

Oracle SQL Firewall Frequently Asked Questions

SQL Firewall is built into Oracle Database 23ai. This FAQ addresses questions you may have about this exciting new capability.

August, 2024, Version [\[1.0\]](#)

Copyright © 2024, Oracle and/or its affiliates

Public

Table of contents

PRODUCT OVERVIEW	4
What is Oracle SQL Firewall?	4
How does SQL Firewall work?	5
What are the key differentiators of SQL Firewall over other firewalls?	5
I have a network firewall. Why do I need SQL Firewall?	7
I have Database Vault. Why do I need SQL Firewall?	8
What is the performance impact of SQL Firewall?	8
Does SQL Firewall inspect encrypted SQL traffic?	8
Can SQL Firewall be used for DBAs and other ad-hoc users?	8
I have Oracle Audit Vault and Database Firewall (AVDF). How do I decide which one to use: SQL Firewall (or) Database Firewall?	9
I have SQL Firewall. Why do I need database auditing?	10
ACCELERATE COMPLIANCE	10
How does SQL Firewall help accelerate compliance?	10
CONFIGURATION	10
How do I administer SQL Firewall?	10
Can I enable SQL Firewall in the CDB?	11
Can I enable SQL Firewall in the PDB?	11
How does SQL Firewall know which statements are authorized?	11
Can I modify the allow-lists once generated and enforced?	11
Can I reuse the allow list for a user across different databases?	11
Can I reuse the allow list for a different user on the same database?	11
Is it possible to have multiple policies enforced at the same time for the same user?	12
How can SQL Firewall help prevent zero-day SQL Injection attacks?	12
What happens to applications whose statements are blocked by the SQL Firewall? Does the connection terminate?	12
What are SQL Firewall violations, and how can I monitor them?	12
Can SQL Firewall track violations on only untrusted database connection paths?	12
How do I extend beyond what I initially captured for allowed IP addresses?	12
How does SQL Firewall identify unique SQL statements for its allowed list?	13
When would you typically want to set TOP_LEVEL to false?	13

Are SQL Firewall violations audited?	13
Do I need to purge SQL Firewall violation logs?	13
Can I use SQL Firewall for ADMINISTRATIVE users like SYSDBA?	13
How can I configure strict enforcement of allowed SQLs in the primary while relaxing the enforcement in Active Data Guard (ADG) standby?	14
Can I configure SQL Firewall protection for a particular schema?	14
Does SQL Firewall work with Real Application Security (RAS)?	14
LICENSE	14
Is SQL Firewall available in Oracle Database Standard Edition?	14
Is SQL Firewall available in Oracle Database Free Offering?	14
How is SQL Firewall licensed?	15
Is SQL Firewall licensed to use on DBaaS offerings in Oracle Cloud?	15
MORE INFORMATION	15
Where can I get hands-on experience using SQL Firewall?	15
Where can I see product demo videos of SQL Firewall?	15
Are there demo scripts available for customers to try hands-on in their environment?	15
Where can I post my queries on this feature?	16
Where is the documentation reference?	16

With Oracle Database 23ai's SQL Firewall, organizations gain a powerful tool to combat the risk of SQL injection and block the misuse of stolen credentials. With SQL Firewall, you can significantly bolster your resilience against SQL injection attacks, protect sensitive data, and preserve the integrity of your databases. SQL Firewall ensures that only authorized SQL statements from trusted database connections are permitted for execution inside the Oracle Database while blocking and logging unauthorized SQL or database connections.



Legacy solutions like Web Application Firewalls (WAFs) often fail to mitigate SQL Injection risks since WAFs can be bypassed, require complex implementation, and, because they rely on signature and pattern recognition, are seldom proof against zero-day attacks. Because SQL Firewall is built into the Oracle Database kernel, it cannot be bypassed. The kernel-resident SQL Firewall has access to the full run-time context required for analyzing SQL queries, and hence cannot be fooled by the use of synonyms or dynamic SQL, and is not impacted by network encryption.

Key Business Benefits

- Protection against zero-day SQL injection attacks
- Protection against compromised or misused service account credentials

For those prioritizing security against SQL Injection threats, SQL Firewall in Oracle Database 23ai is indispensable.

PRODUCT OVERVIEW

What is Oracle SQL Firewall?

SQL Firewall is a database security feature built into the Oracle Database 23ai kernel that inspects ALL incoming database connections and SQL statements and allows/logs/blocks unauthorized activity following database users-specific policies. SQL Firewall ensures that only explicitly authorized SQL is executed, blocking attempts at SQL injection.

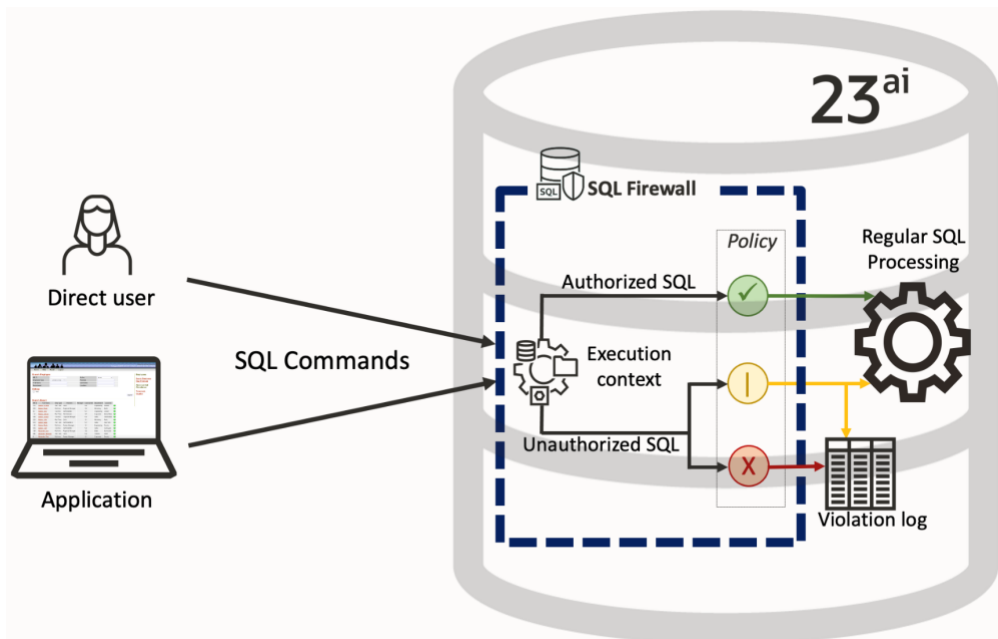


Figure 1. SQL Firewall built into Oracle Database kernel

SQL Firewall uses an allow-list approach, defining a finite set of allowable behaviors instead of attempting to guess the near-infinite choices an attacker might use to break into the database. SQL Firewall works by learning normal SQL traffic for a database user, including what SQL statements are expected from that user and the context the user uses to connect to the database, such as network address, the operating system user, and the program used. Once trained, SQL Firewall can do the following:

- Log and block deviations from normal behavior
- Identify unusual SQL statements
- Identify connections coming from addresses or programs not in the allow-list

How does SQL Firewall work?

SQL Firewall works by learning normal application behavior, including what SQL statements an application issues and the context an application uses to connect to the database, such as network address, operating system user, and program used. You train the SQL Firewall to learn the normal SQL statements and database connections you wish to protect per database user. The normal expected SQL statements and database connections for the database user constitute the allow-list baseline for that user and are represented as SQL Firewall policy.

When the policy with an allow-list baseline is enforced, any SQL statements or database connections originating from that user that deviates from the baseline will be logged and can optionally be blocked.

What are the key differentiators of SQL Firewall over other firewalls?

SQL Firewall protection cannot be bypassed:

Unlike the other firewalls, SQL Firewall runs inside the database kernel. The protection operates at the layer closest to where the data resides, shifting the enforcement point from the network into the database kernel. This ensures that SQL Firewall has comprehensive visibility into ALL SQL traffic irrespective of where it originates, without exceptions—whether local or over the network, encrypted or clear text.

SQL Firewall takes advantage of full run-time context for enforcement decisions:

Unlike firewalls operating outside the database, SQL Firewall has visibility into the full run-time execution context required for comprehensively analyzing SQL queries. This context is what the database builds while running any SQL statement, including steps to resolve synonyms, object names, or dynamically generated names, and current user context. The ability of

SQL Firewall to understand the SQL’s intent with its full execution context to make an enforcement decision gives it a competitive edge over most common firewalls.

What are the other technical differentiators of SQL Firewall over most competitive solutions?

Technical differentiators	Customer benefit	How we deliver that benefit	How we differ from the competition
SQL-based control flexibility	SQL Firewall protects Oracle Databases from SQL injection attacks, including zero-day attacks, resulting from application vulnerabilities.	SQL Firewall compares incoming SQL statements against an allow-list of acceptable statements collected by training the firewall with the customer’s application. If an unfamiliar SQL statement is encountered, a violation is generated and the statement can optionally be blocked.	Unlike other database activity monitoring solutions, SQL Firewall is built into the database. Because it is implemented in the database kernel, SQL Firewall can capture both top-level and recursive SQLs.
No need to install additional hardware or modify network configuration	SQL Firewall is easier to implement than solutions that require additional hardware or require routing database traffic through the external security system in order for it to block. And unlike some third-party solutions, SQL Firewall does not “hack” into database memory, destabilizing the database and risking data corruption.	SQL Firewall is built into the Oracle Database, not a bolt-on solution. That means it can’t be bypassed, requires no separate installation – customers merely need to enable the feature with a PL/SQL command and begin creating policies.	SQL Firewall operates within the Oracle Database, and is tested and certified for all deployment options, including Autonomous Database, Exadata, Exadata Cloud Service, Exadata Cloud at Customer, Base Database, and on-premises. Other solutions either operate at the network level, routing database traffic through their service for inspection and enforcement or use an agent that attempts to monitor and enforce rules by reading activity from database process memory or local network monitoring. These bolt-on solutions rarely work in all architectures, can cause system instability, add additional points of failure, and can complicate patching and upgrade scenarios.
Highly performant	SQL Firewall runs inside the database with less than 2% performance overhead for most workloads.	SQL Firewall is part of the database statement execution path introducing no additional latency.	Competing solutions run as agents on the host or on separate machines and can introduce latency and require significant resources.

Technical differentiators	Customer benefit	How we deliver that benefit	How we differ from the competition
		SQL Firewall policies are cached in kernel memory and have direct access to the SQL signatures in database cursors. SQL Firewall also uses advanced database features like fast-ingest to capture violations without impacting I/O resources for other database operations.	
Session context-based control flexibility	SQL Firewall protects from rogue users attempting to exploit application accounts to connect to the database	SQL Firewall can be configured to block database connections based on session context variables such as IP address, OS user, and client program.	Other database firewalls only block on limited variables such as hostname or IP address.
Supports applications everywhere	SQL Firewall provides protection for Oracle Database-based applications everywhere in the organization; on-premises, in the Oracle Cloud, and in third-party clouds.	SQL Firewall is built into Oracle Database 23ai and is available regardless of how that database is deployed. For example, if a customer moves their application from on-premises to the Oracle Cloud, SQL Firewall is available, and their SQL allow lists can move with the application.	If the customer is using a third-party product, they will need to re-deploy their solution when they move their application. Other database vendors support SQL Firewall-like features; however, these may only be available for certain deployment options or restricted to cloud databases.
Superior manageability	Oracle Database customers have flexibility in how they implement and manage SQL Firewall depending on their use cases.	SQL Firewall supports PL/SQL interfaces, Data Safe GUI console, CLIs and RESTful APIs.	Data Safe console provides visibility and configurability for DBA users, along with other essential database security features. Because SQL Firewall is part of the database, there is no need for a separate team with different skillsets to manage it and no additional workload needed for patches and upgrades.

I have a network firewall. Why do I need SQL Firewall?

Detecting and preventing SQL injection attacks is crucial for safeguarding databases from unauthorized access and potential data breaches. Legacy approaches to using network firewalls offer no respite to mitigating SQL Injection risks, especially zero-day attacks. SQL injection continues to be one of the most popular attack patterns on the Open Web Application Security Project (OWASP) list of the [top ten threats](#) to application security since 2017. Industry-wide CVE security vulnerabilities for SQL injection have shown an alarming increase since 2021, as detailed [here](#).

Moving the protection closer to where the data resides is a strategic approach to mitigating SQL Injection risk. The SQL Firewall built into the Oracle Database kernel makes it impossible to bypass this security control. It helps inspect all incoming SQL statements and database connections for potential SQL Injection attacks. All deviations from the authorized statements and connections are inspected, thereby significantly bolstering your resilience against SQL injection attacks, protecting sensitive data, and preserving the integrity of your databases.

In most practical scenarios, you will want to take a dual-control approach, augmenting traditional network firewall capabilities to block based on a deny-list approach using SQL Injection attack patterns, with SQL Firewall's allow-list capability based on actual application SQL statements and explicitly defined connection paths.

I have Database Vault. Why do I need SQL Firewall?

Database Vault was first available with Oracle 9i release 2. Database Vault provides strong security controls to prevent unauthorized access to sensitive information by privileged users and protect against unintended changes to the database. Using realms, command controls and trusted paths, you can block access to sensitive objects, block execution of critical commands, or block SQL connections from untrusted factors such as time of the day, IP address, hostname, program name, or any number of identifiable attributes associated with the user.

However, the legacy Database Vault constructs like realms or command rules do not discriminate between approved/authorized SQL statements and unexpected/ unauthorized SQL statements. SQL Firewall lets you capture the permitted list of SQL statements with associated trusted database connection paths and log/block the unseen SQL traffic.

SQL Firewall is included with Database Vault, beginning with Oracle Database 23ai. Consider leveraging the capabilities offered in both database security controls to build powerful protection layers that prevent access to sensitive objects unless explicitly authorized. For more details, refer to *Section 13.3.3.1 Using SQL Firewall in an Oracle Database Vault Environment* in the [Database Security Guide](#).

What is the performance impact of SQL Firewall?

Internal performance tests using an OLTP mixed workload show that SQL Firewall introduces a minimal CPU overhead of 1% per transaction when capturing SQL statements at a rate of 140K SQLs/sec from 60 concurrent sessions. During allow-list enforcement, even in extreme scenarios where all incoming SQL statements trigger SQL Firewall violations, the CPU overhead does not cross beyond 1.5% per transaction.

If you audit SQL firewall violations, consider the additional performance overhead introduced. For performance considerations of unified audit, refer to [this technical report](#).

Does SQL Firewall inspect encrypted SQL traffic?

Yes. SQL Firewall can inspect the traffic to and from an Oracle Database even when SQL traffic is encrypted – that includes both native network encryption (NNE) and transport-layer security (TLS). The allow-list enforcement happens as a penultimate step before SQL execution and after the SQL compilation phase, where privilege checks, semantic checks, query re-writes, and decryption of SQL statements occur.

Can SQL Firewall be used for DBAs and other ad-hoc users?

Yes. SQL Firewall policies work at a database account level, whether of an application account or a direct database ad-hoc user, such as a reporting user or a database administrator. This flexibility allows you to gradually build up the protection level of the database, starting from either the database administrators or the application accounts.

SQL Firewall can log (and block) connections that do not come from trusted IP addresses, operating system usernames, or program names. This function is useful when you want to put some protection in place immediately for administrators or ad-hoc users. This feature ensures that direct access to your database comes exclusively from trusted endpoints.

SQL Firewall's trusted SQL capability is ideal for application accounts/users with a well-defined set of established allowed SQL queries to the database and a controlled set of database connection paths. In most cases, ad-hoc users granted access to the database generally operate outside these established baselines and tend to run SQL queries that can vary with each execution, meaning only the connection context control is appropriate. In other words, in most cases, only connection context policies should be configured for ad-hoc users or administrators; SQL statement policies are usually not appropriate.

I have Oracle Audit Vault and Database Firewall (AVDF). How do I decide which one to use: SQL Firewall (or) Database Firewall?

Database Firewall, a component of AVDF, is a network-based SQL-aware firewall that can monitor Oracle and non-Oracle databases' SQL traffic. For non-Oracle databases, AVDF is the obvious choice. For Oracle databases, there are multiple factors that you might want to consider when deciding between the two options:

- Performance Impact:** The database server has no additional CPU overhead with a network-based Database Firewall. All work is done off-server on the Database Firewall machine. For most deployment modes, the Database Firewall introduces zero performance overhead on the database server. In proxy mode, the Database Firewall introduces minor additional network latency, with the degree of latency primarily controlled by the position of the Database Firewall on the network.

SQL Firewall introduces no network latency but does induce minimal CPU overhead, typically not exceeding 2%.
- Deployment Options:** Database Firewall requires a dedicated virtual machine or hardware for installation and has different deployment options: Proxy, Host Monitor, and Out-of-Band. The proxy is the only in-line configuration that supports blocking use cases. However, using the Database Firewall in proxy mode requires changes in the client-side connection to route traffic through it, and network firewall changes are required to block connections to the database from circumventing the Database Firewall.

SQL Firewall is part of the database and requires no client-side changes. It is the best solution for deployments where blocking is the predominant use case or changing client-side configurations is impractical or undesirable.
- Scope of inspection:** Database Firewall can only monitor SQL traffic coming over the network. It lacks visibility into SQL traffic over local BEQ connections and is unaware of internal jobs/ stored procedure executions. For comprehensive monitoring, database auditing is also used with network monitoring.

On the other hand, SQL Firewall runs inside the database and sees ALL traffic, regardless of the point of origin.
- Visibility into run-time execution context:** Enforcement hinges on the firewall's ability to understand SQL with its full run-time execution context, which is required for analyzing SQL queries. Network-based SQL firewalls lack this visibility.

SQL Firewall is built into the database kernel. It utilizes the same run-time execution context that the database builds while running any SQL statement, including steps to resolve synonyms, object names, or dynamically generated names, and current user context. The use of synonyms or dynamic SQL cannot fool SQL Firewall.
- Processing of encrypted traffic:** Handling encrypted SQL traffic requires additional processing in the Database Firewall but is seamless and straightforward in the SQL Firewall. For encrypted traffic, SQL Firewall processing comes after the database has already handed decryption, whether through TLS or native network encryption. Hence, network encryption is completely transparent to the SQL Firewall.
- Policy flexibility:** Database Firewall applies a single policy to a protected database – all connections through the firewall are examined using the same policy, making handling a mixture of ad-hoc and application connections challenging. SQL Firewall applies one policy per database account, meaning you can have a different policy for each protected account within a database. This makes it easier to protect databases with mixed user types or workloads.

SQL Firewall is included with the AVDF license. Consider leveraging the capabilities of both security controls to build powerful protection layers to control access to sensitive objects.

I have SQL Firewall. Why do I need database auditing?

Database auditing produces a record of database activities and helps demonstrate compliance with regulatory requirements. Auditing lets you focus on certain actions of specific users or critical activity in the database. It enables you to configure selective and precise audit policies tracking activities of interest. The most typical use cases of auditing privileged users or critical database operations cause negligible performance impact. In cases where organizations want to audit application usage, exercise appropriate discretion to mitigate any potential performance impact by fine-tuning audit policies.

On the other hand, SQL Firewall is primarily intended for application usage, typically processing extremely high volumes of data. SQL Firewall relies on the [memoptimized rowstore - fast ingest](#) feature introduced in Oracle Database 19c to minimize the potential performance impact of application workloads. You have the flexibility to audit SQL Firewall violations to ensure the abnormal access pattern is recorded as an audit event to demonstrate compliance.

ACCELERATE COMPLIANCE

How does SQL Firewall help accelerate compliance?

SQL Firewall is designed to help address security requirements for ensuring only authorized access required by various regulations, such as Sarbanes-Oxley, PCI, HIPAA, RBI Guidelines, and privacy laws, such as EU GDPR. SQL Firewall provides strong protective controls inside the database, controlling how application data can be accessed by application service accounts/users in an authorized manner.

SQL Firewall helps accelerate compliance with regulatory requirements on preserving the integrity of sensitive data and ensuring that they are accessed in an authorized manner, such as the following:

- US HIPAA's technical safeguards to safeguard protected health information
- PCI DSS's focus to maintain secure access to credit card information
- EU GDPR's requirements related to the processing of individuals' personal data
- RBI's circular on Digital Payment Security Controls (Feb 2021) to protect digital payment products/services

CONFIGURATION

How do I administer SQL Firewall?

You can manage SQL Firewall in multiple ways.

- Oracle Data Safe is your answer if you're looking for UI-based management or to manage multiple SQL Firewalls centrally. Suppose you're already a Data Safe customer using it for user and security assessment, activity auditing, sensitive data discovery, and data masking. In that case, you can use the same Data Safe cloud service to manage the SQL firewall and preserve your investment.

The Data Safe console lets you manage the SQL Firewall. Administrators can use the console to collect SQL activities of application accounts, monitor the collection progress, create SQL Firewall policies with allowlist rules (allowed contexts, allowed SQL statements) from the captured SQL activities, and enable SQL Firewall policies. When a firewall policy is enabled, Data Safe automatically collects the firewall violation logs from the database and stores them in Data Safe. Data Safe extends the configuration to a fleet of Oracle Database targets. You can use Data Safe REST APIs, software developer kits (SDKs), CLI, and Terraform for further automation and integration.

- If you wish to manage SQL Firewall within an individual Oracle Database instance and you are proficient with PL/SQL, use the PL/SQL APIs in the DBMS_SQL_FIREWALL package.

You can practice with both Data Safe and PL/SQL management in the [SQL Firewall LiveLabs workshop](#).

Can I enable SQL Firewall in the CDB?

Yes. SQL Firewall is container-specific. The administration of SQL Firewall, including policy creation with allow-lists, will be effective only in a container. You can enable it on the root container CDB\$ROOT and respective PDBs.

Can I enable SQL Firewall in the PDB?

Yes. SQL Firewall is container-specific. Enable and configure SQL Firewall in each PDB. For every database user you wish to protect in the PDB, let the SQL Firewall learn the normal SQL traffic, create a SQL Firewall policy with allow-list baselines, and enforce it to observe violations against the allowed baselines.

How does SQL Firewall know which statements are authorized?

The SQL Firewall administrator identifies the application service account/ database user whose access needs to be protected. In a controlled environment, the administrator trains the SQL Firewall to learn the normal application workload SQL statements and trusted database connection paths for the database user. The administrator can monitor if the SQL Firewall has learned the intended SQL statements and connection path attributes like client IP addresses, OS usernames, and OS program names. When there are no newer SQL statements or connection attributes in the inflow learning stream, the administrator stops learning to create an SQL Firewall policy for the database user. SQL Firewall policy constitutes the allow-list baseline of allowed and approved SQL statements and allowed database connection paths for the database user.

Can I modify the allow-lists once generated and enforced?

Yes. Following application updates, there might be newer SQLs you need to append to the allow-list or delete some from the existing allow-list. You have two options in Data Safe to modify the existing allow-lists:

Option1:

1. Rerun the SQL Collection after disabling the enforced SQL Firewall policy
2. Update the SQL Firewall policy to append the newer SQL statements and connection paths to the allow-list

Option2 (online):

1. Append the SQL statement(s) from the violation log to the enforced SQL Firewall policy while it is enabled
2. Or delete specific SQL statement(s) from the allow-list
3. Update the session context attributes in the allow-list

The changes will be effective immediately, even in existing current sessions. The entries in the allow lists are version-managed. You may consider using the PLSQL interfaces in SYS.DBMS_SQL_FIREWALL package to also perform this operation.

Can I reuse the allow list for a user across different databases?

Yes, you can [export the SQL Firewall configuration](#), including the allow lists for a database user, and [import](#) them for the same database user in a different database. This gives immense flexibility in defining the allow lists in a controlled environment for a database user account and internally testing them before rolling out the configuration in production databases.

If you want to export/import the SQL Firewall configurations for all users, leverage Oracle Data Pump's [expdp](#) and [impdp](#) commands.

Can I reuse the allow list for a different user on the same database?

No, the SQL Firewall policy with allow lists is unique to a specific database user and cannot be reused for a different user in the same database. You train the SQL Firewall to learn what normal SQL statements and database connections are per database

user you wish to protect. The normal expected SQL statements and database connections for the database user constitute the allow-list baseline for that user and is represented as a SQL Firewall policy for the specific database user.

Is it possible to have multiple policies enforced at the same time for the same user?

No, you cannot have more than one SQL Firewall policy enforced for the same user in a specific database instance. However, you can modify the allow-lists of a SQL Firewall policy to incorporate any changes if required.

How can SQL Firewall help prevent zero-day SQL Injection attacks?

You can use SQL Firewall to control which SQL statements the database can execute. Once the SQL Firewall policy with the allow list is enforced for the user, any unauthorized SQL and database connections that violate the baseline are logged. You can let the violated SQL statement/ database connection proceed further for execution in the database (or) block it from executing within the database.

Blocking mode is a very effective preventive strategy to avoid database attack patterns such as SQL Injection attacks. In contrast, the non-blocking/ observation mode is efficient for the real-time detection of potential database attacks.

What happens to applications whose statements are blocked by the SQL Firewall? Does the connection terminate?

Blocking for SQL statements following a mismatch between the SQL statement and the allow list does not terminate the session. Instead, it raises an *ORA-47605: SQL Firewall violation* error. This prevents anomalous database access without disrupting client connections.

Blocking for context violations will terminate client connections following a mismatch of session contexts against the allowed contexts. In this case, the session is not allowed to proceed to the database.

What are SQL Firewall violations, and how can I monitor them?

Once the SQL Firewall policy is enforced for the database user, checks are enforced on the allow-lists when the user connects to the database and issues SQL statements. SQL Firewall raises and logs violations for every unmatched scenario of database connection or SQL command execution against the entries in the enabled allow-lists of the SQL Firewall policy. The security administrator can monitor the SQL Firewall violation log to detect the presence of these anomalies. Data Safe can collect and store violation logs for analysis and reporting. You might want to audit SQL Firewall violations (especially the blocked ones); their occurrence potentially indicates abnormal database access attempts, including SQL Injection and credential theft/abuse.

Can SQL Firewall track violations on only untrusted database connection paths?

Yes. SQL Firewall enforces checks on the allow-lists in the policy when the user connects to the database and issues SQL statements. You can let the SQL Firewall know if you want to enforce checks on allowed session contexts, allowed SQL statements, or both. Suppose you enforce checks only on allowed session contexts. In that case, *context violation* is triggered if the incoming database connection paths do not match the entries in the allowed session contexts. Session context attributes include client IP addresses, OS program names, and OS usernames. This helps ensure that access to your databases comes exclusively from trusted endpoints defined in the allow-lists.

How do I extend beyond what I initially captured for allowed IP addresses?

The allowed context lists of client IP address, OS program name, and OS username in a SQL Firewall policy let you add additional ones or modify existing ones. SQL Firewall policies also support adding wildcards in the respective group (client IP address, OS program name, and OS username). You can refine them later when trusted database connection paths are well established.

How does SQL Firewall identify unique SQL statements for its allowed list?

SQL Firewall relies on SQL signatures to identify unique SQL statements. The uniqueness of an SQL statement is solely based on its syntactical structure and the database objects accessed. Bind variables and literals are not considered.

Every SQL statement is captured once per session, even if executed multiple times in the same session. Collected statements are normalized to remove redundant white spaces and comments. Hints and literals are replaced. The normalized SQL statement and fully qualified names of database objects accessed are hashed to generate the SQL signature.

When would you typically want to set TOP_LEVEL to false?

The TOP_LEVEL flag, if set to true, ensures that SQL Firewall will capture only user-initiated SQL statements for its allow list and enforce it for user-initiated SQL statements only. Setting it to false will capture SQL statements issued from within PL/SQL units invoked by the user in addition to user-initiated ones and enforce them for both levels.

The primary use case for setting TOP_LEVEL to false is to detect scenarios where PL/SQL units themselves might be vulnerable to SQL injection attacks.

Are SQL Firewall violations audited?

Yes, SQL Firewall violations can be audited if the policy is configured to do so. Oracle mandatorily audits all SQL Firewall administrative procedure executions. Optionally, you can audit SQL Firewall violations. The audited SQL Firewall violation event represents abnormal database access, which helps demonstrate compliance with regulatory requirements. When you enable SQL Firewall in blocking mode, it is a recommended best practice to audit SQL Firewall violations.

Would I get false positives in SQL Firewall violations?

No, unlike signature-based solutions, there are no false positives since something is either in the allowed list or it is not.

When would I need to update the allow list?

In an ideal scenario where SQL Firewall allow-lists capture all the expected SQL statements and trusted database connection paths, violations indicate potential database attacks such as compromised account access or SQL Injection attacks. But if these allow-lists are incomplete or there are newer authorized SQL following application updates, there is a possibility of seeing a surge in violations. Ensure the allow-lists are updated with newer authorized SQL and connection paths for the best results.

Do I need to purge SQL Firewall violation logs?

Yes, but if you are using Data Safe, this can be easily automated. SQL Firewall generates and stores the violations in a log table. Violations are expected to be small on a well-trained system. Data Safe allows you to manage the purge of violation logs once they are collected in Data Safe. When enabled, violation logs are automatically purged in the database if they are older than seven days and have already been collected in Data Safe. Alternatively, you can manage the purge on-demand with a PL/SQL interface or schedule it using the DBMS_SCHEDULER job.

Can I use SQL Firewall for ADMINISTRATIVE users like SYSDBA?

No. SQL Firewall is designed for application accounts/users. You cannot create SQL capture for administrative accounts such as SYS, SYSBACKUP, SYSDG, SYSKM, SYSRAC, AUDSYS, DVSYS, and LBACSYS. The ORA error you will likely encounter is: *ORA-47621: The SQL Firewall operation is not allowed on user SYS.*

How can I configure strict enforcement of allowed SQLs in the primary while relaxing the enforcement in Active Data Guard (ADG) standby?

Currently, there is no way to have different levels of enforcement in primary and standby databases in ADG for SQL Firewall for the same user. However, the SQL firewall policy is configurable and enforceable at the database user level. So, the SQL Firewall protection can differ if a different database user is used for operations at ADG standby databases.

Can I configure SQL Firewall protection for a particular schema?

No, SQL Firewall protection is configurable at the level of the database user. However, when you train the SQL Firewall to learn what normal SQL statements are for the database you wish to protect, you can ensure that all the relevant authorized SQL statements for the application schema are collected to create the allow-list of the SQL Firewall policy. You can view the allowed SQL statements of the policy, along with database objects being accessed and generate reports for offline validation with the application team if required.

Does SQL Firewall work with Real Application Security (RAS)?

Sort of. SQL Firewall will capture XS\$NULL's activities for RAS if the user's capture is enabled. However, it won't distinguish between RAS user identities. XS\$NULL is a database schema used by RAS to represent RAS users.

Can SQL Firewall be enforced on enterprise users?

Sort of. SQL Firewall will capture global user activities if capture is enabled for enterprise user identities such as CMU-AD, OCI IAM, OID, or Entra ID users. However, it won't distinguish between different enterprise user identities.

LICENSE

Is SQL Firewall available in Oracle Database Standard Edition?

No. SQL Firewall is available only for Oracle Database Enterprise Edition (version 23ai and later) and for these Database offerings:

- Oracle Base Database Service Enterprise Edition - High Performance (**BaseDB EE-HP**)
- Oracle Base Database Service Enterprise Edition - Extreme Performance (**BaseDB EE-EP**)
- Oracle Exadata Database Service on Dedicated Infrastructure (**ExaDB**)
- Oracle Exadata Database Service on Cloud@Customer (**ExaDB**)

SQL Firewall is not available for these Database offerings:

- Oracle Base Database Service Standard Edition (**BaseDB SE**)
- Oracle Base Database Service Enterprise Edition (**BaseDB EE**)

Oracle Database Vault is included in most DBaaS offerings on OCI. Hence, SQL Firewall is also available in most DBaaS offerings such as Autonomous Database, Oracle Database Service for Azure, and Oracle Database@Google Cloud.

Is SQL Firewall available in Oracle Database Free Offering?

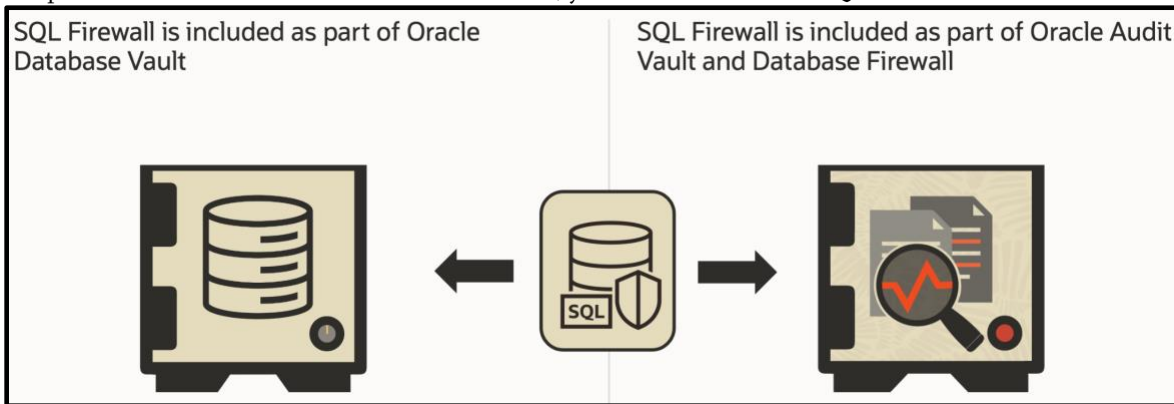
Yes, SQL Firewall is available in Oracle Database Free Offering. For details, refer to [Table 1-11 Security](#) in Database Licensing Manual.

How is SQL Firewall licensed?

SQL Firewall must be licensed for use. There are two paths to its license:

- SQL Firewall is included with Oracle Database Vault. Database Vault is an extra cost option.
 - Refer to [Table 1-15](#) in the Database Licensing Manual for details.
- SQL Firewall is included with Oracle Audit Vault and Database Firewall (AVDF). AVDF is a separate product and requires a license.
 - Refer [AVDF20 Licensing Manual](#) for details.

If you have purchased a license for Database Vault or AVDF, you are licensed to use SQL Firewall as shown here:



Is SQL Firewall licensed to use on DBaaS offerings in Oracle Cloud?

Oracle Database Vault is included in most DBaaS offerings on OCI. Hence, SQL Firewall is also available. The DBaaS offerings that do not include Database Vault (and therefore do not include SQL Firewall) are Oracle Base Database -Standard and Enterprise Edition.

MORE INFORMATION

Where can I get hands-on experience using SQL Firewall?

Oracle LiveLabs provides a platform to try the most common key features quickly. The SQL Firewall LiveLabs workshop is available at <https://apexapps.oracle.com/pls/apex/r/dbpm/livelabs/view-workshop?wid=3875>. You can experience SQL Firewall from the Data Safe console, including real-world practical use cases of how SQL Firewall can help detect and block access with stolen credentials and potential SQL Injection attempts.

Where can I see product demo videos of SQL Firewall?

A SQL Firewall demo video is accessible at <https://www.youtube.com/watch?v=81N23MDhYXU>.

Are there demo scripts available for customers to try hands-on in their environment?

Yes. Apart from LiveLabs, which allows you to try it in a sandbox environment or your own OCI tenancy with a real-world sample application workload, you can try the SQL Firewall configuration using the demo scripts in the GitHub location [here](#). Please remember that SQL Firewall is only available in your Oracle Database versions 23ai and later.

Where can I post my queries on this feature?


The [Developer Community forum](#) provides a platform for quick answers to your product questions from Oracle community experts.


Where is the documentation reference?


You can find more details in our official documentation [here](#).

Connect with us

Call **+1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

 blogs.oracle.com

 facebook.com/oracle

 twitter.com/oracle

Copyright © 2024, Oracle and/or its affiliates. This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.