Overview of AWS Security - Database Services

June 2016

(Please consult **http://aws.amazon.com/security/** for the latest version of this paper)

Notices

Database Services

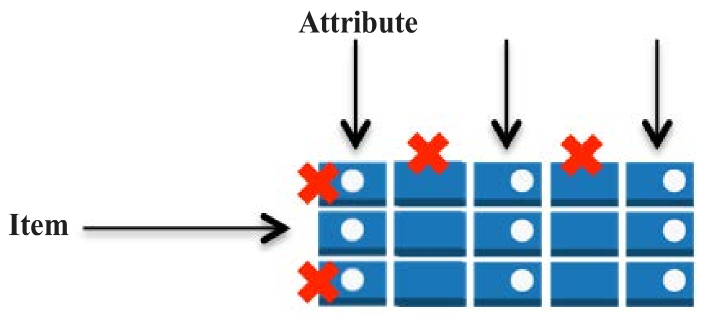
Amazon Web Services provides a number of database solutions for developers and businesses—from managed relational and NoSQL database services, to in-memory caching as a service and petabyte-scale data-warehouse service.

Amazon DynamoDB Security

Amazon DynamoDB is a managed NoSQL database service that provides fast and predictable performance with seamless scalability. Amazon DynamoDB enables you to offload the administrative burdens of operating and scaling distributed databases to AWS, so you don't have to worry about hardware provisioning, setup and configuration, replication, software patching, or cluster scaling.

You can create a database table that can store and retrieve any amount of data, and serve any level of request traffic. DynamoDB automatically spreads the data and traffic for the table over a sufficient number of servers to handle the request capacity you specified and the amount of data stored, while maintaining consistent, fast performance. All data items are stored on Solid State Drives (SSDs) and are automatically replicated across multiple availability zones in a region to provide built-in high availability and data durability.
You can set up automatic backups using a special template in AWS Data Pipeline that was created just for copying DynamoDB tables. You can choose full or incremental backups to a table in the same region or a different region. You can use the copy for disaster recovery (DR) in the event that an error in your code damages the original table, or to federate DynamoDB data across regions to support a multi-region application.

To control who can use the DynamoDB resources and API, you set up permissions in AWS IAM. In addition to controlling access at the resource-level with IAM, you can also control access at the database level—you can create database-level permissions that allow or deny access to items (rows) and attributes (columns) based on the needs of your application. These database-level permissions are called fine-grained access controls, and you create them using an IAM policy that specifies under what circumstances a user or application can access a DynamoDB table. The IAM policy can restrict access to individual items in a table, access to the attributes in those items, or both at the same time.



You can optionally use web identity federation to control access by application users who are authenticated by Login with Amazon, Facebook, or Google. Web identity federation removes the need for creating individual IAM users; instead, users can sign in to an identity provider and then obtain temporary security credentials from AWS Security Token Service (AWS STS). AWS STS returns temporary AWS credentials to the application and allows it to access the specific DynamoDB table.

In addition to requiring database and user permissions, each request to the DynamoDB service must contain a valid HMAC-SHA256 signature, or the request is rejected. The AWS SDKs automatically sign your requests; however, if you want to write your own HTTP POST requests, you must provide the signature in the header of your request to Amazon DynamoDB. To calculate the signature, you must request temporary security credentials from the AWS Security Token Service. Use the temporary security credentials to sign your requests to Amazon DynamoDB.

Amazon DynamoDB is accessible via SSL-encrypted endpoints. The encrypted endpoints are accessible from both the Internet and from within Amazon EC2.

Amazon Relational Database Service (Amazon RDS) Security

Amazon RDS allows you to quickly create a relational database (DB) instance and flexibly scale the associated compute resources and storage capacity to meet application demand. Amazon RDS manages the database instance on your behalf by performing backups, handling failover, and maintaining the database software. Currently, Amazon RDS is available for Amazon Aurora, MySQL, PostgreSQL, Oracle, Microsoft SQL Server, and MariaDB database engines.

Amazon RDS has multiple features that enhance reliability for critical production databases, including DB security groups, permissions, SSL connections, automated backups, DB snapshots, and multi-AZ deployments. DB instances can also be deployed in an Amazon VPC for additional network isolation.

Access Control

When you first create a DB Instance within Amazon RDS, you will create a master user account, which is used only within the context of Amazon RDS to control access to your DB Instance(s). The master user account is a native database user account that allows you to log on to your DB Instance with all database privileges. You can specify the master user name and password you want associated with each DB Instance when you create the DB Instance. Once you have created your DB Instance, you can connect to the database using the master user credentials. Subsequently, you can create additional user accounts so that you can restrict who can access your DB Instance.

Using AWS IAM, you can further control access to your RDS DB instances. AWS IAM enables you to control what RDS operations each individual AWS IAM user has permission to call.

Network Isolation

For additional network access control, you can run your DB Instances in an Amazon VPC. Amazon VPC enables you to isolate your DB Instances by specifying the IP range you wish to use, and connect to your existing IT infrastructure through industry-standard encrypted IPsec VPN. Running Amazon RDS in a VPC enables you to have a DB instance within a private subnet. You can also set up a virtual private gateway that extends your corporate network into your VPC, and allows access to the RDS DB instance in that VPC. Refer to the Amazon VPC User Guide for more details.

DB Instances deployed within an Amazon VPC can be accessed from the Internet or from Amazon EC2 Instances outside the VPC via VPN or bastion hosts that you can launch in your public subnet. To use a bastion host, you will need to set up a public subnet with an EC2 instance that acts as a SSH Bastion. This public subnet must have an Internet gateway and routing rules that allow traffic to be directed via the SSH host, which must then forward requests to the private IP address of your Amazon RDS DB instance.

DB Security Groups can be used to help secure DB Instances within an Amazon VPC. In addition, network traffic entering and exiting each subnet can be allowed or denied via network ACLs. All network traffic entering or exiting your Amazon VPC via your IPsec VPN connection can be inspected by your on-premises security infrastructure, including network firewalls and intrusion detection systems.

Encryption

You can encrypt connections between your application and your DB Instance using SSL. For MySQL and SQL Server, RDS creates an SSL certificate and installs the certificate on the DB instance when the instance is provisioned. For MySQL, you launch the mysql client using the --ssl_ca parameter to reference the public key in order to encrypt connections. For SQL Server, download the public key and import the certificate into your Windows operating system. Oracle RDS uses Oracle native network encryption with a DB instance. You simply add the native network encryption option to an option group and associate that option group with the DB instance. Once an encrypted connection is established, data transferred between the DB Instance and your application will be encrypted during transfer. You can also require your DB instance to only accept encrypted connections.

Amazon RDS supports Transparent Data Encryption (TDE) for SQL Server (SQL Server Enterprise Edition) and Oracle (part of the Oracle Advanced Security option available in Oracle Enterprise Edition). The TDE feature automatically encrypts data before it is written to storage and automatically decrypts data when it is read from storage. If you require your MySQL data to be encrypted while “at rest” in the database, your application must manage the encryption and decryption of data.

Note that SSL support within Amazon RDS is for encrypting the connection between your application and your DB Instance; it should not be relied on for authenticating the DB Instance itself.

While SSL offers security benefits, be aware that SSL encryption is a compute intensive operation and will increase the latency of your database connection. To learn more about how SSL works with MySQL, you can refer directly to the MySQL documentation found here. To learn how SSL works with SQL Server, you can read more in the RDS User Guide.

Automated Backups and DB Snapshots

Amazon RDS provides two different methods for backing up and restoring your DB Instance(s): automated backups and database snapshots (DB Snapshots).

Turned on by default, the automated backup feature of Amazon RDS enables point-in-time recovery for your DB Instance. Amazon RDS will back up your database and transaction logs and store both for a user-specified retention period. This allows you to restore your DB Instance to any second during your retention period, up to the last 5 minutes. Your automatic backup retention period can be configured to up to 35 days.

During the backup window, storage I/O may be suspended while your data is being backed up. This I/O suspension typically lasts a few minutes. This I/O suspension is avoided with Multi-AZ DB deployments, since the backup is taken from the standby.

DB Snapshots are user-initiated backups of your DB Instance. These full database backups are stored by Amazon RDS until you explicitly delete them. You can copy DB snapshots of any size and move them between any of AWS' public regions, or copy the same snapshot to multiple regions simultaneously. You can then create a new DB Instance from a DB Snapshot whenever you desire.

DB Instance Replication

Amazon cloud computing resources are housed in highly available data center facilities in different regions of the world, and each region contains multiple distinct locations called Availability Zones. Each Availability Zone is engineered to be isolated from failures in other Availability Zones, and to provide inexpensive, low-latency network connectivity to other Availability Zones in the same region.

Amazon RDS provides high availability and failover support for DB instances using Multi-AZ deployments. Multi-AZ deployments for Oracle, PostgreSQL, MySQL, and MariaDB DB instances use Amazon technology, while SQL Server DB instances use SQL Server Mirroring. Note that Amazon Aurora stores copies of the data in a DB cluster across multiple Availability Zones in a single region, regardless of whether the instances in the DB cluster span multiple Availability Zones. In a Multi-AZ deployment, Amazon RDS automatically provisions and maintains a synchronous standby replica in a different Availability Zone. The primary DB instance is synchronously replicated across Availability Zones to a standby replica to provide data redundancy, eliminate I/O freezes, and minimize latency spikes during system backups. In the event of DB instance or Availability Zone failure, Amazon RDS will automatically failover to the standby so that database operations can resume quickly without administrative intervention. Running a DB instance with high availability can enhance availability during planned system maintenance, and help protect your databases against DB instance failure and Availability Zone disruption.

Amazon RDS also uses the PostgreSQL, MySQL, and MariaDB DB engines' built-in replication functionality to create a special type of DB instance called a Read Replica from a source DB instance. Updates made to the source DB instance are asynchronously copied to the Read Replica. You can reduce the load on your source DB instance by routing read queries from your applications to the Read Replica. Read Replicas allow you to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads."

Automatic Software Patching

Amazon RDS will make sure that the relational database software powering your deployment stays up-to-date with the latest patches. When necessary, patches are applied during a maintenance window that you can control. You can think of the Amazon RDS maintenance window as an opportunity to control when DB Instance modifications (such as scaling DB Instance class) and software patching occur, in the event either are requested or required. If a “maintenance” event is scheduled for a given week, it will be initiated and completed at some point during the 30-minute maintenance window you identify.

The only maintenance events that require Amazon RDS to take your DB Instance offline are scale compute operations (which generally take only a few minutes from start-to-finish) or required software patching. Required patching is automatically scheduled only for patches that are security and durability related. Such patching occurs infrequently (typically once every few months) and should seldom require more than a fraction of your maintenance window. If you do not specify a preferred weekly maintenance window when creating your DB Instance, a 30-minute default value is assigned. If you wish to modify when maintenance is performed on your behalf, you can do so by modifying your DB Instance in the AWS Management Console or by using the ModifyDBInstance API. Each of your DB Instances can have different preferred maintenance windows, if you so choose.

Running your DB Instance as a Multi-AZ deployment can further reduce the impact of a maintenance event, as Amazon RDS will conduct maintenance via the following steps: 1) Perform maintenance on standby, 2) Promote standby to primary, and 3) Perform maintenance on old primary, which becomes the new standby.

When an Amazon RDS DB Instance deletion API (DeleteDBInstance) is run, the DB Instance is marked for deletion. Once the instance no longer indicates ‘deleting’ status, it has been removed. At this point the instance is no longer accessible and unless a final snapshot copy was asked for, it cannot be restored and will not be listed by any of the tools or APIs.

Event Notification

You can receive notifications of a variety of important events that can occur on your RDS instance, such as whether the instance was shut down, a backup was started, a failover occurred, the security group was changed, or your storage space is low. The Amazon RDS service groups events into categories that you can subscribe to so that you can be notified when an event in that category occurs. You can subscribe to an event category for a DB instance, DB snapshot, DB security group, or for a DB parameter group. RDS events are published via AWS SNS and sent to you as an email or text message. For more information about RDS notification event categories, refer to the RDS User Guide.

Amazon Redshift Security

Amazon Redshift is a petabyte-scale SQL data warehouse service that runs on highly optimized and managed AWS compute and storage resources. The service has been architected to not only scale up or down rapidly, but to significantly improve query speeds even on extremely large datasets. To increase performance, Redshift uses techniques such as columnar storage, data compression, and zone maps to reduce the amount of IO needed to perform queries. It also has a massively parallel processing (MPP) architecture, parallelizing and distributing SQL operations to take advantage of all available resources.

When you create a Redshift data warehouse, you provision a single-node or multi-node cluster, specifying the type and number of nodes that will make up the cluster. The node type determines the storage size, memory, and CPU of each node. Each multi-node cluster includes a leader node and two or more compute nodes. A leader node manages connections, parses queries, builds execution plans, and manages query execution in the compute nodes. The compute nodes store data, perform computations, and run queries as directed by the leader node. The leader node of each cluster is accessible through ODBC and JDBC endpoints, using standard PostgreSQL drivers. The compute nodes run on a separate, isolated network and are never accessed directly.

After you provision a cluster, you can upload your dataset and perform data analysis queries by using common SQL- based tools and business intelligence applications.

Cluster Access

By default, clusters that you create are closed to everyone. Amazon Redshift enables you to configure firewall rules (security groups) to control network access to your data warehouse cluster. You can also run Redshift inside an Amazon VPC to isolate your data warehouse cluster in your own virtual network and connect it to your existing IT infrastructure using industry-standard encrypted IPsec VPN.

The AWS account that creates the cluster has full access to the cluster. Within your AWS account, you can use AWS IAM to create user accounts and manage permissions for those accounts. By using IAM, you can grant different users permission to perform only the cluster operations that are necessary for their work.

Like all databases, you must grant permission in Redshift at the database level in addition to granting access at the resource level. Database users are named user accounts that can connect to a database and are authenticated when they login to Amazon Redshift. In Redshift, you grant database user permissions on a per-cluster basis instead of on a per-table basis. However, a user can see data only in the table rows that were generated by his own activities; rows generated by other users are not visible to him.

The user who creates a database object is its owner. By default, only a superuser or the owner of an object can query, modify, or grant permissions on the object. For users to use an object, you must grant the necessary permissions to the user or the group that contains the user. And only the owner of an object can modify or delete it.

Data Backups

Amazon Redshift distributes your data across all compute nodes in a cluster. When you run a cluster with at least two compute nodes, data on each node will always be mirrored on disks on another node, reducing the risk of data loss. In addition, all data written to a node in your cluster is continuously backed up to Amazon S3 using snapshots. Redshift stores your snapshots for a user-defined period, which can be from one to thirty-five days. You can also take your own snapshots at any time; these snapshots leverage all existing system snapshots and are retained until you explicitly delete them.

Amazon Redshift continuously monitors the health of the cluster and automatically re-replicates data from failed drives and replaces nodes as necessary. All of this happens without any effort on your part, although you may see a slight performance degradation during the re-replication process.

You can use any system or user snapshot to restore your cluster using the AWS Management Console or the Amazon Redshift APIs. Your cluster is available as soon as the system metadata has been restored and you can start running queries while user data is spooled down in the background.

Data Encryption

When creating a cluster, you can choose to encrypt it in order to provide additional protection for your data at rest. When you enable encryption in your cluster, Amazon Redshift stores all data in user-created tables in an encrypted format using hardware-accelerated AES-256 block encryption keys. This includes all data written to disk as well as any backups.

Amazon Redshift uses a four-tier, key-based architecture for encryption. These keys consist of data encryption keys, a database key, a cluster key, and a master key:

- Data encryption keys encrypt data blocks in the cluster. Each data block is assigned a randomly-generated AES- 256 key. These keys are encrypted by using the database key for the cluster.
- The database key encrypts data encryption keys in the cluster. The database key is a randomly-generated AES- 256 key. It is stored on disk in a separate network from the Amazon Redshift cluster and encrypted by a master key. Amazon Redshift passes the database key across a secure channel and keeps it in memory in the cluster.
- The cluster key encrypts the database key for the Amazon Redshift cluster. You can use either AWS or a hardware security module (HSM) to store the cluster key. HSMs provide direct control of key generation and management, and make key management separate and distinct from the application and the database.
- The master key encrypts the cluster key if it is stored in AWS. The master key encrypts the cluster-key-encrypted database key if the cluster key is stored in an HSM.

You can have Redshift rotate the encryption keys for your encrypted clusters at any time. As part of the rotation process, keys are also updated for all of the cluster's automatic and manual snapshots.

Note that enabling encryption in your cluster will impact performance, even though it is hardware accelerated. Encryption also applies to backups. When restoring from an encrypted snapshot, the new cluster will be encrypted as well.

To encrypt your table load data files when you upload them to Amazon S3, you can use Amazon

S3 server-side encryption. When you load the data from Amazon S3, the COPY command will decrypt the data as it loads the table.

Database Audit Logging

Amazon Redshift logs all SQL operations, including connection attempts, queries, and changes to your database. You can access these logs using SQL queries against system tables or choose to have them downloaded to a secure Amazon S3 bucket. You can then use these audit logs to monitor your cluster for security and troubleshooting purposes.

Automatic Software Patching

Amazon Redshift manages all the work of setting up, operating, and scaling your data warehouse, including provisioning capacity, monitoring the cluster, and applying patches and upgrades to the Amazon Redshift engine. Patches are applied only during specified maintenance windows.

SSL Connections

To protect your data in transit within the AWS cloud, Amazon Redshift uses hardware-accelerated SSL to communicate with Amazon S3 or Amazon DynamoDB for COPY, UNLOAD, backup, and restore operations. You can encrypt the connection between your client and the cluster by specifying SSL in the parameter group associated with the cluster. To have your clients also authenticate the Redshift server, you can install the public key (.pem file) for the SSL certificate on your client and use the key to connect to your clusters.

Amazon Redshift offers the newer, stronger cipher suites that use the Elliptic Curve Diffie-Hellman Ephemeral protocol. ECDHE allows SSL clients to provide Perfect Forward Secrecy between the client and the Redshift cluster. Perfect Forward Secrecy uses session keys that are ephemeral and not stored anywhere, which prevents the decoding of captured data by unauthorized third parties, even if the secret long-term key itself is compromised. You do not need to configure anything in Amazon Redshift to enable ECDHE; if you connect from a SQL client tool that uses ECDHE to encrypt communication between the client and server, Amazon Redshift will use the provided cipher list to make the appropriate connection.

Amazon ElastiCache Security

Amazon ElastiCache is a web service that makes it easy to set up, manage, and scale distributed in-memory cache environments in the cloud. The service improves the performance of web applications by allowing you to retrieve information from a fast, managed, in-memory caching system, instead of relying entirely on slower disk-based databases. It can be used to significantly improve latency and throughput for many read-heavy application workloads (such as social networking, gaming, media sharing, and Q&A portals) or compute-intensive workloads (such as a recommendation engine). Caching improves application performance by storing critical pieces of data in memory for low-latency access. Cached information may include the results of I/O-intensive database queries or the results of computationally-intensive calculations.

The Amazon ElastiCache service automates time-consuming management tasks for in-memory cache environments, such as patch management, failure detection, and recovery. It works in conjunction with other Amazon Web Services (such as Amazon EC2, Amazon CloudWatch, and Amazon SNS) to provide a secure, high-performance, and managed in- memory cache. For example, an application running in Amazon EC2 can securely access an Amazon ElastiCache

Cluster in the same region with very low latency. Using the Amazon ElastiCache service, you create a Cache Cluster, which is a collection of one or more Cache Nodes. A Cache Node is a fixed-size chunk of secure, network-attached RAM. Each Cache Node runs an instance of the Memcached or Redis protocol-compliant service and has its own DNS name and port.

Multiple types of Cache Nodes are supported, each with varying amounts of associated memory. A Cache Cluster can be set up with a specific number of Cache Nodes and a Cache Parameter Group that controls the properties for each Cache Node. All Cache Nodes within a Cache Cluster are designed to be of the same Node Type and have the same parameter and security group settings.

Amazon ElastiCache allows you to control access to your Cache Clusters using Cache Security Groups. A Cache Security Group acts like a firewall, controlling network access to your Cache Cluster. By default, network access is turned off to your Cache Clusters. If you want your applications to access your Cache Cluster, you must explicitly enable access from hosts in specific EC2 security groups. Once ingress rules are configured, the same rules apply to all Cache Clusters associated with that Cache Security Group.

To allow network access to your Cache Cluster, create a Cache Security Group and link the desired EC2 security groups (which in turn specify the EC2 instances allowed) to it. The Cache Security Group can be associated with your Cache Cluster at the time of creation, or using the "Modify" option on the AWS Management Console. IP-range based access control is currently not enabled for Cache Clusters. All clients to a Cache Cluster must be within the EC2 network, and authorized via Cache Security Groups.

ElastiCache for Redis provides backup and restore functionality, where you can create a snapshot of your entire Redis cluster as it exists at a specific point in time. You can schedule automatic, recurring daily snapshots or you can create a manual snapshot at any time. For automatic snapshots, you specify a retention period; manual snapshots are retained until you delete them. The snapshots are stored in Amazon S3 with high durability, and can be used for warm starts, backups, and archiving.

Further Reading

https://aws.amazon.com/security/security-resources/

Introduction to AWS Security Processes
Overview of AWS Security - Storage Services
Overview of AWS Security - Database Services
Overview of AWS Security - Compute Services
Overview of AWS Security - Application Services
Overview of AWS Security - Analytics, Mobile and Application Services
Overview of AWS Security – Network Services