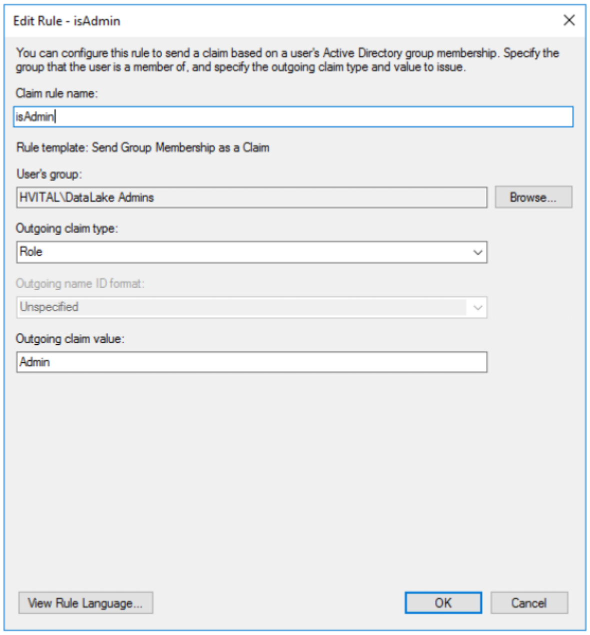
Claim rule name: groups

Rule template: Send LDAP Attributes as Claims

Attribute store: Active Directory

Mapping of LDAP attributes to outgoing claim types:

LDAP Attribute (Select or type to add more)	Outgoing Claim Type (Select or type to add more)
Token-Groups - Unqualified Names	Group



Edit Rule - isAdmin

You can configure this rule to send a claim based on a user's Active Directory group membership. Specify the group that the user is a member of, and specify the outgoing claim type and value to issue.

Claim rule name: isAdmin

Rule template: Send Group Membership as a Claim

User's group: HVITAL\DataLake Admins

Outgoing claim type: Role

Outgoing name ID format: Unspecified

Outgoing claim value: Admin

5. Log into the AWS Management Console, navigate to the stack **Outputs** tab.

6. Select the **Value** of the **ConsoleUrl** key, you will be redirected to the AD FS Management Console.

Appendix B: Solution Components

AWS KMS Key

The data lake AWS KMS key (alias: **datalake**) is created to provide encryption of all dataset objects that the solution owns and stores in Amazon S3. Additionally, the AWS KMS key is used to encrypt the secret access key in each user's Amazon Cognito user pool record for API access to the data lake.

Amazon CloudFront

The solution configures an Amazon CloudFront distribution to serve HTTPS requests for the data lake console.

Amazon S3

The solution uses a default Amazon S3 bucket to store datasets and manifest files associated with packages that users upload to the data lake. Additionally, the bucket stores the manifest files generated for a user when they check out their cart, which is a collection of packages. All access to this bucket (get and put actions from the package and manifest microservices) is controlled via signed URLs. All objects stored in this bucket are encrypted using the data lake AWS KMS key.

A second Amazon S3 bucket hosts the data lake console. This console is a static website that uses Amazon Cognito for user authentication. End users do not have direct access to the S3 endpoint. All access should be done via the Amazon CloudFront distribution.

Amazon Athena with AWS Glue

This solution automatically configures an AWS Glue crawler within each data package and schedules a daily scan to keep track of the changes. The crawlers crawl through your datasets and inspect portions of it to infer a data schema and persist the output as one or more metadata tables that are defined in your AWS Glue Data Catalog.

Once created, this catalog provides a unified metadata repository across a variety of data sources and formats, integrating with Amazon Athena and Amazon Redshift Spectrum to interactively query and analyze data directly in your data lake, and with Amazon EMR, AWS Glue extract, transform, and load (ETL) jobs and any application compatible with the Apache Hive data warehouse so you can categorize, clean, enrich, and move your data.

Amazon Cognito User Pool

The data lake console is secured for user access with Amazon Cognito and provides an administrative interface for managing data lake users through integration with Amazon Cognito user pools. Only Administrators can create users and groups, once users are created the solution will automatically send an invitation to the user to join the data lake. Note that if you use the federated template, all administrative tasks should be done on the AD server. When an Administrator creates a new user, he/she will assign the user one of the following roles, with the associated permissions:

- **Member:** The member role can perform non-administrative actions within the data lake. These actions include the following:
    - View and search packages if the owner or visible package in a member group
    - Add, remove, and generate manifests for packages in their cart
    - Create, update, and delete packages they created
    - Create and update metadata on the packages they created
    - Add and remove datasets from the packages they created
    - View their data lake profile and API access information
    - Generate a secret access key if an Administrator has granted them API access
- **Admin:** The admin role has full access to the data lake. The admin role can perform the following actions in addition to the member role actions:
    - Create user invitations and assign users to one or more groups
    - Create, update, delete groups
    - Update, disable, enable, and delete data lake users
    - Assign, delete, and reassign users to groups
    - Create, revoke, enable, and disable a user's API access
    - Update data lake settings
    - Create, update, and delete governance settings

Data Lake API and Microservices

The data lake API receives requests via HTTPS. When an API request is made, Amazon API Gateway leverages a custom authorizer (AWS Lambda function) to ensure that all requests are authorized.

The data lake microservices is a series of AWS Lambda functions that provide the business logic and data access layer for all data lake operations. Each AWS Lambda function assumes an IAM role with least privilege access (minimum permissions necessary) to perform its designated functions. The following sections outline each data lake microservice.

Admin Microservice

The `data-lake-admin-service` is an AWS Lambda function that processes data lake API requests sent to the `/admin/*` endpoints. The admin microservice handles all administrative services including user and group management, general settings, governance settings, API keys, and role authorization for all operations within the data lake.

Cart Microservice

The `data-lake-cart-service` is an AWS Lambda function that processes data lake API requests sent to the `/cart/*` endpoints. The cart microservice handles all cart operations including item lists, adding items, removing items, and generating manifests for user carts.

Manifest Microservice

The `data-lake-manifest-service` is an AWS Lambda function that manages import and export of manifest files. The manifest microservice uploads import manifest files, which allows existing Amazon S3 content to be bulk imported into a package. It also generates export manifest files for each package in a user's cart at checkout.

Package Microservice

The `data-lake-package-service` is an AWS Lambda function that processes data lake API requests sent to `/packages/*` endpoints. The package microservice handles all package operations including list, add package, remove package, update package, list metadata, add metadata, update metadata, list datasets, add dataset, remove dataset, process manifest, run AWS Glue on-demand crawler, list and access AWS Glue tables, and view dataset on Amazon Athena.

Search Microservice

The `data-lake-search-service` is an AWS Lambda function that process data lake API requests sent to `/search/*` endpoints. The search microservice handles all search operations including query, index document, and remove indexed document.

Profile Microservice

The `data-lake-profile-service` is an AWS Lambda function that processes data lake API requests sent to `/profile/*` endpoints. The profile microservice handles all profile operations for data lake users, including get and generate secret access key.

Logging Microservice

The `data-lake-logging-service` is an AWS Lambda function that interfaces between the data lake microservices and Amazon CloudWatch Logs. Each microservice sends operations and access events to the logging service, which records the events in Amazon CloudWatch Logs. You can access this log (*datalake/audit-log*) in the CloudWatch console.

Amazon DynamoDB Tables

The data lake solution uses Amazon DynamoDB tables to persist metadata for the data packages, settings, and user cart items. The following tables are provisioned during deployment and only accessed via the data lake microservices:

- **data-lake-packages**: persistent store for data package title and description, and list of groups that can access the package
- **data-lake-metadata:** persistent store for metadata tag values associated with packages
- **data-lake-datasets:** persistent store for dataset pointers to Amazon S3 objects
- **data-lake-cart:** persistent store for user cart items
- **data-lake-keys:** persistent store for user access key ID references
- **data-lake-settings:** persistent store for data lake configuration and governance settings

Amazon Elasticsearch Service Cluster

The solution uses an Amazon Elasticsearch Service cluster to index data lake package data for searching. The cluster is accessible only by the search microservice and via Cognito authentication.

Appendix C: Collection of Anonymous Data

This solution includes an option to send anonymous usage data to AWS. We use this data to better understand how customers use this solution to improve the services and products that we offer. When enabled, the following information is collected and sent to AWS:

- **Solution ID:** The AWS solution identifier
- **Unique ID (UUID):** Randomly generated, unique identifier for each data lake solution deployment
- **Timestamp:** Data-collection timestamp
- **Cluster Size:** Size of the Amazon Elasticsearch cluster the solution will deploy

Note that AWS will own the data gathered via this survey. Data collection will be subject to the AWS Privacy Policy. To opt out of this feature, modify the AWS CloudFormation template mapping section as follows:

```
Solution:
   Data:
      SendAnonymousUsageData: "Yes"
```

to

```
Solution:
   Data:
      SendAnonymousUsageData: "No"
```

Send Us Feedback

We welcome your questions and comments. Please post your feedback on the AWS Solutions Discussion Forum.

You can visit our GitHub repository to download the templates and scripts for this solution, and to share your customizations with others.

Document Revisions

Date	Change	In sections
November 2016	Initial release	-
June 2018	Granular permissions, integration of AWS Glue and Amazon Athena	Overview; Cost; Solution Features; Appendix B; Appendix C
September 2018	Active Directory Federation	Overview; Architecture Overview; Solution Features; Appendix A

Notices

This document is provided for informational purposes only. It represents AWS's current product offerings and practices as of the date of issue of this document, which are subject to change without notice. Customers are responsible for making their own independent assessment of the information in this document and any use of AWS's products or services, each of which is provided "as is" without warranty of any kind, whether express or implied. This document does not create any warranties, representations, contractual commitments, conditions or assurances from AWS, its affiliates, suppliers or licensors. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.