Amazon Redshift Security Best Practices

First published January 2025

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers, or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. 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.

Contents

Abstract

Data is a key differentiator, critical to driving business value. But to achieve this, the right data must be available to the right people and systems at the right time. Data must be protected from external and internal actors who might maliciously or accidentally misuse information. Amazon Redshift is a cloud-based data warehouse that offers a comprehensive set of analytics capabilities to enable data driven decision-making. Amazon Redshift offers broad security related configurations and controls to help ensure information is appropriately protected. This whitepaper aims to inform and guide technical leadership, architects, data engineers and developers to understand available options and implement relevant controls to match security requirements and provide a security hardened posture.

Are you well-architected?

The AWS Well-Architected Framework helps you understand the pros and cons of the decisions you make when building systems in the cloud. The six pillars of the framework allow you to learn architectural best practices for designing reliable, secure, efficient, cost-effective, and sustainable systems and ongoing operational excellence. Using the AWS Well-Architected Tool, available at no additional charge in the AWS Management Console, you can review your workloads against these best practices by answering a set of questions for each pillar.
Using the Data Analytics Lens, we describe a collection of customer-proven best practices for designing well-architected analytics workloads.
For more expert guidance and best practices for your cloud architecture—reference architecture deployments, diagrams, and whitepapers—see the AWS Architecture Center.

A note on the AWS Shared Responsibility Model

When considering cloud-based workloads, the AWS Shared Responsibility Model distinguishes between security of the cloud and security in the cloud.
It is the responsibility of AWS to protect infrastructure running AWS services (*security of the cloud*), including Amazon Redshift. As such, AWS will handle operating system updates, database patching, firewall updates, and so on. These operational activities happen automatically and behind the scenes so that you can focus on your business intelligence applications running on Amazon Redshift, rather than low-level security and management tasks.
In an Amazon Redshift provisioned cluster, these activities take place in a configurable maintenance window. For a serverless cluster, it is transparent.
If you use the browser-based Amazon Redshift Query Editor v2, this is an AWS managed application and again, patches and updates are conducted by AWS so that you can use the tool with confidence and without having to conduct updates yourself.
Because Amazon Redshift is a managed service, security and management related tasks of the underlying hardware are taken care of for you. When running your workloads however, responsibility shifts to you under *security in the cloud*. With Amazon Redshift, this means considering your specific factors such as data sensitivity and any related compliance requirements. It is your responsibility to ensure that you meet these requirements while using Amazon Redshift for your workloads, and the security related practices outlined here should help you to do so.

Introduction

Protecting analytics infrastructure, managing access to data, and maintaining visibility over user actions are just some of the key security challenges when trying to use data for data-driven decision making. Companies need to govern data to empower employees but keep data safe, confidential, and with judicious access at scale. For example, sales managers might need access to sales, customer and product related information to deliver key performance indicators (KPIs) and dashboards. Whereas a finance analyst might only need access to financial management information to provide accounts reporting.

Another consideration is the threat of malicious actors who are persistently threatening to steal data. Data warehouses, which bring together multiple sources of information to produce rich data sets, can be desirable targets because they contain comprehensive data valuable to the business.

Without strong security controls in place, not only could data be at risk to such external actors, but it also hinders the ability to empower internal teams with access to the data they need. This can lead to anti-patterns where data and related infrastructure are overly closed off, data silos emerge, and data-driven culture suffers.

Amazon Redshift is a fast, petabyte-scale, cloud data warehouse that tens of thousands of customers rely on to power their analytics workloads.

Using Amazon Redshift allows you to bring together data from many sources to produce meaningful insights with best-in-class security and compliance with SOC, PCI, FedRAMP, HIPAA and others. For more information, see Compliance validation for Amazon Redshift.

This whitepaper will equip technical leadership, architects, data analysts, and developers with the information needed to make sure that appropriate controls are put in place, while still enabling teams to access the right data and drive business value.

This whitepaper will cover four main aspects:

- Authentication and authorization – Making sure that user identities and authorizations are appropriately controlled when managing Amazon Redshift resources and accessing the data warehouse.
- Data access control – Authorization options to make sure users can access the data they need with relevant protections and constraints.
- Networking – Network design considerations to help ensure resources are secure.
- Monitoring best practice adherence – Methods to help monitor best practice adoption.

Because there are many layers to consider when discussing security in Amazon Redshift, the following diagram might help you understand the different controls available and how they relate to your overall architecture.

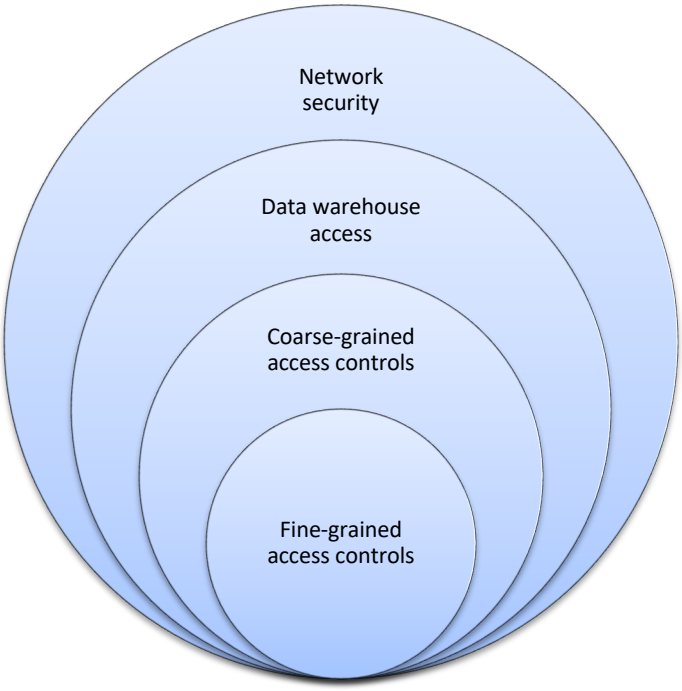

Figure 1 – The different layers of security controls.

Example scenarios

It is common for customers to require compliance with PCI, HIPAA, and SOC, among others, or with frameworks such as GDPR. While Amazon Redshift has been assessed as part of multiple compliance programs, you must implement processes and controls as part of your Amazon Redshift workloads. This whitepaper discusses many of the security controls that you might need to implement. See the following list for example scenarios along with references to locations in the whitepaper where you can find more information about specific requirements.

- **PCI (Payment Card Industry) compliance**
  - **Segregation of duties** – You can use AWS Identity and Access Management (IAM) to control access to Amazon Redshift resources, as discussed in Amazon Redshift resource . Similarly, authentication and authorizations in the data warehouse are discussed in Data warehouse .
  - **Protecting card holder data** – You can use encryption of your data at rest and in transit with Amazon Redshift. See Data .
  - **Restricting access to cardholder data by business need to know** – Amazon Redshift role-based access controls (RBAC) can help when assigning different sets of permissions to different users. See **Error! Reference source not found.**.
- GDPR **(General Data Protection Regulation) framework**

- Use RBAC controls to assist with data privacy by design. Use fine-grained access controls to restrict access to certain rows and columns of a table, depending on the user role and business need. See Fine-grained access control: Row and column level access policies.
- The use of data tokenization or masking can assist with limiting access to sensitive fields, such as personally identifiable information (PII). See Data tokenization and .

Authentication and authorization

Authentication verifies the identity of the user trying to access resources, while authorization determines what resources and actions that authenticated entity is permitted to access or perform based on defined policies and roles.
This section will focus on:

- **Amazon Redshift resource management** – Controlling access to Amazon Redshift resources when managing data warehouses.
- **Data warehouse access** – Methods for authenticating users to the data warehouse and granting specific authorizations to the user. The various methods and features within Amazon Redshift that control access to data, based on a user's authorization, will be discussed in Data access .

Amazon Redshift resource management

When managing Amazon Redshift resources, you might be required to create, update or delete Amazon Redshift provisioned clusters, Amazon Redshift Serverless namespaces and workgroups, manage access to Amazon Redshift Query Editor v2 (a browser based SQL editor for your Redshift data warehouses) or define other related configurations. This requires having relevant permissions to perform actions against corresponding Amazon Redshift resources. Because these processes involve managing AWS resources and configuration, the authentication and authorization around these processes is managed by how you sign in to or use AWS. This is different from the controls that live within the data warehouse, which determine which users can access and manage database objects—these controls will be discussed in the Data warehouse section.
There are different ways to sign in to AWS:

- As the AWS account root user.
- As an AWS Identity and Access Management (IAM) user.
- Assuming an IAM role.
- You can also have a single sign-on (SSO) experience as a federated identity by using credentials provided through an identity provider (IdP). When you access AWS using federation, you are indirectly assuming a role.

It is recommended to assume IAM roles rather than using IAM users when managing and accessing resources, because this helps ensure that temporary rather than long lived AWS credentials are used. This is a general recommendation for authorizing service use within your

AWS account and is not unique to using Amazon Redshift. You can learn more by reviewing additional information about using roles.

Furthermore, it can be useful to align specific IAM roles with specific job functions and scope permissions. This approach adheres to least privilege principles, among the IAM best practices in Security best practices in IAM.

For example, in the context of managing your Amazon Redshift resources, you might create an IAM role for a *data warehouse admin*, specifically for administration type tasks. This would have a relatively permissive set of allowed IAM policy actions, such as permissions to create, update, and delete Amazon Redshift resources, so that relevant activities can be carried out only by users who can assume that IAM role. A less permissive *data warehouse analyst* IAM role should be used to allow data warehouse users access to only see required resources, use the Amazon Redshift Query Editor v2, and other required permissions. This separation creates a clear distinction in permissions, which helps to protect against malicious or accidental changes. Further, this approach assists with audit and traceability, because actions can be tracked by IAM role to help determine who took a particular action. This can be done using AWS CloudTrail for auditing API calls. Additional IAM roles with appropriately scoped permissions can be introduced to match specific teams and role functions. Reviewing access patterns using AWS IAM Access Analyzer is also recommended so you can make sure that roles are relevant and well-maintained.

When building IAM policies to associate with IAM roles, use the principle of least privilege access. Only grant permissions that are required for the job function and refer to specific API actions in the IAM policy rather than using wildcard (*) identifiers, which might provide broader access than intended. Consider using resource constraints to give permissions only to the required Amazon Redshift resources, for example, restricting access to specific clusters, namespaces, or workgroups. Restricting this access will help prevent modification or deletion of the respective resources, but will not prevent data analysts connecting using Java Database Connectivity (JDBC) for example. For more information about using identity-based policies with Amazon Redshift, see Using identity-based policies (IAM policies) for Amazon Redshift. For more information about managing access permissions to Amazon Redshift resources, see Overview of managing access permissions to your Amazon Redshift resources.

It is recommended to use your IdP with AWS IAM Identity Center integration to provide an SSO experience to authenticate users and assume roles for authorization. The assumed role permissions would then determine the specific actions the user can take.

Data warehouse access

After Amazon Redshift resources have been set up, users can start accessing the data warehouse to begin loading, transforming, and querying data. When doing so, it is important to give users access only to the data that they need, but also to give a frictionless experience so that data-driven culture can flourish. This involves using features of Amazon Redshift to control the authentication and authorization process.

There are several ways to connect to a Redshift data warehouse, depending on the use case and preferred tooling. Common approaches include Amazon Redshift Query Editor v2, client

tools or application access using the Amazon Redshift driver and programmatic access including through the Amazon Redshift Data API. For more information, see Querying a database.
There are some differences between the available methods between provisioned clusters and serverless workgroups, but in general the available methods are:

- **IdP federation** –
  - Provide SSO with IAM Identity Center. Further connecting Amazon Redshift to IAM Identity Center for authentication and authorization enables identity propagation with other analytics services.
  - Use an external IdP to deliver an SSO experience.
- **IAM authentication** –Use IAM permissions to generate temporary database sign-in credentials.
- **Local users** – Authenticate using a database username and password.

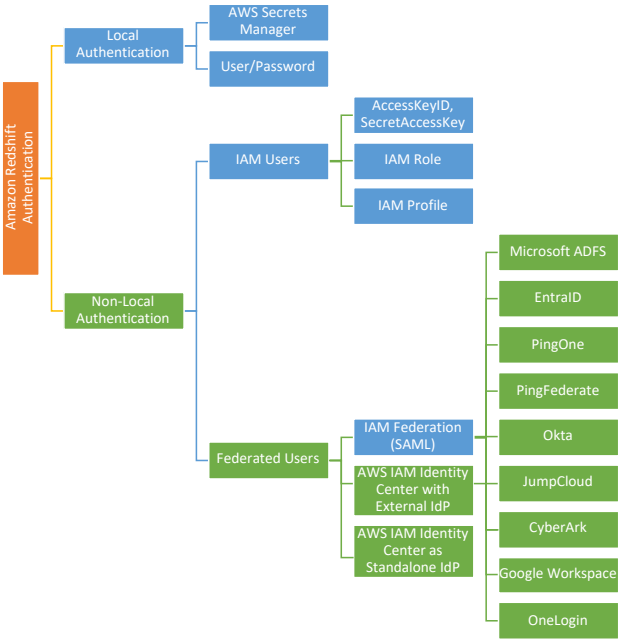

Figure 2: – Federated compared to local options for accessing the database. Green represents the recommended best practice approach

The various methods are shown in the preceding image, distinguishing between local authentication where user information is managed within the data warehouse, or the recommended non-local authentication using services such as IAM or IdP federation.

Identity provider federation

You can use IAM Identity Center or an external IdP to manage your user identities. Either approach uses the IdP for user authentication. Then, as the user connects to the data warehouse, they can be granted certain authorizations. This workflow can be achieved using an IAM Identity Center connection or native federation.

SSO with IAM Identity Center connection

It is a recommended best practice to use IAM Identity Center to provide an SSO experience for Amazon Redshift data warehouse users. This approach attaches Redshift roles to the user identity so you can use the RBAC capabilities of Amazon Redshift. Further, it allows identity propagation, meaning that identity information can be shared with compatible AWS services for a simplified experience.

IAM Identity Center can be used to manage your identities or it can be automatically synced with your existing IdP using the System for Cross-domain Identity Management (SCIM) v2.0 protocol. This means that you can have a single place for managing workforce identities in AWS. IAM Identity Center provides an SSO experience across analytics services such as Amazon Redshift Query Editor v2, Amazon QuickSight, and AWS Lake Formation.

A single IAM Identity Center instance can manage permissions across multiple AWS accounts without configuring each of your accounts manually. You can also configure IAM Identity Center to run alongside or replace single AWS account access management using IAM.

Amazon Redshift now connects with IAM Identity Center. In fact, multiple Redshift clusters or workgroups can be connected, so that your Redshift data warehouses have a single and consistent view of users and their attributes.

AWS analytics services, including Amazon Redshift, QuickSight, Amazon EMR, Lake Formation, and Amazon Simple Storage Service (Amazon S3) using S3 Access Grants, now use trusted identity propagation with IAM Identity Center to manage and audit access to data and resources based on user identity. Identity information can be passed between connected business intelligence and data analytics applications. For example, a QuickSight user can create an Amazon Redshift data source without an admin needing to define complex data permissions. Instead, the identity propagation means that the relevant user information is made available to Amazon Redshift and the respective role can be automatically assumed. This makes identity and access management between Amazon Redshift and complimentary analytics services more straightforward.

This is made possible because connecting to IAM Identity Center allows mapping users to roles regardless of the IdP federation used. This gives a standardized approach that combines the benefits of authentication through your IdP with the capabilities available with the RBAC features of Amazon Redshift.

For more information about Amazon Redshift integration with IAM Identity Center, see Integrate Identity Provider (IdP) with Amazon Redshift Query Editor v2 using AWS IAM Identity Center for seamless Single Sign-On and Simplify access management with Amazon Redshift and AWS Lake Formation for users in an External Identity Provider.

Amazon Redshift also supports direct IdP integration with SAML 2.0-compliant IdPs, including Microsoft Entra ID, Active Directory Federation Services, Okta, Ping Federate, and others.

Native federation

Native federation offers similar security-related benefits, but is more restricted in compatibility and cannot take advantage of identity propagation. Currently, only Microsoft Azure Entra ID is supported by native federation. The external IdP can be registered with Amazon Redshift using

a SQL command. Further, when users connect to Amazon Redshift, their external IdP group memberships are mapped to Redshift roles. Mapping to roles brings the various security related benefits of the RBAC features of Amazon Redshift. Users and roles can be created automatically or pre-created to facilitate granting permissions in advance.
See Native identity provider (IdP) federation for Amazon Redshift and Integrate Amazon Redshift native IdP federation with Microsoft Azure AD using a SQL client for more information.

IAM authentication

Rather than relying on static, long-lived credentials in the data warehouse, the system can be configured to allow users to generate sign-in credentials based on their IAM credentials. Amazon Redshift provisioned clusters provide the GetClusterCredentials API operation that generates temporary database user credentials. So, a user or role with the associated IAM permissions can call this API to dynamically create sign-in credentials. With this method, a username can be provided and a temporary password is generated to connect to the database. This can be used to sign in as an existing database user or an auto-create option can be specified to automatically create database users when needed. In accordance with the principle of least privilege access, limit this permission to specific IAM roles or users according to their requirements. Alongside the database user, you can also specify database group names that the user will join when signed in; these memberships are only valid for the session, they are not persisted.
In addition to this, temporary credentials can be generated from the user's IAM identity with the GetClusterCredentialsWithIAM API provided by Amazon Redshift provisioned clusters. With this method, a user is mapped one-to-one with an IAM identity and a temporary password is generated to connect to the database. According to the principle of least privilege access, limit this permission to specific IAM roles or users based on their requirements.
With Amazon Redshift serverless, you can follow a similar pattern using the GetCredentials API to retrieve temporary sign-in credentials.
Further, SQL clients can be configured with Amazon Redshift JDBC or Open Database Connectivity (ODBC) drivers to automatically manage the process of retrieving temporary credentials and connecting to Amazon Redshift. This requires providing IAM credentials that can be used for calling the `GetClusterCredentials` operation. You can learn more about configuring a JDBC or ODBC connection to use IAM credentials in Options for providing IAM credentials.

Local users

While this method is straightforward to set up, it is not generally recommended for production workloads because it leads to identity information being created directly in the data warehouse. Relevant permissions must then be manually maintained and using this method can lead to users and permissions becoming outdated. This approach involves an admin or superuser with relevant permissions signing in to the data warehouse and running Redshift SQL commands to create or update users. This can add overhead because it means user identities

and associated permissions have to be maintained in the data warehouse. It can also pose security concerns because users and permissions can quickly become out of date.
The admin also runs commands to associate the user with applicable database groups or roles. This again adds manual overhead to manage and maintain associations. See Users for more information.

Secrets Manager integration

There is also the option to take advantage of integration with AWS Secrets Manager. With this method, instead of specifying credentials directly, a reference to a secret stored in Secrets Manager that contains database and sign-in credentials is provided. This is beneficial because it means that hard coded or plain text secrets don't need to be used, which reduces the risk of unintended access. This can be especially useful for managing admin user credentials.
When creating your Amazon Redshift data warehouse, it is recommended to use this option to manage admin credentials. By default, Amazon Redshift automatically rotates the secret every 30 days and updates the admin user password in the data warehouse. This makes sure that the same credentials are not used for prolonged periods. Secret rotation can be set to the required frequency or triggered manually. For information about creating a secret, see Creating a secret for database connection credentials.

Recommendations

In general, using an IdP for federated sign in and connecting Amazon Redshift to IAM Identity Center is recommended. You can use IAM Identity Center to manage identities, or use syncing to use your existing IdP for user identity management. This brings security advantages because it allows a single source of truth for identity management, with permissions and associated roles managed by the IdP rather than having to be additionally maintained in Amazon Redshift. It makes data warehouse users' sign-in experience simpler because they can use an SSO approach rather than having to manage additional credentials.
Whichever federation provider is used, connecting Amazon Redshift with IAM Identity Center means that users in the directory are mapped to Redshift local database users and user security groups in the external directory are mapped to Redshift roles. This allows RBAC features to control user access to data and further allows identity propagation so that user information can be passed between supporting services. This gives a simplified, consistent approach alongside better auditability.

Best practice recommendations

- For managing Amazon Redshift resources, use IAM roles scoped to specific personas and job functions. This makes sure that you adhere to the principle of least access. It is recommended to use IAM Identity Center to provide an SSO experience for authenticating to AWS in order to assume IAM roles and perform related functions.

- If you have a user directory managed through an IdP but are still using local data warehouse users within Amazon Redshift for authentication, then it is recommended to consider using IdP federation instead of local, data warehouse-based sign-ins. Consider using AWS IAM Identity Center as a single source of truth for your identities in AWS. IAM Identity Center can be used as an IdP or synced with your existing IdP. This allows you to create an SSO experience across analytics services including Amazon Redshift Query Editor v2, QuickSight, and Lake Formation.
- If you are using IAM Identity Center, consider connecting it with Amazon Redshift. A single IAM Identity Center can be connected to multiple Redshift clusters or workgroups. After being connected, IAM Identity Center helps ensure that your Redshift data warehouses have consistent user information including user attribute information. Further, this connection maps users to Amazon Redshift roles when they sign in to the data warehouse, meaning that you can use the RBAC capabilities within Amazon Redshift. This method also enables identity propagation so that supporting services can use role information to provide data access control.
- If you have previously created Redshift groups to control the permissions granted to users, consider mapping these to Redshift roles so that the SSO sign-in process can assume existing roles with the relevant permissions assigned to them. There are example scripts on GitHub that might assist in automating this process.
- Use AWS Secrets Manager integration to manage admin credentials.

Data access control

This section will explore how you can manage and control the data within your Amazon Redshift data warehouse.

While securing access to your Redshift resources is crucial, it's equally important to make sure that the right personas across your business have the appropriate level of access to the data that they need to perform their roles effectively. In this part, we'll dive into strategies and techniques for implementing fine-grained access control to your Redshift data, enabling you to balance data security and data usability.

Database security

Database security refers to authorization controls. These controls can be put in place to make sure that data warehouse users can only access relevant data and perform permitted tasks. Traditionally, this might have involved a large number of coarse-grained policies designed for one-to-one mapping to users or services. This can lead to an excessive number of policies that become hard to manage. For data restriction requirements, traditional methods can result in additional data sets being created to support specific views of information. Again, this adds complexity and increases the chance of mismanagement over the long term. Amazon Redshift has capabilities that can help avoid complex policies and architecture, while maintaining that access and permissions are controlled.

Role based access control

It is a recommended best practice to use Redshift roles to manage permissions rather than granting permissions directly to users or using database groups. Redshift roles permit a scalable approach to managing permissions in addition to enabling capabilities such as fine-grained access control.

RBAC is a powerful approach that can help simplify database permissions and make sure that users can only perform authorized tasks. By defining roles and associating them with a set of permissions, RBAC allows admins to grant these roles to database users rather than assigning permissions directly. This provides a scalable and manageable way to control access, because users with an assigned role can only perform the tasks that the role has authorization for.

This is similar to using database groups to define permissions, however one of the advantages of using RBAC is that roles can be assigned to other roles. This means that you can build a hierarchy leading to fewer grants. This is beneficial because it should lead to more accurate permissions and less time managing access.

With RBAC, you can use coarse-grained security to grant access to certain schemas or tables within your Amazon Redshift data warehouse. Additionally, RBAC supports the application of row- and column-level policies to roles, enabling fine-grained control over data access. The RBAC pattern with fine-grained controls provides a scalable approach for accessing data, because it alleviates the need for different database tables (or similar) to provide different views of data according to allowed access. For more information, see Roles.

Fine-grained access control: Row and column level access policies

Row and column level security enables granular access control over data. Unlike coarse-grained controls that grant access to entire databases, schemas, or tables, these fine-grained controls can be customized to specific users or roles, allowing access to match particular job functions or workload requirements.

With column level security, you can restrict users to view and query only a subset of table fields. For example, you cannot allow SELECT on sensitive data fields, such as address, for all users. Similarly, row level security creates policies that restrict access to specific rows based on a predicate. For example, you can restrict access to specific categories within a field. You can combine row and column constraints as required.

These controls implement the principle of least privilege, surfacing only the required information. Use them where applicable to control access by different job roles and personas. It is generally recommended to use these policies in tandem with Redshift roles to apply scalable data access controls.

For examples of restricting column access, see GRANT examples. For more information about row level security policies, see row level security.

Metadata security

Redshift data warehouses provide a range of metadata that can be queried through system tables and views. This metadata can be useful information, but if your warehouse spans different teams or domains, you should control the information that different users, groups, or roles can see. Otherwise, metadata about non-authorized objects might be visible. This is particularly relevant if you are using a multi-tenant approach or have users from across different teams or domains. Without metadata security, users might be able to see information about data warehouse objects that they do not have access to.
To help ensure that users can only view metadata for the objects that they have access to, implement metadata security. This is a system level setting that can be turned on or off. Turning it on will help prevent users from accessing metadata for objects that they are not authorized to view.
For information about setting this configuration, see Metadata Security.

Data encryption

Data encryption is an essential security measure to protect data throughout its lifecycle. AWS recommends encryption as an additional access control to complement identity, resource, and network-oriented options.

When implementing encryption, you can choose to do so either on the client side or server side:

- **Client-side encryption** – You manage the encryption/decryption process, key management, and related tooling.
- **Server-side encryption** – Amazon Redshift automatically encrypts data as it is written and decrypts it as it is retrieved.

Encryption at rest

To help ensure that data at rest is secure, encrypt your Amazon Redshift data warehouse using server-side encryption. This can be applied during cluster creation or for encrypting a currently unencrypted cluster. The AWS Management Console view for applying encryption during a provisioned cluster creation is shown in the following figure.

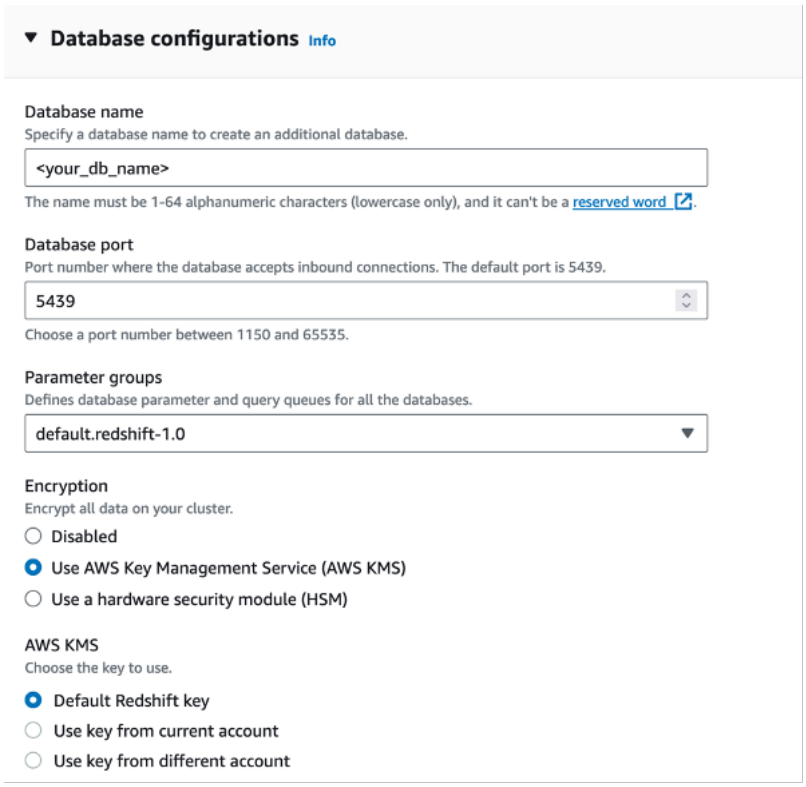

Figure 3 – Enable encryption during data warehouse creation or for an existing warehouse using the default AWS managed key or a customer managed key

When creating an Amazon Redshift Serverless namespace, it is encrypted by default. You can use AWS Key Management Service (AWS KMS) to encrypt the database with either the default AWS-managed key or a customer-managed key. Using a customer-managed key allows you to rotate, disable, and audit the key according to your specific requirements, which might be required for certain compliance standards. You can read more at Encryption at rest.

During data warehouse creation, it is recommended to change the default database name to a value specific to your needs rather than using default values. This can help reduce the likelihood of inadvertent or malicious access.

Encryption in transit

When connecting to Amazon Redshift from clients or other services, it is important to use Transport Layer Security (TLS) or Secure Sockets Layer (SSL) encrypted connections to protect data as it is communicated over networks. This makes sure that data moving into or being retrieved from the database is encrypted in transit. To support this, AWS Certificate Manager automatically installs certificates on each data warehouse so that SSL/TLS can be used. Whenever possible, configure Amazon Redshift to only allow SSL/TLS connections.

When connecting from clients, you might also want to require additional checks, such as verifying the identity of the server, commonly through passing the parameters verify-ca or verify-full in your JDBC or ODBC connection string. In this case, you can install root certificates on the client, which allow these checks. For more information about this process, see Configuring security options for connections.

To protect requests to the Amazon Redshift API, calls must be conducted as part of the AWS Signature Version 4 signing process. This can be achieved using the AWS Command Line Interface (AWS CLI), AWS SDK or other toolkits. If you use the Amazon Redshift Query Editor v2, data is transmitted between the query editor and the Amazon Redshift cluster over a TLS-encrypted channel.

Data sharing

A key component of your data warehouse is making the right data available to the teams who need it to drive and optimize business decisions. As you collect more data and more personas and teams across your organization look to consume analytical data, it is critical to provide this information in a secure and reliable manner.

Traditionally, this might have involved onboarding additional users into a monolithic data warehouse. However, this can cause security concerns over managing permissions, generate resource contention, and make the data warehouse team a bottleneck, slowing down access to insights.

An alternative approach is to export data relevant to different consuming teams, who can then load and query the data in their own environments. However, this again causes security concerns because control of the data is lost; other environments might not implement proper controls, there might be redundant data proliferation, and subsequent processing by different teams can lead to different answers to the same question, ultimately leading to a decline in data trust.

Amazon Redshift data sharing can address many of these concerns. With an Amazon Redshift data share, data duplication is minimized because the underlying data asset is made available to consuming data warehouses instead of being copied. This removes the need for export extract, transform, and load (ETL) pipelines and minimizes redundant data. It also allows for straightforward ongoing management of shared data, with the ability to manage and remove access when needed.

Consuming teams can bring their own Amazon Redshift data warehouse and access relevant data that has been shared with them, without causing contention on a centralized, shared resource. Because consuming teams manage their own warehouse, bottlenecks from using a single team to manage resources are minimized, and control is kept close to the owning team, meaning security and access is more likely to be kept up to date.

The Amazon Redshift data sharing pattern allows control to be retained by the owning team because shared data can be revoked at any time by the producer, meaning that the consuming team loses access to this information. It is also possible to apply fine-grained access control

over this shared data using Lake Formation on Amazon Redshift data shares. For more information, see Managing permissions for data in an Amazon Redshift datashare.

Data tokenization and masking

Data tokenization can help protect sensitive data by replacing or masking specific fields that might contain sensitive information, such as PII. Detokenization can later retrieve the original values, but its use can be controlled to only permit access by authorized roles.
This process can be implemented in Amazon Redshift using user-defined functions. It should be used to protect sensitive fields that need to be stored in the warehouse yet remain retrievable by certain roles.
Alternatively, you can conduct a one-way hash or tokenization process to protect data. This gives a consistent result for the same value, allowing future records to be matched, but does not allow subsequent retrieval of the original data. This approach should be favored if the original value is not required in the warehouse.
For more information, see Data tokenization.

Dynamic data masking

Dynamic data masking in Amazon Redshift allows data obfuscation at query time, rather than over the underlying data stored in the warehouse. This allows sensitive fields to be fully or partially masked to protect sensitive data, with the masking applied based on database user permissions.
These controls can be applied at the column level or even the cell level, through the combination of column-level rules and a conditional column.
For more information, see Dynamic data masking.
Use a tokenization or data masking strategy when you need to have sensitive fields in the warehouse. If you only need to be able to make joins or similarly demarcate certain values, consider creating synthetic keys or IDs that allow joining and querying without storing unnecessary sensitive data in the warehouse.

Auditing

Amazon Redshift logs information in the following log files:

- **Connection log** – logs authentication attempts, connections, and disconnections
- **User log** – logs information about changes to database user definitions
- **User activity log** – logs each query before it's run on the database

This information is available through system tables and views, but it's recommended to output the logs to Amazon S3 or Amazon CloudWatch. This will persist the information and enable further analytics. For Amazon Redshift Serverless, audit logging data can only be sent to CloudWatch.
When exporting the data to Amazon S3 or CloudWatch, access must be controlled using the relevant IAM permissions for the S3 buckets or CloudWatch log groups, because the data is now outside the database.

The connection and user logs are primarily useful for security purposes, such as monitoring connections, attempted connections, and user updates. For more information, see Database audit logging.
In addition to database logging, Amazon Redshift, Amazon Redshift Serverless, data API, data sharing, and Query Editor v2 actions can be logged by AWS CloudTrail. Use CloudTrail to monitor and analyze activity related to your Amazon Redshift resources. For more information, see Logging with CloudTrail.

Data lifecycle management

Amazon Redshift provides automated snapshots of your database by default. These snapshots are retained for 1 day, but this duration is configurable up to 35 days. This helps manage the lifecycle of your snapshots.
You can also take manual snapshots, which are stored indefinitely by default. To control the proliferation of these manual snapshots, you should:

- Use a retention period on manual snapshots to make sure that they are deleted after a certain time.
- Implement a strategy such as using the AWS CLI command batch-delete-cluster-snapshots to delete snapshots in bulk.

By following these practices, you can effectively manage the lifecycle of your Amazon Redshift snapshots and help prevent the accumulation of unnecessary data.

Recommendations

We have discussed various practices to help secure your data within Amazon Redshift data warehouses.

- **Use Redshift RBAC to manage data warehouse permissions** – It is necessary to implement controls over your data to make sure that it can be safely made available for analytics. This requires the ability to use consistent, scalable controls to help ensure that users can only access the data they require. In Amazon Redshift, this can be achieved using a combination of RBAC and fine-grained access policies.
- **Encrypt data at rest and make sure that connecting clients encrypt data in transit** – Data encryption is a key strategy to protect data throughout its lifecycle. This is critical in the data warehouse, which consumes and stores large amounts of data that is then queried by consumers. Amazon Redshift supports encryption strategies by supporting encryption in transit through common protocols such as TLS. Encryption at rest of the data warehouse is also supported.
- **Use Amazon Redshift data sharing to allow access to data from consuming Redshift data warehouses** –Making data available to other teams or business areas is a key requirement of a data environment, which often leads to security concerns and scaling issues. Amazon Redshift data sharing offers a straightforward and secure way of sharing data while allowing producers to retain control.

- **Use tokenizing or masking strategies where sensitive data is needed** – If sensitive data is not required in the data warehouse, it is generally recommended to drop or obfuscate this before ingestion into the warehouse. If, however, it is required for a business purpose, data tokenization or obfuscation strategies can be used to make sure that only allowed personas can access the sensitive data.
- **Collect and monitor auditing information** –Throughout the operation of your data warehouse, it is important to maintain visibility of activity for monitoring and performance reasons, in addition to security. Information made available by Amazon Redshift or through Redshift API calls can be persisted to allow ongoing monitoring and storage of historic activity.

Networking

In this section, we will explore the networking and design considerations when setting up your data warehouse. Network design can improve the security posture of your data warehouse, but it can also introduce complexities in allowing users or consuming services access to the needed data. This section will discuss how you can build a secure environment and achieve the required integrations and access to allow data to be used across business intelligence and other applications.

When you launch an Amazon Redshift data warehouse, the underlying infrastructure is created and managed within a private, closed-off, AWS-managed account. The warehouse is then made available within your own Amazon Virtual Private Cloud (Amazon VPC) through network interfaces.

You can choose to have the data warehouse available in public or private subnets within your VPC. Unless you have a strong use case not to, choose a private subnet to help isolate your warehouse from the public internet. By default, Amazon Redshift has a publicly accessible setting that is turned off (shown in the following figure), so instances outside your VPC cannot connect to the database.

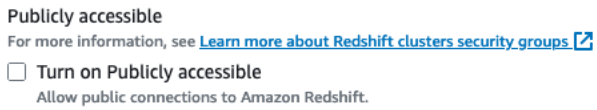

Figure 4 – Unless you have a specific use case not to, keep the default setting of Publicly accessible off, this helps to prevent unintended accessibility to your data warehouse

Isolating your Amazon Redshift warehouse can bring challenges in allowing your analysts and consuming applications to access it. There are several approaches to provide access:

- Amazon Redshift Query Editor v2 – A browser-based query editor
- **Network load balancer** – to accept connections and forward to the Amazon Redshift data warehouse
- **Bastion server** with port forwarding

- Amazon Redshift VPC endpoints – For communication with the Amazon Redshift API service
- **Application specific solutions** – For example, see Integrate Power BI with Amazon Redshift for insights and analytics

If you need to connect from a corporate network, consider setting up an AWS Site-to-Site VPN or AWS Direct Connect. These can offer private connectivity between your network and your Amazon Redshift VPC.

With your resources available in specified subnets, you can use network access control lists (NACLs) to apply stateless rules over traffic at the subnet level. The default NACL allows all inbound and outbound traffic when associated with a subnet, as shown in the following figure. You can add allow or deny rules as needed or define and apply your own NACL, depending on your specific network security requirements. You can read more about network access control lists in Control subnet traffic with network access control lists.

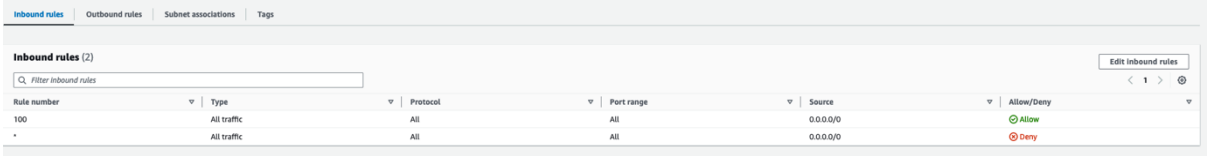

Figure 5 – The default network access control list associated with your VPC subnet will allow all traffic. You can add allow and deny rules as required

While you might or might not require additional network constraints such as network access control lists, we recommend you attach VPC security groups to your Redshift warehouse to apply stateful rules on ingress and egress traffic (as shown in the following figure). A security group controls the traffic that is allowed in and out of the resources that it is associated with. For example, after you associate a security group with an Amazon Elastic Compute Cloud (Amazon EC2) instance, it controls the inbound and outbound traffic for the instance.

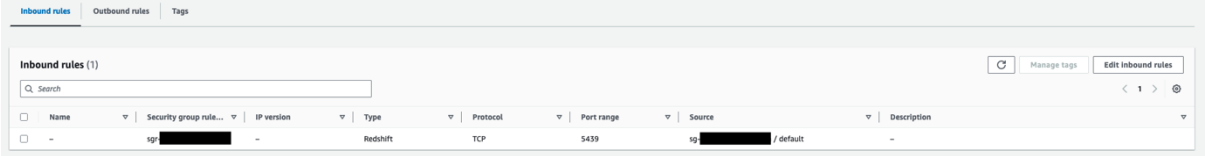

Figure 6 – You can associate security groups with your data warehouse that allow only the relevant port access from allowed sources. You can similarly set outbound rules.

The associated security groups should also follow the principle of least privilege access, allowing only the required port access. For example, an inbound rule allowing access to the default Amazon Redshift port 5439 from only the required sources such as your corporate domain or VPC Classless Inter-Domain Routing (CIDR) range. For more information, see Managing VPC security groups.

Amazon Redshift offers a comprehensive set of capabilities through integration with other AWS services. For example, during data load or unload operations, Amazon Redshift can efficiently

read from and write to Amazon S3. If your Amazon Redshift warehouse has a route to the internet, it will use public S3 endpoints to allow access to your data and enable you to seamlessly upload and unload data to S3.
However, if your Amazon Redshift warehouse is in an isolated subnet, with no route to the internet, you can use Amazon S3 gateway endpoints to interact with Amazon S3 from your Amazon Redshift VPC. This enables a route through the internal AWS network, allowing you to access your data without a direct internet connection.
Similarly, other related AWS services can be consumed using the appropriate VPC endpoint for that service, as shown on the following figure. In particular, you might want to add an AWS Secrets Manager endpoint so that Amazon Redshift can use this network path when retrieving credentials.

Figure 7 – Add endpoints for various AWS services to your VPC to enable internal network routes

When you set up alternative network paths, such as with VPC endpoints, Amazon Redshift has an *Enhanced VPC Routing* setting that should be turned on, as shown in the following figure. This tells Amazon Redshift to use the strictest network path available. For example, if you are interacting with Amazon S3 and you have an S3 endpoint in your VPC in addition to a route to the internet, Amazon Redshift will use the endpoint path rather than the public endpoint. For more information, see Enhanced VPC routing in Amazon Redshift.

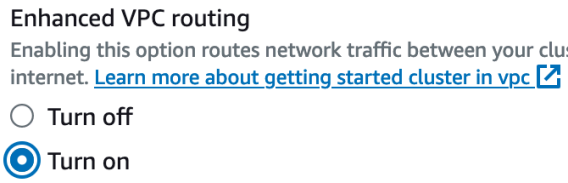

Figure 8 – Turn on enhanced VPC routing to so that Amazon Redshift uses the most specific network path available for related services.

When connecting to your Amazon Redshift data warehouse, your clients will use an endpoint, such as a JDBC endpoint. To help avoid exposing your Amazon Redshift endpoint directly, you can use a custom domain name (also known as a canonical name or CNAME) that routes clients to either your Amazon Redshift cluster or Amazon Redshift Serverless workgroup. This custom domain name provides a security enhancement by protecting the endpoint and can make the endpoint name easier to remember.
To set up this custom domain name, you will need to register a domain name with Amazon Route 53. The process is outlined in more detail in Custom domain names for client connections.

Monitoring best practice adherence

This whitepaper concludes by demonstrating how to automatically measure your Amazon Redshift provisioned clusters against suggested best practices. This will provide you with the information needed to monitor your deployments, assess overall adherence to best practices, and maintain visibility as clusters are deployed or evolve over time.

You can monitor your Redshift provisioned clusters for certain settings using AWS Config and related rules. These rules allow you to monitor your deployed settings and check for adherence to your defined practices.

There is a pre-built set of rules available as part of an AWS Config conformance pack that checks for many of the best practices discussed here, but you can also define and use your own custom rules. You can see details of the pre-built conformance pack on GitHub. You can use this as a conformance pack template or customize it to create your own template.

Deploy the conformance pack using AWS Config, which can then monitor your resources and detect settings on an ongoing basis (shown in the following figure).

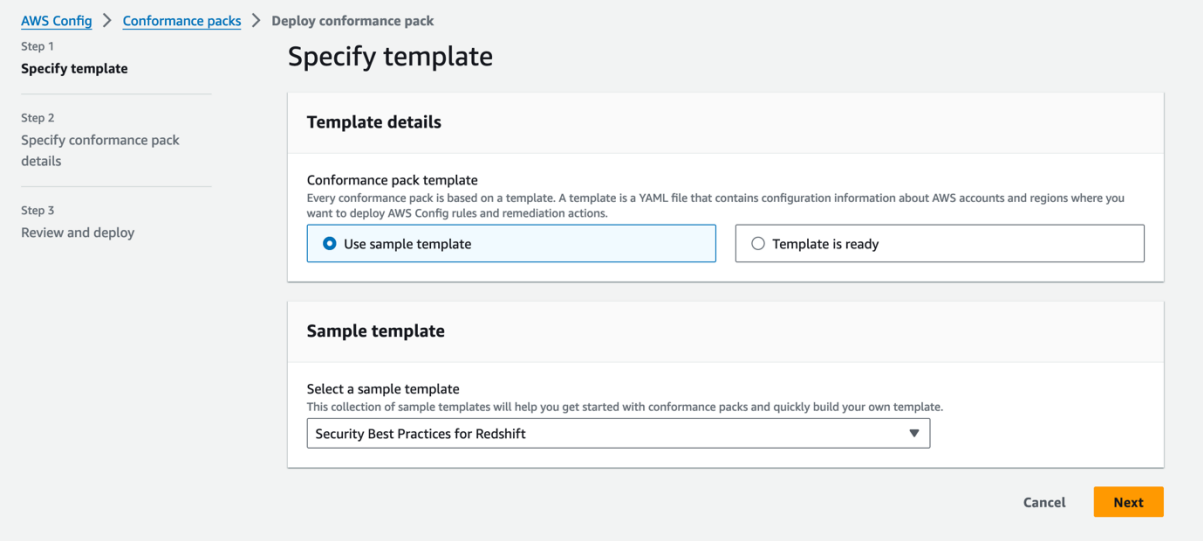

Figure 9 – Deploying the Security Best Practices for Redshift conformance pack using AWS Config.

After being deployed, the rules will be evaluated, and you can view dashboards and metrics on conformance status. For example, if a resource doesn't meet a required condition, it will be highlighted, as shown in the following figure.

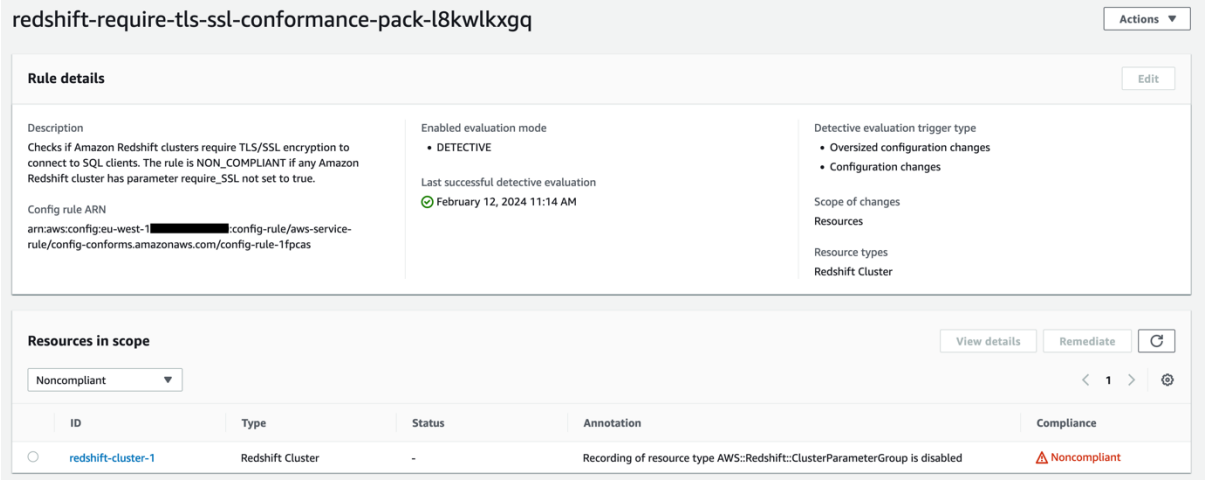

Figure 10 – A rule violation is detected because the required SSL parameter is not set to true on this cluster.

Further, you can view conformance across one or more clusters at an aggregate level and monitor how these change over time through conformance pack dashboards, as shown on the following figure.

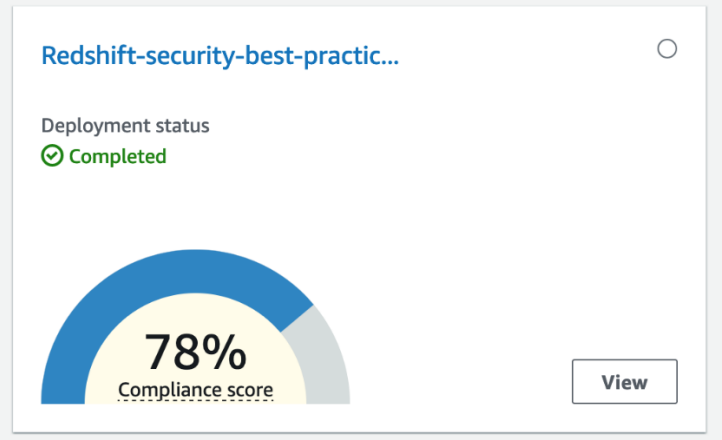

Figure 11 – The percentage of passing rule-resource combinations can be tracked to give an indication to overall conformance across your clusters.

While AWS Config and the pre-built set of security best practice rules can help you quickly monitor your resources for general best practices, there is no one-size-fits-all approach for all scenarios. You can use the recommended rules or use them as a starting point to create your own ruleset.

Conclusion

In this whitepaper, we have covered best practices to secure your Amazon Redshift data warehouses and help ensure that you can scale access to data to drive business value.

In Authentication and authorization, we discussed strategies for providing a single sign-on (SSO) experience to Amazon Redshift and assuming an identity with specific authorizations. A key recommendation was to use AWS Identity and Access Management (IAM) Identity Center to allow SSO and the assumption of roles to control access within the data warehouse.

In Data access control, we extended the exploration of RBAC to show how using roles to manage permissions, in combination with row and column level access policies, provides a scalable, fine-grained access control methodology within Amazon Redshift. We also discussed topics including encryption at rest and in transit to help protect data, and options for tokenizing or masking sensitive data.

In Networking and Monitoring best practice adherence, we showed methods to help isolate resources and monitor your deployments for specific practices that you can define.

The guidance covered in this whitepaper must be taken in the unique context of your business and domain and implemented accordingly. It is also not static. New features and capabilities will be introduced to assist in security challenges, and at the same time, new threats and requirements will emerge. Therefore, it is critical to remain up-to-date with the latest features and guidance and reflect these in your data warehouse architecture, practices, and configuration.

Contributors

Document author:

- Gregory Knowles, Senior Data and AI Specialist Solutions Architect, AWS

Contributions from:

- Yanzhu Ji, Senior Product Manager – Technical, AWS

Reviewed by:

- Alejandra Catalina Abrusci, Senior Solutions Architect, AWS
- Sushil Shah, Senior Big Data Solutions Architect, AWS
- Bukky Gibson, Partner Solution Architect, AWS

Document revisions

Date	Description
January, 2025	First publication