Security Best Practices for Managed Oracle Databases on AWS

Amazon RDS for Oracle and Amazon RDS Custom for Oracle

First published August 28, 2024

Last updated August 28, 2024

Notices

Contents

Abstract and introduction

Abstract

Amazon Relational Database Service (Amazon RDS) provides a managed platform on which customers can run a variety of relational databases, including MySQL, MariaDB, PostgreSQL, SQL Server, Oracle, DB2, Amazon Aurora MySQL-Compatible Edition, and Amazon Aurora PostgreSQL-Compatible Edition. This whitepaper outlines best practices for securing these resources from data leaks, deletion, natural disasters, and other calamities, with a focus on Amazon RDS for Oracle and Amazon RDS Custom for Oracle. The target audience for this whitepaper includes database administrators, enterprise architects, systems administrators, and developers who would like to run their database workloads on Amazon RDS.

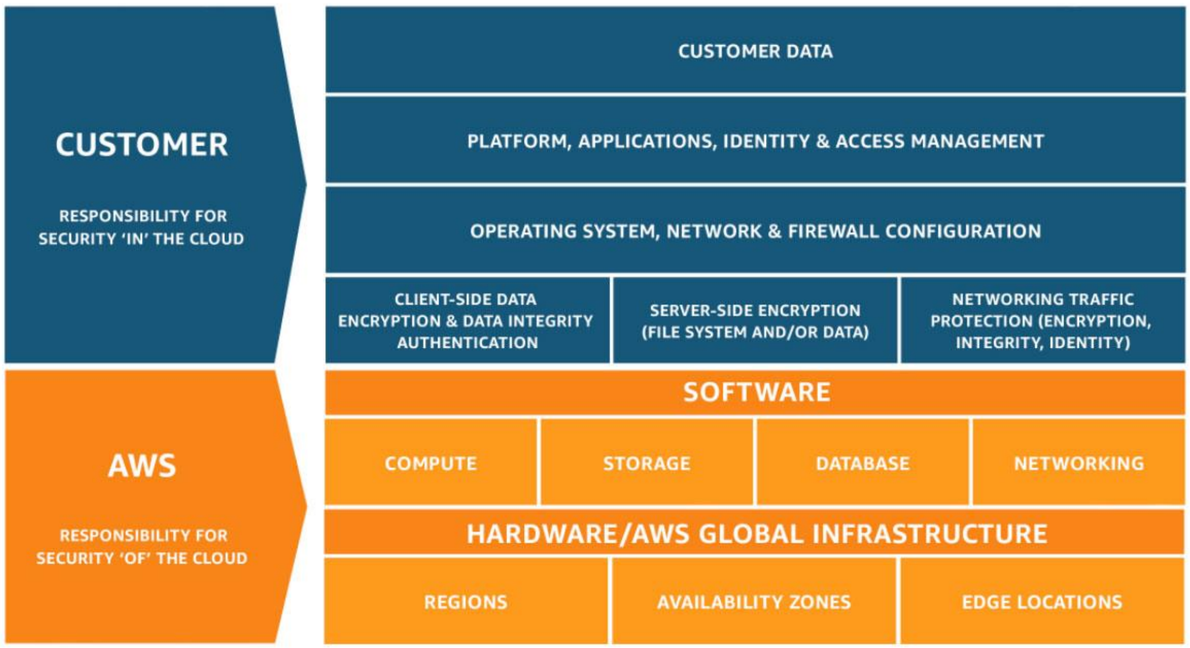
Introduction

Security is a critical facet of operating any information system. Securing data systems in the cloud is a shared responsibility between AWS and its customers. Customers benefit from the work that AWS does to secure the cloud platform on which customers operate, and they can choose to use additional services and features offered by AWS to further secure their data. However, customers must still secure the data systems they deploy in the cloud.

Security is complicated. To help you build and operate secure systems, the security pillar of the AWS Well-Architected Framework provides general guidance and best practices for designing, building, and operating secure systems in the cloud. This document expands upon that guidance for RDS for Oracle and RDS Custom for Oracle by diving specifically into the security features of Oracle Database and the Amazon RDS managed service. We also explore complementary and supporting services like Amazon Elastic Cloud Compute (Amazon EC2), Amazon Virtual Private Cloud (Amazon VPC), AWS Identity and Access Management (IAM), AWS CloudTrail, and Amazon CloudWatch.

Shared responsibility model

Security in the cloud alongside the security of the cloud



AWS implements a shared responsibility model with regard to resources in the cloud. In short, AWS is responsible for the security of the cloud, and customers are responsible for security in the cloud.

Security of the cloud – AWS provides secure global facilities and infrastructure that spans regions, availability zones and edge locations. Layered on top of this infrastructure are compute, storage, database, and networking resources that serve as the foundation for every service offered by AWS. From the data centers, all the way up to the software that manages these services, AWS provides a secure cloud on which our customers can build secure and compliant applications.

Security in the cloud – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your organization's requirements, and applicable laws and regulations. AWS provides a set of features and services to help you secure your data. This paper helps you understand how to apply the shared responsibility model when using Amazon RDS.

Infrastructure

AWS Nitro–based instances

The AWS Nitro System breaks apart the traditional role of the hypervisor by providing dedicated hardware and software to control the CPU, storage, networking, bios, and other physical hardware. The net result is that virtually all server resources go toward running your workload rather than managing a hypervisor.

From a security perspective, the AWS Nitro system provides a specific security chip. This chip is locked down and does not allow administrative or human access, including from Amazon employees. Furthermore, this chip is constantly monitoring the security of the instance hardware and firmware.

More details on the AWS Nitro System and a full list of Amazon EC2 instances that are built on the Nitro system can be found in Instances built on the AWS Nitro System.

IAM permissions and policies

IAM is the cornerstone of resource management on AWS. IAM defines the access permissions granted to entities to create, modify, and delete resources on AWS. For example, in order to create an RDS instance, one must first have the appropriate IAM permissions to do so.

Permissions within IAM are defined using policies. A policy is a document outlining a certain set of operations (create, modify, delete) that can be applied to certain services or resources. Once a policy is defined, that policy can be attached to an IAM identity (user, group, or role). These identities can be assumed by people directly accessing AWS or by other resources on AWS. For example, it is common for an EC2 instance to assume a role that contains one or more policies. Let's say that one of those policies allows for writing files to an Amazon S3 bucket but does not allow for reading from the same bucket. In this case, application code running on that EC2 instance can generate log files and write them to the specified S3 bucket, but the application will not have access to read or delete those files. Similarly, let's say that a person has an AWS CloudFormation template that defines an RDS for Oracle instance. In order to provision that template, the user will need to assume a role associated with a policy that grants permissions to create the instance.

It is of critical importance to secure IAM identities and use restrictive policies. In order to secure IAM identities, it is best practice to only use the AWS account root user to create other users, and then lock away those root credentials. Likewise, when creating policies, it is best practice to only grant the very narrow and specific permissions required to accomplish the task at hand. For example, if you wish to create a policy that allows a user to modify a single RDS instance, you can restrict access to the specific Amazon Resource Name (ARN) of that instance. Or perhaps you would like to grant access to create new RDS instances but not to delete them. This can be controlled with a restrictive policy. This fine-grained access control is in addition to attaching the AmazonRDSDataFullAccess built-in policy to a given identity.

Networking

Network isolation

The main construct available to users to control network access to RDS resources is Amazon VPC. RDS instances exist inside of a virtual private cloud (VPC). With a VPC, one can establish various subnets and define their connectivity to one another and the outside world. In nearly all circumstances, relational databases should be inaccessible from the internet, and they should not have access to network resources outside of the VPC. By adding your RDS instances to private subnets (no direct route to an internet gateway), devices outside of your VPC do not have direct access to your RDS database, and so long as there is no network address translation (NAT) device available to the subnet, your RDS instances do not have direct access to devices outside of your VPC.

VPCs also provide network access control lists (ACLs) to allow you to control network traffic at the subnet level. Traffic can be filtered based on protocol, port, and source.

Another component that allows you to control traffic are security groups. Security groups also allow you to control access at the network level, enabling traffic to be filtered based on protocol, port, and source. However, security groups also have the ability to filter another security group. For example, if you have a fleet of EC2 instances that need to communicate with your RDS instance, you can apply a security group named “ApplicationServers” to those EC2 instances and a second security group named “RdsResource” to your RDS database. In order to allow the EC2 instances to communicate with the RDS instance, you would add a rule to the RdsResource security group that allows ingress from the ApplicationServers security group. Adding a rule is more convenient than limiting access based solely on source IP ranges and provides more flexibility should underlying IP resources change.

Now you have deployed your RDS resources in a private subnet, inaccessible from external network devices, you need to administratively connect to those RDS resources. In this case, the most common approach is to use a bastion host. A bastion host is a compute resource that has access to both RDS resources and the outside world. In this scenario, though the bastion host is publicly accessible, the security group associated with this host should only allow access from known IP ranges. In turn, that security group will be granted access to the RDS resources that are not publicly accessible. From a user’s perspective, one can directly SSH to the bastion host and issue commands from the bastion host. Alternatively, a user can opt to create an SSH tunnel that will allow them to locally use applications on their computer and tunnel the communication through the bastion host.

VPC flow logs

Once controls are put into place to manage network isolation, the next step is to audit network traffic. The primary tool to accomplish this is VPC flow logs. VPC flow logs identify network traffic flow. Flow logs can be useful in diagnosing overly restrictive network ACLs or security groups, or they can help you uncover gaps in your network controls that are allowing traffic that should be prohibited.

VPC flow logs have no impact on network performance and can be published to Amazon CloudWatch Logs, Amazon Simple Storage Service (Amazon S3), or Amazon Data Firehose. Each of these destinations provides a different benefit. Amazon S3 is a cost-effective way to capture VPC flow logs as part of an audit trail that might be infrequently accessed. CloudWatch Logs provides a common logging location, along with other log types, and it is straightforward to query from the AWS Management Console. Firehose is a great tool for ongoing, near real-time analysis of your VPC flow log records.

VPC endpoints for RDS API access

In some circumstances, you might run applications in your VPC that need the ability to provision, modify, or delete RDS resources. For example, you might run an EC2 instance that has assumed a role that gives it the ability to stop RDS instances in your development environment during off hours. Normally, the API call made to stop the RDS instance would need to travel on the public internet. If your EC2 instance is on a private subnet without an NAT gateway, it will not be able to connect to the RDS API to issue the command to stop the instance. By enabling AWS PrivateLink, your EC2 instance can now issue the stop command without being on a public subnet or using an NAT gateway. Traffic remains on the Amazon network, and neither your EC2 instance nor the relevant RDS instance is exposed to the public internet.

Encryption

KMS

One of the most basic necessities of security when working with databases is to encrypt your data. To encrypt data requires encryption keys. The management of those keys is a critical consideration when securing your databases. Fortunately, the AWS Key Management Service (AWS KMS) is specifically designed to create, manage, and rotate encryption keys, and it seamlessly integrates with the RDS platform. AWS KMS allows users to create symmetric and asymmetric keys for encryption and decryption, as well as for HMAC authentication. AWS KMS is not specific to RDS and is integrated with a variety of other AWS services. This integration allows for a unified key management system across the AWS. AWS KMS offers keys that are scoped to a single AWS Region to provide key isolation, as well as keys that can be replicated between Regions for seamless cryptographic functionality around the globe.

Encryption at rest

Encryption at rest is straightforward to implement on the RDS platform. Whenever you create an RDS database instance, you need only check the box that indicates that your volume is to be encrypted, and then select the appropriate key to use for that encryption. From that point forward, the RDS platform will encrypt the entire database volume using the specified key. Snapshots and automated backups created from this volume will also be encrypted using the same key. When selecting a key, you can either choose a customer managed key that you have created or the AWS managed key. Using a customer managed key offers you greater control with regard to key rotation, key material origin, and Region. Additionally, if you are planning to share an encrypted snapshot, it is possible to grant access to a customer managed key, but you cannot grant access to an AWS managed key.

So far we have discussed the encryption of on-disk data. However, cell-level or column-level encryption will depend on the specific database engine you are running on RDS.

Encryption in transit

Please refer to the Encryption in-transit section below under the Oracle Database–specific security recommendations.

Authentication and authorization

Secrets Manager and password rotation

Although there are various ways to authenticate to a database instance, perhaps the most common is using a username and password. Often these credentials are stored unencrypted in code or configuration files. This presents a significant security risk. If your application requires username and password authentication, you should consider storing those credentials in AWS Secrets Manager. Secrets Manager allows you to encrypt your credentials and then access them using an IAM role, as discussed previously in this whitepaper. Encryption ensures that usernames and passwords are never stored in plain text and are not embedded in application code. AWS provides tutorials on how to use Secrets Manager in your organization.

Kerberos authentication

Customers who use Microsoft Active Directory might wish to use Kerberos authentication with their RDS resources. RDS for Oracle and RDS Custom for Oracle both support Kerberos authentication. Kerberos authentication shifts the management of users and passwords to a centralized Microsoft Active Directory, reducing user-management work for database administrators. Kerberos authentication is a more secure approach to credential management than managing standalone usernames and passwords.

Auditing and monitoring

CloudTrail integration

CloudTrail allows users to monitor actions performed in an AWS account. Actions can be initiated by a user, role, or AWS service. Regardless of the source of these actions (console, CLI, SDK), these actions are recorded for future auditing, governance, and compliance of your AWS account. CloudTrail is enabled automatically and does not require any manual setup.

AWS CloudTrail is an important part of RDS security, as it provides an audit mechanism for changes made to database resources. If, for example, an RDS instance is created and you need to know who created that instance, that information can be identified using CloudTrail. Additionally, CloudTrail provides the CloudTrail Insights feature that helps you detect anomalies that CloudTrail discovers, alerting you to a potential security risk.

Event notifications

RDS event notification provides a mechanism by which an Amazon Simple Notification Service (Amazon SNS) topic can be triggered when events related to RDS take place. For example, you might create an SNS topic that sends an email when triggered. You could then tie this SNS topic to an RDS event that is invoked when a database instance has an availability issue such as a shutdown or restart. Alternately, you can use an AWS Lambda SNS endpoint, so that programmatic action takes place in response to the invoked event.

CloudWatch metrics and alerting

Amazon CloudWatch metrics are a critical component of managing any RDS database. Although most RDS metrics are performance related, when certain performance metrics such as CPU increase beyond what is expected, it can be an indication of a bigger problem. CloudWatch metrics provide an alerting feature called Amazon CloudWatch alarms. These alarms can be configured like RDS events to invoke an SNS topic in response to CloudWatch metrics crossing specified thresholds.

Publish database logs to CloudWatch

RDS databases capture a variety of logs. Those logs vary by engine and are stored by default on storage local to the database instance. When working with a large number of database instances, it is not practical to connect to each instance to review its logs. Furthermore, processing the logs to find patterns or anomalies is also not a trivial undertaking. Fortunately, RDS offers the ability to publish database logs to Amazon CloudWatch Logs. By sending your logs to CloudWatch Logs, you now have a single, centralized repository to view logs. What's more, CloudWatch Logs offers the ability to query your logs through CloudWatch Logs Insights and detect anomalies in your logs using a log anomaly detector.

Database Activity Streams

Database Activity Streams in Aurora and RDS sends a near real-time feed of database audit events to Amazon Kinesis Data Streams for consumption by monitoring and compliance tools. Third-party tools like IBM's Security Guardium and Imperva's SecureSphere Database Audit and Protection can consume audit data from Kinesis Data Streams to monitor events occurring in the database. You can also build your own Kinesis data stream to process audit events. Activity streams are always encrypted. Database administrators don't have access to create, manage, or monitor activity streams unless they're explicitly granted those privileges through IAM. Once privileges are granted, you can monitor internal threats to your data systems.

Configuration

Master user

The master user is the closest login to root provided on an RDS instance. This user is created at the same time as an RDS instance and has the highest level of permissions available for the RDS instance. It is a recommended best practice to provide a strong password for this login, create other users with some subset of the master user's permissions, and then store the master user credentials in a secure location. This practice helps ensure the principle of least privilege and keeps your database secure.

Parameter groups

RDS parameter groups allow you to create a set of parameters that can be applied to one or many RDS instances of the same database engine type. For example, you might create a parameter group for Oracle Database 19c Enterprise Edition (EE), which can then be applied to the 19c EE instances in your account in the specified Region. You benefit from centralized management and the standardization of parameters. Inside of the parameter group, you can configure engine-specific security features. These features are explored further later in this whitepaper.

Patch management

Patch management is critical to database security. Over time, new patches are released for any given database engine that can contain performance or stability improvements, but often contain additional security fixes. In the case of RDS, patches are applied at the OS level and the database-engine level. In most cases, it is best practice to enable the Automatic Minor Version Upgrade feature. This feature ensures that your database instances are always kept up to date with the latest security patches. In order to minimize disruption to existing workloads, the minor version upgrade happens during your specified maintenance window.

Operating system updates come in two varieties. The first is optional updates. Get notified when an optional operating system update is available by using the RDS event functionality discussed earlier and specifically subscribing to RDS-EVENT-0230.

The second category of operating system updates is the mandatory category. Unlike optional updates, which can be skipped at your discretion, mandatory updates come with an apply date. If you do not apply the update by that date, it will be automatically applied during your specified maintenance window.

Oracle Database–specific security features

This section of the whitepaper discusses features specific to RDS for Oracle and RDS Custom for Oracle that meet the security and compliance requirements of Oracle Database workloads hosted on AWS. Features that are not available in Oracle Standard Edition 2 (SE2) and that require additional licensing are outlined in the Oracle licensing guide. However, you must validate your licensing contract with Oracle to confirm that use of products and features discussed in this whitepaper does not violate license terms or conditions.

Network isolation

In addition to securing RDS instances in a VPC using security groups and network access control lists (ACLs), the following Oracle Database features provide additional network-level security for your instances.

ACLs

When accessing external resources from Oracle Database, it is recommended that you create fine-grained access control policies that allow specific users to access a defined list of external resources. Using ACLs, Oracle Database allows access restrictions to external network services using PL/SQL APIs such as UTL_TCP, UTL_SMTP, UTL_HTTP, and more. DBMS_NETWORK_ACL_ADMIN provides various APIs to control access to external network resources from an RDS for Oracle instance. This feature is available on RDS for Oracle and RDS Custom for Oracle without additional cost in both editions. If you would like to prevent outbound network access that is initiated from the database layer, you can disable egress traffic in the security group assigned to the database instance.

Database firewall

You might want to implement additional protection from security threats, such as SQL injection attacks, with a SQL grammar-based analysis (for example, by comparing against an approved list of application SQLs). Oracle Audit Vault and Database Firewall (AVDF) provides a database firewall that monitors activity or blocks SQL statements on the network, based on a firewall policy. You can configure the database firewall for monitoring and blocking or only for monitoring. To implement monitoring and blocking, you must configure the firewall in proxy mode, where database traffic is routed through the Database Firewall. Oracle Database Firewall monitors data access, enforces access policies, highlights anomalies, and helps protect against network-based attacks originating from outside or inside the organization.

RDS for Oracle does not support AVDF because of its restricted access to the operating system. However, you can use AVDF features with RDS Custom for Oracle, which provides the flexibility to install additional software components on the operating system. AVDF comprises two product components that are licensed together as one product.

The following table summarizes the network-level security features for RDS for Oracle and RDS Custom for Oracle.

Feature	Availability in RDS for Oracle	Availability in RDS Custom for Oracle
Security groups	Yes	Yes
Network ACL	Yes	Yes
Database ACL	Yes	Yes
Database Firewall	No	Yes

Encryption

To protect your sensitive data stored in the database, it is vital to ensure that the data is encrypted while it is stored in the database files and backups as well as while it travels across the network between the database and clients.

Encryption at rest

By default, Oracle Database stores data in the underlying physical data files unencrypted. Bad-faith actors could potentially access physical data files or backups and use a hex editor to access sensitive data, bypassing security enforcements of the database layer. It is therefore crucial to encrypt data at rest to protect your sensitive data. We previously discussed encrypting data at rest using AWS KMS. You can also choose to encrypt your database using Oracle Transparent Data Encryption (TDE). TDE is an Oracle EE feature that allows encryption of a subset of the data (column level or table level). Encryption with AWS KMS is available for both EE and SE2 and encrypts your entire RDS storage. Refer to Oracle Database Encryption Options on RDS for more information on available encryption options in RDS for Oracle.

KMS encryption	TDE
Available for all editions	EE-specific, with an additional license for Oracle Advanced Security
Encrypts entire RDS instance storage (Amazon EBS volumes)	Supports granular encryption at column and tablespace levels
Easier to set up and manage	Not all TDE features are supported on RDS for Oracle. For example, customers have no access to a wallet or password.
Transparent to application with minimal performance overhead	Transparent to application with minimal performance overhead

RDS Custom for Oracle encrypts underlying Amazon Elastic Block Store (Amazon EBS) volumes using encryption in AWS KMS. Additionally, RDS Custom for Oracle instances can use TDE encryption.

TDE and encryption in AWS KMS can be simultaneously enabled for instances to meet specific security and compliance requirements. However, this configuration needs to be tested well to assess the performance impact of double encryption.

Encryption in-transit

Encryption in-transit secures the communication between the Oracle Database server and client from various security threats. Examples of such threats include an unauthorized party intercepting data in transit, altering it, and retransmitting it (data modification attack) or the repetitive retransmission of an entire set of valid data (replay attack). To encrypt the communication between an Oracle Database on AWS and the application and clients, you can either use secure sockets layer (SSL) encryption or Oracle native network encryption (NNE). The table below shows a quick comparison of these two options.

SSL-based network encryption	NNE
Based on industry standard SSL/TLS protocol	Oracle proprietary encryption method

-- Complex setup with certificates -- Overhead to manage the certificate, which eventually expires -- Requires changes to client and server	Easier setup as no certificates involved
Stronger security posture compared to NNE	Lesser performance overhead compared to SSL
Provides nonrepudiation for server connections to prevent third-party attacks	No non-repudiation of the server connection, which means no protection against third-party attacks
-- Connections to SSL-enabled listener port must be encrypted -- Need to maintain two listeners if some clients do not want to participate in the network encryption	Can be configured to allow clients to fall back to unencrypted traffic when NNE is not supported by the client

SSL/transport layer security (TLS) connections provide a layer of security by encrypting data that moves between your client and database instance. Optionally, your SSL/TLS connection can perform server identity verification by validating the server certificate installed on your database instance. With SSL/TLS–based encryption, the client and server perform a TLS handshake before authentication:

- The client and server establish which algorithm is to be used for encryption
- The server sends its certificate to the client, and the client verifies that the server's certificate was signed by a trusted certificate authority (CA). This step verifies the identity of the server. Similarly, the client sends its own certificate to the server, and the server verifies that the client's certificate was signed by a trusted CA.
- The client and server exchange key information using public key cryptography. Based on this information, the client and server each generate a session key that is used for data encryption for the duration of a single communication session.
- If the handshake is successful, then connection is accepted, and the communication is encrypted between the database and client using the session key.

You can conveniently enable SSL/TLS–based encryption in RDS for Oracle using option groups. RDS makes certificate management tasks easier. Refer to Using SSL/TLS to encrypt a connection to a DB instance to learn more about how certificates are managed for SSL/TLS–based encryption in RDS for Oracle. When SSL-based encryption is enabled on RDS for Oracle, RDS uses a second SSL-enabled listener port for SSL connections. You can still use the default port for clear text communications (typically from clients within the VPC) and the SSL port for encrypted communications. You can also block connections to the default listener to prevent unencrypted communications by adjusting the security group rules for the instance.

With NNE, when a connection is made, the server selects which algorithm to use, if any, from algorithms specified in the sqlnet.ora configuration. The server searches for a match between the algorithms available on both the client and the server and picks the first algorithm in its own list that also appears in the client list. If one side of the connection does not specify an algorithm list, all the algorithms installed on that side are acceptable. For RDS for Oracle, option groups streamline configuration of NNE and option group settings allow you to configure various NNE-related parameters, such as the encryption algorithm to be used. In RDS Custom for Oracle, you can configure NNE, as in the case of self-managed databases, because you have direct access to configuration files.

SSL/TLS and NNE are no longer part of the Oracle Advanced Security option. You can choose to enable either SSL-based or NNE encryption for an RDS for Oracle instance with all licensed editions using option groups, but the encryption methods cannot be used together. RDS Custom for Oracle allows the use of SSL/TLS–based encryption and NNE at the same time by manually configuring the IGNORE_ANO_ENCRYPTION_FOR_TCPS SQL*Net parameter.

Encrypted backups

It is as important to keep the backup files encrypted as it is the physical database files. RDS for Oracle and RDS Custom for Oracle use Amazon EBS snapshots for taking automated and manual snapshots of the database instance. The snapshot-based backup preserves the encryption setting of the instance for encryptions with both KMS and TDE. If the instance is encrypted, the backup remains encrypted too. However, you might choose additional backup and data extraction tools for various requirements, in which case you need to use encryption mechanisms supported by those respective tools (for example, RMAN backup encryption or expdp encryption support)

The following table summarizes encryption features available for RDS for Oracle and RDS Custom for Oracle.

Feature	Availability in RDS for Oracle	Availability in RDS Custom for Oracle
Encryption in transit using SSL	Yes: RDS streamlines configuration with an option group	Yes: to be customized and managed by customers
Encryption in transit using NNE	Yes: RDS streamlines configuration with an option group	Yes: customer managed
Encryption at rest using AWS KMS	Yes: RDS simplifies encryption key management	Yes: mandatory. RDS simplifies encryption key management.

Encryption at rest using TDE	Yes: EE only, RDS simplifies wallet management	Yes: customer is responsible for wallet management

Auditing

Because of multiple compliance requirements and rising security threats, security auditing has become increasingly important. Security auditing is an effective method of enforcing strong internal controls that allow you to a) monitor business operations to find any activities that deviate from company policy and b) meet various regulatory compliance requirements. Security auditing allows you to record the activity on the database for future examination and is one part of an overall database security strategy. In this section we briefly cover auditing options available for Oracle databases on AWS.

RDS actions (API calls) are logged by AWS CloudTrail. This logging allows you to track AWS resource-level operations (for example, answering questions such as who stopped a production instance).

You might have questions of your database activities, such as who updated the salary column of an employee table. To answer such questions and meet your auditing requirements, you can enable auditing of database activities using Oracle standard auditing, unified auditing, or fine-grained auditing. RDS for Oracle supports mixed-mode unified auditing, while RDS Custom for Oracle supports both mixed-mode and pure-mode unified auditing. In RDS for Oracle, the AUDIT_TRAIL parameter is set to NONE in the default parameter group. To enable auditing in RDS for Oracle, set the parameter to one of the values in the following table by creating a custom parameter group and changing the parameter value for that group. When audit records are stored at the operating system level, RDS for Oracle also supports integration of audit files with CloudWatch Logs.

Parameter value	Meaning in Amazon RDS for Oracle	Integrated with CloudWatch logs ?
DB	Directs all audit records to the database audit trail (sys.aud$), except for records that are always written to the operating system audit trail	No
DB,EXTENDED	Does all the actions of AUDIT_TRAIL=DB and also populates the SQL Bind and SQL text columns of the SYS.AUD$ table	No
XML	Directs all audit records in XML format to an operating system file	Yes
XML,EXTENDED	Does all the actions of AUDIT_TRAIL=XML, adding the SQL Bind and SQL Text columns	Yes
OS	Directs all audit records to an operating system file	Yes

Audit trails generated in the database when the audit_trail parameter is set to DB and DB,EXTENDED need to be removed using DBMS_AUDIT_MGMT. Audit trails generated at the OS level will be automatically purged by RDS with a configurable retention.

In RDS Custom for Oracle, you have full access to various auditing configurations and parameter settings to customize the instance to meet your specific requirement.

Database alert.log, listener.log, and trace files might be useful when investigating security issues (for example, querying who relaxed the TNS settings to allow unencrypted traffic). You can configure RDS for Oracle to publish alert.log and listener.log to CloudWatch Logs for longer retention and analysis. For more information about various logs in RDS for Oracle, see Oracle Database Log files.

When analyzing the root cause of past incidents, the Oracle LogMiner feature can be used to mine transaction logs to learn about the history of activities in the database (for example, checking if a data-removal job got executed with the wrong parameters).

Database Activity Streams integrates with Oracle unified auditing to provide a near real-time stream of audited activities. As previously discussed, the stream is pushed to Kinesis Data Streams and can be consumed by event-driven applications and integrated with third-party database compliance solutions. Database Activity Streams facilitates the separation of duties between security and compliance personnel and database administrators. It can be used to build a centralized auditing store based on Amazon S3 for your Oracle databases, which can be queried using analytical services such as Amazon Athena. Database Activity Streams is currently available only for RDS for Oracle (non-multitenant architecture) and not available for RDS Custom for Oracle.

Refer to the blog series Security Auditing in RDS for Oracle for further details on auditing options in RDS for Oracle.

When assigning permissions to database users, implement the least-privilege policy to ensure users have only the minimum permissions necessary to perform their task. You can use the DBMS_PRIVILEGE_CAPTURE API on RDS for Oracle and RDS Custom for Oracle to record the usage of privileges granted to users. This information can then be used to revoke unused user privileges. This feature is not available in SE2.

With the flexibility offered by RDS Custom for Oracle, you can choose to implement any auditing options on RDS Custom for Oracle instances. Additionally, audit logs from RDS Custom for Oracle instances can be integrated with the Oracle Audit Vault.

The following table summarizes the auditing options in RDS for Oracle and RDS Custom for Oracle.

Feature	Availability in RDS for Oracle	Availability in RDS Custom for Oracle	Scope or granularity

CloudTrail	Yes	Yes	API calls at AWS resource level
Traditional standard auditing	Yes	Yes	Database transactions and logins
Fine-grained auditing	Yes: only on EE	Yes	Auditing based on fine-grained conditions
Unified auditing	Yes: only mixed mode	Yes	Database transactions and logins
Audit log integration with CloudWatch	Yes: native integration for audit logs generated at OS level	No native integration	Stream the auditing information for longer retention and analysis
Database Activity Streams	Yes	No	Segregation of duties, streaming audit data
LogMiner	Yes	Yes	Root cause analysis of database events
Audit Vault	No	Yes	Centralized auditing store
Privilege analysis	Yes: only on EE	Yes	Apply least-privilege permissions

Authentication and authorization

To enforce the security of a managed service, RDS for Oracle does not provide access to the Operating System of the database host or to SYS/SYSTEM users at the database layer. The RDS master user has maximum user permissions in an RDS for Oracle instance, and this user can be used for all administrative requirements, including creating and managing other required users. If your application requires the SYSDBA role, SYS/SYSTEM user access on the database layer, or shell access on the database host, then RDS Custom for Oracle would be the preferred deployment option, as you have full access to the database and host. It is recommended to follow best practices when creating and managing database users, such as using strong password management policies and controlling access using the principle of least privilege (assigning the minimum permissions required to perform a task).

Oracle databases support various authentication mechanisms, from straightforward password-based authentication to advanced authentication mechanisms such as Kerberos and RADIUS. RDS for Oracle supports Kerberos authentication (in both editions) to provide the benefits of single sign-on and centralized authentication of Oracle Database users. With this feature, you can allow your database users to authenticate against RDS for Oracle using either the credentials stored in the AWS Directory Service for Microsoft Active Directory or the credentials stored in your on-premises Microsoft Active Directory. A forest trust relationship is established between your on-premises Microsoft Active Directory and AWS Managed Microsoft AD. Authorization happens locally within the database in RDS for Oracle.

The flexibility of RDS Custom for Oracle offers a wider range of authentication methods, including central authorization. Refer to the blog series Enable Kerberos Authentication with RDS Custom for Oracle to configure Kerberos authentication and Oracle Centrally Managed Users (CMUs) in RDS Custom for Oracle.

The following table summarizes the various authentication mechanisms available in RDS for Oracle and RDS Custom for Oracle.

Feature	Type of authentication and authorization	Availability in RDS for Oracle	Availability in RDS Custom for Oracle
Password authentication	Local authentication and authorization	Yes	Yes
Kerberos authentication	Central authentication and local authorization	Yes: only with AWS Managed Microsoft AD	Yes
RADIUS	Central authentication and local authorization	No	Yes
OS authentication	Central authentication and local authorization	No	Yes
CMUs	Central authentication and authorization	No	Yes
Oracle Enterprise User Security	Central authentication and authorization	No	Yes

Access control

Oracle Database supports various access control mechanisms to protect your sensitive data in RDS for Oracle and RDS Custom for Oracle.

Oracle Data Redaction allows you to redact sensitive data in real time without masking the data stored in the database. A common use case is redacting credit card data in the application screen used by call center agents. Redaction happens while presenting the data to users, based on the configured policies. During the time that data is being redacted, data processing proceeds as normal and back-end referential integrity constraints are preserved. Data Redaction helps you comply with industry regulations such as the Payment Card Industry Data Security Standard (PCI DSS) and the Sarbanes-Oxley Act. Oracle Data Redaction is an EE-specific feature licensed under Oracle Advanced Security.

Oracle Label Security (OLS) allows you to implement differing access requirements for data in a database through multilevel access controls based on the classification of the data and the access label of the application user. OLS provides centralized application security with hierarchical access controls to sensitive projects. OLS is an extra-cost option available only in the EE. It is an out-of-the-box solution for row-level security with no coding or software development required, allowing administrators to focus on policies. OLS provides an interface for creating policies, specifying enforcement options, defining data-sensitivity labels, establishing user label authorizations, and protecting individual tables or schemas. OLS is best suited for situations where access control decisions are based on the sensitivity of the information. OLS is used in military and intelligence organizations where row-level data classification is used to enforce access restrictions. For example, information with a certain grade of sensitivity can only be seen by users that fall into a specific job category.

Oracle Virtual Private Database (VPD) allows you to filter users by generating a dynamic predicate for SQLs based on the user context. Create policies to control database access at the row and column level. VPD is an EE-only feature. Though VPD and Label Security address similar requirements, VPD provides more flexibility in controlling data access requirements by comparing the value of an attribute in an individual row with an application context value.

Oracle Database Vault can restrict database administrators from accessing sensitive application data. It helps to reduce the risk of insider and outside threats and address compliance requirements, including separation of duties. Database Vault is an EE-only feature available only in RDS Custom for Oracle.

The following table summarizes various access control options in RDS for Oracle and RDS Custom for Oracle.

Feature	Availability in RDS for Oracle	Availability in RDS Custom for Oracle
Data Redaction	Yes: EE only, licensed under Oracle Advanced Security	Yes: licensed under Oracle Advanced Security

OLS	Yes: EE only, extra-cost option	Yes: EE only, extra-cost option
VPD	Yes: EE Only	Yes: EE Only
Database Vault	No	Yes

Miscellaneous security features

Security patches

Prompt application of security fixes is crucial to keep your data secure. Oracle releases critical security patches as Release Updates (RUs) every quarter that include fixes for various security vulnerabilities. RDS for Oracle makes those RUs available within a few weeks of their general availability.

Note: RDS for Oracle mandates Automatic Minor Version Upgrade for instances that have the Java Virtual Machine (JVM) option added.

In RDS Custom for Oracle, you can apply RUs as well as one-off patches at your convenience.

Data masking and subsetting

Masking or exclusion of sensitive data is important to consider when sharing production data with lower environments that might not be protected with all security enforcements. Oracle Data Masking and Subsetting extracts entire copies or subsets of application data from the database and masks sensitive data, so it can be safely shared for nonproduction use. This is an extra-cost option available only in EE. Data Masking and Subsetting is available in RDS Custom for Oracle. However, it is not available on RDS for Oracle due to access restrictions.

Security assessment and discovery of sensitive data

Your database configuration will need to meet the security and compliance posture defined by your organization. Automatic discovery of sensitive data is a requirement for many organizations. The Oracle Database Security Assessment Tool (DBSAT) enables organizations to maintain a strong security and compliance posture for their Oracle databases, as well as facilitating automatic discovery of sensitive data. DBSAT is available in RDS Custom for Oracle. However, it is not available for RDS for Oracle due to access restrictions.

Conclusion

This whitepaper outlines the most important security best practices for deploying and protecting RDS for Oracle and RDS Custom for Oracle instances. By using RDS and Oracle database security features for data access control, data encryption, auditing and monitoring, organizations can safely run their critical Oracle database workloads in AWS.

Contributors

Contributors to this document include:

- Jobin Joseph, Senior Database Specialist Solutions Architect, AWS
- Steve Abraham, Principal Database Specialist Solutions Architect, AWS

Further reading

For additional information, refer to:

- AWS Architecture Center
- Security in Amazon RDS
- Security Pillar of the AWS Well-Architected Framework
- Amazon RDS for Oracle
- Amazon RDS Custom for Oracle

Document revisions

Date	Description
August 28, 2024	First publication