Red Hat Data Grid 8.2

Red Hat Data Grid 8.2 Release Notes

Get release information for Data Grid 8.2
Red Hat Data Grid 8.2 Red Hat Data Grid 8.2 Release Notes

Get release information for Data Grid 8.2
Abstract

Find out about features and enhancements in Data Grid 8.2, learn about current known issues and resolved issues, review the configurations that Red Hat supports, and more.
Table of Contents

RED HAT DATA GRID ................................................................. 4
DATA GRID DOCUMENTATION .................................................... 5
DATA GRID DOWNLOADS ............................................................. 6
MAKING OPEN SOURCE MORE INCLUSIVE ..................................... 7

CHAPTER 1. DATA GRID RELEASE INFORMATION .............................. 8
1.1. WHAT’S NEW IN DATA GRID 8.2 GA ........................................ 8
  1.1.1. Data Grid Server ............................................................ 8
       Default authorization ....................................................... 8
       Credential stores for added security ................................... 9
       Audit logging ............................................................... 9
       Endpoint IP filtering ..................................................... 9
       Truststore for client cert validation ................................... 9
       Security realm identity caching ....................................... 10
       Implicit connectors for single-port endpoints ..................... 10
       Metrics endpoint aligned with previous versions ................. 10
  1.1.2. Data Grid Console ........................................................ 10
       User experience improvements ....................................... 11
       Role-based access control ............................................ 11
       Counter management ................................................... 11
  1.1.3. Data Grid Command Line Interface ................................... 11
       Backing up and restoring Data Grid clusters ...................... 11
       Performance test tool .................................................. 11
       Assigning user roles and permissions ............................... 11
       Cross-site replication operations .................................. 11
       Credential keystore management .................................... 12
       Native CLI ................................................................. 12
  1.1.4. Cross-site replication .................................................... 12
       Automatic state transfer operations ................................. 12
       SPI for custom conflict resolution .................................. 12
       Ability to verify cross-site views from CLI and REST .......... 13
  1.1.5. Hot Rod clients ............................................................ 13
       Improved near cache performance ................................... 13
       New Hot Rod client configuration properties .................... 14
  1.1.6. Query API ................................................................. 14
  1.1.7. REST API ................................................................. 14
       Streaming keys and entries ............................................ 14
       Working with the access control list cache ....................... 15
       Retrieving query and index statistics ............................. 15
       Cache configuration .................................................... 15
  1.1.8. Data Grid marshalling .................................................. 15
       ProtoStream ............................................................... 16
       Deserialization allow list ............................................. 16
       Creating SerializationContextInitializer implementations .... 16
       @ProtoAdaptor for marshalling of external classes ............ 17
       Kyro and Protostuff marshellers deprecated ..................... 17
  1.1.9. Data Grid configuration ................................................ 17
       Authorization: user roles and permissions ....................... 17
       Data Grid cache configuration fragments ......................... 18
### JGroups INSERT BEFORE attribute
- JGroups default stacks
- Cache health
- JDBC string-based cache stores

#### 1.1.10. Spring applications

#### 1.2. SUPPORTED JAVA VERSIONS IN DATA GRID 8.2
- Embedded caches
- Remote caches
- Java 8 deprecation

### CHAPTER 2. KNOWN AND FIXED ISSUES

#### 2.1. KNOWN ISSUES FOR DATA GRID
- 2.1.1. Open Issues for Data Grid 8.2
- Data Grid 8.2

#### 2.2. FIXED IN DATA GRID 8.2

### CHAPTER 3. TECHNOLOGY PREVIEWS

#### 3.1. TECHNOLOGY PREVIEW FEATURES
RED HAT DATA GRID

Data Grid is a high-performance, distributed in-memory data store.

**Schemaless data structure**
- Flexibility to store different objects as key-value pairs.

**Grid-based data storage**
- Designed to distribute and replicate data across clusters.

**Elastic scaling**
- Dynamically adjust the number of nodes to meet demand without service disruption.

**Data interoperability**
- Store, retrieve, and query data in the grid from different endpoints.
DATA GRID DOCUMENTATION

Documentation for Data Grid is available on the Red Hat customer portal.

- Data Grid 8.2 Documentation
- Data Grid 8.2 Component Details
- Supported Configurations for Data Grid 8.2
- Data Grid 8 Feature Support
- Data Grid Deprecated Features and Functionality
DATA GRID DOWNLOADS

Access the Data Grid Software Downloads on the Red Hat customer portal.

NOTE

You must have a Red Hat account to access and download Data Grid software.
Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright’s message.
CHAPTER 1. DATA GRID RELEASE INFORMATION

Learn about new features and get the latest Data Grid release information.

1.1. WHAT’S NEW IN DATA GRID 8.2 GA

Data Grid 8.2 improves usability, increases performance, and enhances security. Find out what’s new.

IMPORTANT

Red Hat advises customers to wait until Data Grid 8.2.1 becomes available before you upgrade to Data Grid 8.2 from 8.1.x.

This notice relates to JDG-4678 and applies to deployments where caches are configured with persistence using Single File cache stores. See the Known issues section for more information.

1.1.1. Data Grid Server

Data Grid Server provides a flexible, durable, and highly scalable data store for the Java Virtual Machine (JVM).

Default authorization

Data Grid Server configuration now enables authorization to restrict user access based on a set of default roles and permissions.

For example, users need to be assigned the admin role for permission to perform administrative operations on Data Grid clusters. When users without sufficient permission attempt to manipulate server resources, the following message is logged:

The user is not allowed to access the server resource: ISPN000287: Unauthorized access: subject 'Subject with principal(s): [myusername]' lacks 'ADMIN' permission

The following shows the default Data Grid Server configuration with authorization enabled:

```
<cache-container name="default" statistics="true">
  <transport cluster="${infinispan.cluster.name:cluster}"
    stack="${infinispan.cluster.stack:tcp}"
    node-name="${infinispan.node.name:}"/>
  <security>
    <authorization/>
  </security>
</cache-container>
```

Enables authorization for server administration and management. You can remove the authorization element to allow unrestricted access.

NOTE

This configuration does not affect cache configuration. You must separately enable authorization for your caches.
Creating and Modifying Users
User Roles and Permissions
Configuring User Authorization

Credential stores for added security
To protect sensitive text strings, such as passwords, you can now add them to a credential keystore instead of directly in Data Grid Server configuration.

Use the credentials command with the Data Grid CLI to set things up.

Storing Credentials in Keystores

Audit logging
Audit logs let you track changes to your Data Grid clusters so you know when changes occur and which users make them.

Use the org.infinispan.AUDIT logging category to enable audit logging and configure how configuration events and administrative operations are recorded.

Audit Logs

Endpoint IP filtering
You can now configure IP address filtering rules for Data Grid Server endpoints to accept or reject connections, as in the following example:

```
<endpoints socket-binding="default" security-realm="default">
  <ip-filter>
    <accept from="192.168.0.0/16"/> 1
    <accept from="10.0.0.0/8"/> 2
    <reject from="/0"/> 3
  </ip-filter>
  ...
</endpoints>
```

1. Accepts connections from clients with IP addresses in the `192.168.0.0/16` block.
2. Accepts connections from clients with IP addresses in the `10.0.0.0/8` block.
3. Rejects connections from clients with any other IP address.

Use the server connector ipfilter command with the Data Grid CLI to inspect and modify IP filter rules.

Endpoint IP Filtering

Truststore for client cert validation
Data Grid Server now lets you add client trust stores to the server-identities configuration if you want to enforce mutual SSL/TLS certificate validation between servers and clients.

```
<security-realm name="default">
  <server-identities>
    <ssl>
      <keystore path="server.pfx"
               keystore-password="password" alias="server"/>
    </ssl>
  </server-identities>
</security-realm>
```
Specifies a trust store that Data Grid Server uses to verify client identities.

Requires the trust store to contain public certificates for all clients. If you do not include the `truststore-realm` element, the trust store needs only a certificate chain.

- Trust Store Realms

Security realm identity caching
Data Grid Server now caches identities for security realms to improve performance.

You can configure the identity cache for security realms with the `cache-max-size` and `cache-lifespan` attributes, as in the following example that shows the default values:

```
<security-realm name="cache-max-size="256" 1
    cache-lifespan="-1"> 2
...
</security-realm>
```

1 Sets the maximum size for the identity cache.
2 Sets the lifespan of entries in the identity cache. By default, entries do not expire.

- Server configuration schema

Implicit connectors for single-port endpoints
You no longer need to define Hot Rod and REST connectors in Data Grid Server configuration when using a single port for endpoints.

For example, as of Data Grid 8.2, the following `endpoints` configuration implicitly uses default Hot Rod and REST connectors:

```
<endpoints socket-binding="default" security-realm="default"/>
```

The following `endpoints` configuration, by contrast, explicitly includes Hot Rod and REST connectors:

```
<endpoints socket-binding="default" security-realm="default">
    <hotrod-connector name="hotrod"/>
    <rest-connector name="rest"/>
</endpoints>
```

Metrics endpoint aligned with previous versions
For compatibility between versions, Data Grid Server now exports all metrics that were available from Data Grid 7.3 and later.

1.1.2. Data Grid Console
Data Grid Console provides a graphical user interface for monitoring and maintaining remote Data Grid clusters.

**User experience improvements**
With Data Grid 8.2, the console offers several user experience and usability enhancements.

- You are now prompted to create users from the welcome page if no users already exist.
- All text strings have been reviewed and edited to align with PatternFly UX writing guidelines.
- Contextual help, in the form or labels and descriptions, have been updated for clarity.

**Role-based access control**
Data Grid Console applies security authorization configuration and restricts user access based on their assigned roles and permissions.

**Counter management**
Data Grid Console improves management of counters in this release, letting you delete and filter counters.

### 1.1.3. Data Grid Command Line Interface

The Data Grid Command Line Interface (CLI) lets you perform administrative operations on remote Data Grid clusters.

**Backing up and restoring Data Grid clusters**
The CLI provides a `backup` command that can create archives of Data Grid resources that include cached entries, cache configurations, Protobuf schemas, and server scripts. You can then restore Data Grid clusters from backup archives after a restart or migration.

- Backing Up and Restoring Data Grid Clusters
- Backup Command Reference

**Performance test tool**
The `benchmark` command lets you run performance tests against caches with the CLI.

- Benchmark Command Reference

**Assigning user roles and permissions**
The CLI extends the `user` command with a `roles` subcommand that lets you view, grant, and deny user roles. You can now dynamically update user role assignments to control authorization settings and restrict access to Data Grid clusters and caches.

- User Command Reference

**Cross-site replication operations**
As of this release, the CLI lets you perform additional cross-site replication operations with the `site` command.

- `site name` Returns the name of the local site.
- `site view` Returns a list of names for all sites that can back up to each other.
- `site state-transfer-mode` Configures cross-site state transfer to occur manually or automatically.

- Site Command Reference
Credential keystore management
The CLI includes a `credentials` command that manages credential keystores for Data Grid Server.

- Credentials Command Reference

Native CLI
Data Grid 8.2 adds a native CLI that you can run on Linux, macOS, or Windows and use as an `oc` client plugin.

1. Download the native CLI from the Red Hat customer portal at Data Grid Software Downloads.
2. Open the README included with the distribution for installation instructions and example usage.

**NOTE**
The native CLI is currently available as a Technology Preview.

1.1.4. Cross-site replication
Cross-site replication lets you back up Data Grid clusters across multiple geographic regions.

Automatic state transfer operations
When issues occur and backup locations go offline, you must manually perform cross-site state transfer operations with the CLI or via JMX or REST.

However, when using the asynchronous backup strategy, Data Grid can now automatically perform cross-site state transfer operations after resolving conflicts. When it detects that a backup location has come back online, and the network connection is stable, Data Grid initiates bi-directional state transfer between backup locations. For example, Data Grid simultaneously transfers state from LON to NYC and NYC to LON.

**NOTE**
Automatic state transfer is possible only with the asynchronous backup strategy.

- Automatic State Transfer

SPI for custom conflict resolution
Data Grid provides an SPI that lets you customize conflict resolution for asynchronous Active/Active backup configurations.

The `XSiteMergePolicy` enum provides the following options for conflict resolution:

**DEFAULT**
Uses the default algorithm for handling conflicts from concurrent writes.

**PREFER_NON_NULL**
If a write/remove conflict occurs, this algorithm keeps the write operation and discards the remove operation. The default algorithm applies to all other conflicts.

**PREFER_NULL**
If a write/remove conflict occurs, this algorithm keeps the remove operation and discards the write operation. The default algorithm applies to all other conflicts.

**ALWAYS_REMOVE**
Removes conflicting entries from both sites.

You can specify the conflict resolution policy, including a custom implementation, with the `merge-policy` attribute, for example:

```xml
<distributed-cache name="eu-customers">
    <backups merge-policy="org.mycompany.MyCustomXSiteEntryMergePolicy">
        <backup site="LON" strategy="ASYNC"/>
    </backups>
</distributed-cache>
```

- Configuring Cross-Site Conflict Resolution
- `org.infinispan.xsite.spi.XSiteEntryMergePolicy`

**Ability to verify cross-site views from CLI and REST**

You can now verify cross-site view through the CLI or REST API.

For the CLI, invoke the `site view` command to retrieve a list of names for all sites that backup to each other.

From the REST API, invoke the following `GET` request:

```
GET /rest/v2/cache-managers/{cacheManagerName}
```

Data Grid responds with the list of backup locations in JSON format, as follows:

```
"sites_view": [ 
    "LON",
    "NYC"
]
```

- Site Command Reference
- Getting Basic Cache Manager Information (REST API)

### 1.1.5. Hot Rod clients

Hot Rod is a custom binary TCP protocol that provides high-performance data access to client applications in different programming languages.

**IMPORTANT**

If you are using Java 8 with your Hot Rod client, you must upgrade to at least Java 8u252 to avoid fatal `SSLHandshakeException` errors with Data Grid 8.2. See the [Known issues](#) for more information.

**Improved near cache performance**

Data Grid Server now includes bloom filters that optimize performance for write operations by reducing the total number of invalidation messages.

Enable bloom filters for near caches with the `nearCacheUseBloomFilter()` method.

- Near Caches
New Hot Rod client configuration properties
As of Data Grid 8.2, the Hot Rod client configuration API provides the following configuration properties:

- `infinispan.client.hotrod.transport_factory` specifies a transport factory to use. The default is `org.infinispan.client.hotrod.impl.transport.netty.DefaultTransportFactory`.

- `infinispan.client.hotrod.cache.<cache_name>.marshaller` specifies a marshaller to use on a per-cache basis.

- `infinispan.client.hotrod.ssl_ciphers` lists ciphers, separated with spaces and in order of preference, that are used during the SSL handshake to negotiate a cryptographic algorithm for key encryption.

- `infinispan.client.hotrod.ssl_provider` specifies the security provider to use when creating an SSL engine and defaults to OpenSSL.

- `infinispan.client.hotrod.cache.<cache_name>.transaction.transaction_manager_lookup` specifies a `TransactionManagerLookup` to use on a per-cache basis.

**NOTE**
Some properties, such as `infinispan.client.hotrod.trust_store_path`, are now deprecated. Check the Deprecations and removals article in the Red Hat Knowledgebase for more details.

- Hot Rod client configuration API

### 1.1.6. Query API

The Data Grid Query API lets you index caches and search values using relational or full-text queries with the Ickle query language.

Data Grid 8.2 brings you a significantly improved query implementation that is now based on Hibernate Search 6, which brings support for Apache Lucene 8 indexing capabilities. This release offers the following query enhancements:

- Faster indexing.

- Statistics for indexed, non-indexed, and hybrid queries.

- Strong-typed indexing configuration that replaces string key/value properties.

For complete documentation on indexing and querying, see [Querying Values in Caches](#).

**NOTE**
Review migration details to learn how to adapt your query configuration for Data Grid 8.2. See the [Data Grid Migration Guide](#) for more information.

### 1.1.7. REST API

The Data Grid REST API lets you interact with remote clusters and caches over HTTP.

**Streaming keys and entries**
The Data Grid REST API now lets you retrieve all keys or entries in caches in JSON format. Invoke GET requests as follows:

**Streaming keys**
```
GET /rest/v2/caches/{cacheName}?action=keys
```

**Streaming entries**
```
GET /rest/v2/caches/{cacheName}?action=entries
```
- Retrieving All Keys from Caches
- Retrieving All Entries from Caches

**Working with the access control list cache**
Data Grid 8.2 includes an access control list (ACL) cache that stores user role mappings. You can interact with the ACL cache via the REST API.

**Viewing user ACL information**
```
GET /rest/v2/security/user/acl
```

**Flushing the ACL cache**
```
POST /rest/v2/security/cache?action=flush
```
- Retrieving the ACL of a user

**Retrieving query and index statistics**
Obtain information about queries and indexes in caches with GET requests.
```
GET /v2/caches/{cacheName}/search/stats
```
- Retrieving Query and Index Statistics

**Cache configuration**
The Data Grid REST API now provides improved responses when retrieving cache configuration to make it easier to compare and validate cache configuration on remote clusters with cache configuration in your local project.

**NOTE**
If cache configurations contain deprecated attributes, Data Grid automatically converts them for compatibility with the current schema.

To ensure that your cache configurations are easy to compare, applications should always use the most recent schema.

**1.1.8. Data Grid marshalling**
Data Grid includes a ProtoStream API in addition to other marshaller implementations that allow you to transmit custom Java objects across the network and to persistent storage.
ProtoStream

Data Grid 8.2 upgrades the ProtoStream API to 4.4.1.Final.

NOTE

Changes to the the ProtoStream API in Data Grid 8.2 affect upgrade from previous Data Grid 8 versions.

For more information, see Data Grid 8 upgrade notes

Deserialization allow list

In keeping with Red Hat’s commitment to using inclusive language the term "white list" has been changed to "allow list" for configuring serialization of your Java classes.

Data Grid 8.1

```xml
<cache-container>
  <serialization>
    <allow-list>
      <class>org.infinispan.test.data.Person</class>
      <regex>org.infinispan.test.data.*</regex>
    </allow-list>
  </serialization>
</cache-container>
```

Data Grid 8.2

```xml
<cache-container>
  <serialization>
    <allow-list>
      <class>org.infinispan.test.data.Person</class>
      <regex>org.infinispan.test.data.*</regex>
    </allow-list>
  </serialization>
</cache-container>
```

Creating SerializationContextInitializer implementations

Data Grid now provides an @AutoProtoSchemaBuilder annotation that generates an implementation of a class or interface that extends SerializationContextInitializer. This provides a more efficient and reliable mechanism to create Protobuf schema and marshellers when storing custom Java objects in Data Grid caches.

NOTE

In previous Data Grid versions, you used the MessageMarshaller API and ProtoSchemaBuilder annotation to create Protobuf schema. You should migrate to the @AutoProtoSchemaBuilder annotation and start using that instead.

- ProtoStream annotations
- Creating serialization context initializers
- Migrating applications to the AutoProtoSchemaBuilder annotation
@ProtoAdaptor for marshalling of external classes
Data Grid adds support for the `@ProtoAdaptor` annotation that you can add to adaptor classes for any external, third-party Java object classes.

- Creating ProtoStream adapter classes

Kyro and Protostuff marshellers deprecated
The Kyro and Protostuff marshellers are now deprecated. See the Deprecations and removals article in the Red Hat Knowledgebase for more details.

1.1.9. Data Grid configuration
Data Grid offers schema-based configuration options for caches as well as for customizing underlying mechanisms such as security and cluster transport.

Authorization: user roles and permissions
Data Grid 8.2 has improved role-based access control (RBAC) functionality that secures access to Data Grid installations as well as caches.

To enable authorization for Cache Manager access, add the `authorization` element to the `cache-container` as in the following example:

```
<cache-container name="default" statistics="true">
  <security>
    <authorization/>
  </security>
</cache-container>
```

To enable authorization for caches, add the `authorization` element as follows:

```
<distributed-cache name="myCache" mode="SYNC">
  <security>
    <authorization/>
  </security>
</distributed-cache>
```

Cluster role mapper
Data Grid 8.2 introduces the `ClusterRoleMapper`, which is the default mechanism that Data Grid uses to associate security principals to authorization roles.

This role mapper uses a persistent replicated cache to dynamically store principal-to-role mappings for the default roles and permissions.

Table 1.1. Default user roles

<table>
<thead>
<tr>
<th>Role</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>admin</td>
<td>ALL</td>
<td>Superuser with all permissions including control of the Cache Manager lifecycle.</td>
</tr>
<tr>
<td>Role</td>
<td>Permissions</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>--------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>deployer</td>
<td>ALL_READ, ALL_WRITE, LISTEN, EXEC, MONITOR, CREATE</td>
<td>Can create and delete Data Grid resources in addition to application permissions.</td>
</tr>
<tr>
<td>application</td>
<td>ALL_READ, ALL_WRITE, LISTEN, EXEC, MONITOR</td>
<td>Has read and write access to Data Grid resources in addition to observer permissions. Can also listen to events and execute server tasks and scripts.</td>
</tr>
<tr>
<td>observer</td>
<td>ALL_READ, MONITOR</td>
<td>Has read access to Data Grid resources in addition to monitor permissions.</td>
</tr>
<tr>
<td>monitor</td>
<td>MONITOR</td>
<td>Can view statistics via JMX and the metrics endpoint.</td>
</tr>
</tbody>
</table>

### New permissions

The **CREATE** permission lets users create and remove container resources such as caches, counters, schemas, and scripts.

![NOTE]

> **NOTE**

The **CREATE** permission replaces the **schema_manager** and **script_manager** roles that users require to add and remove schemas and scripts to Data Grid Server.

The **MONITOR** permission allows access to JMX statistics and the **metrics** endpoint.

For more information about authorization, see the following in the Security Guide:

- Role Mappers
- Access Control List (ACL) Cache
- User Roles and Permissions
- Permissions

### Data Grid cache configuration fragments

As of Data Grid 8.2, you no longer need to include `infinispan` and `cache-container` elements in your cache configuration.

To create caches you need to provide only the `*-cache` elements.

For example, to create a distributed cache that uses synchronous mode you can use the following configuration:

```xml
<distributed-cache name="myCache" mode="SYNC" />
```
To create a replicated cache that uses Protobuf encoding for entries you can use the following configuration:

```xml
<replicated-cache name="books">
  <encoding media-type="application/x-protostream"/>
</replicated-cache>
```

**JGroups INSERT BEFORE attribute**

You can now use the `INSERT_BEFORE` value to customize JGroups cluster transport with the `stack.combine` inheritance attribute, for example:

```xml
<ASYM_ENCRYPT asym_keylength="2048"
  asym_algorithm="RSA"
  change_key_on_coord_leave = "false"
  change_key_on_leave = "false"
  use_external_key_exchange = "true"
  stack.combine="INSERT_BEFORE"
  stack.position="pbcast.NAKACK2"/>
```

1 Inserts the `ASYM_ENCRYPT` protocol before `pbcast.NAKACK2` in a JGroups stack.

**NOTE**

The Data Grid 8.2 schema also includes the `INSERT_ABOVE` and `INSERT_BELOW` attributes.

- `INSERT_ABOVE` is the same as `INSERT_AFTER`. `INSERT_BELOW` is the same as `INSERT_BEFORE`.

**Inheritance Attributes**

**JGroups default stacks**

Data Grid 8.2 changes the configuration for re-transmission requests for the UNICAST3 and NAKACK2 protocols in the default JGroups stacks.

In some cases, such as when a long Garbage Collection (GC) pause takes place, nodes cannot process JGroups messages from other nodes in the cluster. When those nodes become available again, they request the sending nodes to re-transmit the JGroups messages using XMIT requests.

To improve performance and avoid issues with cluster transport issues from failed re-transmission requests, the following changes apply:

- The value of the `xmit_interval` property is increased from 100 milliseconds to 200 milliseconds.
- The `max_xmit_req_size` property now sets a maximum of 500 messages per re-transmission request, instead of a maximum of 8500 with UDP or 64000 with TCP.

**Cache health**

Data Grid includes a new **FAILED** status. As of this version, available health status for caches is as follows:
### Health Status

<table>
<thead>
<tr>
<th>Health Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTHY</td>
<td>Indicates a cache is operating as expected.</td>
</tr>
<tr>
<td>HEALTHY_REBALANCING</td>
<td>Indicates a cache is in the rebalancing state but otherwise operating as expected.</td>
</tr>
<tr>
<td>DEGRADED</td>
<td>Indicates a cache is not operating as expected and possibly requires troubleshooting.</td>
</tr>
<tr>
<td>FAILED</td>
<td>Added in 8.2 to indicate that a cache could not start with the supplied configuration.</td>
</tr>
</tbody>
</table>

**JDBC string-based cache stores**

JDBC string-based cache stores now creates a `_META` table in addition to the data table used to store cache entries. The `_META` table holds metadata that ensures any existing database content is compatible with the current Data Grid version and configuration.

### 1.10. Spring applications

Data Grid provides Spring Cache and Spring Session implementations.

As of Data Grid 8.2, your Spring applications can use the ProtoStream marshaller to encode and decode Java objects into Protocol Buffers (Protobuf) format.

For more information, see the Spring Boot Starter

### 1.2. SUPPORTED JAVA VERSIONS IN DATA GRID 8.2

Red Hat supports different Java versions, depending on how you install Data Grid.

**Embedded caches**

Red Hat supports Java 8 and Java 11 when using Data Grid for embedded caches in custom applications.

**Remote caches**

Red Hat supports Java 11 only for Data Grid Server installations. For Hot Rod Java clients, Red Hat supports Java 8 and Java 11.

**Java 8 deprecation**

As of Data Grid 8.2, support for Java 8 is deprecated and currently planned for removal in Data Grid 8.4.

Users with embedded caches in custom applications should plan to upgrade to Java 11 or to Java 17 when support becomes available.

Hot Rod Java clients running in applications that require Java 8 can continue using older versions of client libraries. Red Hat supports using older Hot Rod Java client versions in combination with the latest Data Grid Server version.

**Additional resources**

- [Supported Configurations for Data Grid 8.2](#)
- Data Grid Deprecated Features and Functionality
CHAPTER 2. KNOWN AND FIXED ISSUES

Learn about known issues in Data Grid and find out which issues are fixed.

2.1. KNOWN ISSUES FOR DATA GRID

For issues that affect Data Grid clusters running on Red Hat OpenShift, you should refer to the Data Grid Operator 8.2 release notes.

Upgrading to Data Grid 8.2 from 8.1 fails and data is corrupted if caches persist data to Single File cache stores

Issue: JDG-4678

Description: The first time you restart Data Grid after upgrading from 8.1 to 8.2, Data Grid fails with the following log messages:


Workaround: If you persist data to a Single File cache store in Data Grid 8.1.x, you should wait until Data Grid 8.2.1 becomes available before you upgrade your deployment.

Session externalization from Red Hat JBoss Web Server to Data Grid 8.2 is available with 7.3.8 or 8.1.1 versions of the Tomcat session client

Issue: JDG-4599

Description: Data Grid 8.2 does not yet include the Tomcat session client, which will be available shortly after 8.2 GA.

Workaround: Use Data Grid Server 8.2 in combination with a Data Grid 7.3.8 or 8.1.1 version of the Tomcat session client with the following configuration:

```xml
<Manager className="org.wildfly.clustering.tomcat.hotrod.HotRodManager"
    configurationName="default"
persistenceStrategy="${persistenceStrategy}"
    server_list="127.0.0.1:11222"
    protocol_version="2.9"
    auth_realm="default"
sasl_mechanism="DIGEST-MD5"
    auth_server_name="infinispan"
    auth_username="admin"
    auth_password="changeme"/>
```

Hot Rod clients using Java 8 need upgrade to avoid SSLHandshakeException

Issue: JDG-4279
**Description:** Using JDK 8 with Hot Rod clients results in the following fatal exception if your Java version is not 8u252 at a minimum:

```java
SSLHandshakeException: Remote host closed connection during handshake at sun.security.ssl.SSLSocketImpl
```

**Workaround:** Ensure you are using Java 8u252 at a minimum. This version includes required security features for Application-Layer Protocol Negotiation (ALPN).

### Maximum idle expiration with cross-site replication does not behave as expected

**Issue:** JDG-4605

**Description:** When Data Grid attempts to coordinate maximum idle expiration across clusters using with cross-site replication, it does not correctly handle the metadata for `maxIdle` values. As a result, entries do not expire on both clusters according to the maximum idle time.

**Workaround:** There is no workaround for this issue.

### Hot Rod NodeJS clients cannot establish connections to Data Grid Server with authentication mechanisms other than PLAIN

**Issue:** JDG-3868

**Description:** Hot Rod NodeJS clients can connect to remote Data Grid clusters that use the **PLAIN** authentication mechanism only. If you configure Hot Rod endpoints with any other authentication mechanism, NodeJS clients cannot successfully connect to Data Grid Server.

**Workaround:** Configure Data Grid Server to use the **PLAIN** authentication mechanism with the Hot Rod endpoint, as in the following example:

```xml
<hotrod-connector>
  <authentication>
    <sasl mechanisms="PLAIN"
      server-name="infinispan"
      qop="auth"/>
  </authentication>
</hotrod-connector>
```

See [Configuring endpoint authentication mechanisms](#) for more information.

### Data Grid Conflict Resolution Performance

**Issue:** JDG-3636

**Description:** In some test cases, Data Grid partition handling functionality took longer than expected to perform conflict resolution.

**Workaround:** There is no workaround for this issue.

### Data Grid Does Not Passivate JWS Sessions Correctly

**Issue:** JDG-2796

**Description:** When externalizing sessions from JBoss Web Server (JWS), sessions are not passivated correctly if using the **FINE** persistence strategy.

**Workaround:** There is no workaround for this issue.
2.1.1. Open Issues for Data Grid 8.2

You can find the complete list of issues for Data Grid with the following Jira query:

Data Grid 8.2
List of All Open Issues for Data Grid 8.2

Query JQL

```
project = JDG AND affectedVersion in ("RHDG 8.2 CR1", "RHDG 8.2 CR2", "RHDG 8.2 GA") AND status not in (Closed, Verified, "Ready for QA", "QA In Progress", Resolved) AND type =Bug
```

2.2. FIXED IN DATA GRID 8.2

List of All Issues Resolved in Data Grid 8.2

Features and enhancements, documentation improvements, bug fixes.

Query JQL

```
project = JDG AND fixVersion in ("RHDG 8.2 CD", "RHDG 8.2 CR1", "RHDG 8.2 CR2", "RHDG 8.2 GA")
```

Notable Fixes

This release includes the following notable fixes:

- **JDG-4315** IllegalArgumentException when performing rolling upgrades with transactional caches.
- **JDG-3972** Inconsistent query behavior when indexed caches contain non-indexed Protobuf entities.
- **JDG-4520** DB2TableManager looks into all schemas during table existence check.
- **JDG-4425** JGroups retransmission requests are too frequent and too large.
- **JDG-4420** ClassNotFoundException happens when putting a custom object with jboss-marshalling on EAP embedded module.
- **JDG-4387** Simple cache with statistics enabled results in NullPointerException in EvictionManagerImpl.
- **JDG-4375** How to disable TLS 1.1 and enable only TLS 1.2.
- **JDG-4370** Make AWS dependencies optional in EAP modules.
- **JDG-4351** Slow performance can happen and can face a transaction timeout due to the cost of invoking Arrays.fill(keys, null) against a huge array on IdentityIntMap#clear().
- **JDG-4344** Experiencing a memory leak on the HotRod client when we put entries with a big value size.
- **JDG-4339** Conflict resolution fails in transactional cache.
- **JDG-4315** IllegalArgumentException when doing Rolling Upgrades on transactional caches.
• **JDG-4281** Infinispan operator CR not accepting changes in logging level.
• **JDG-4152** Infinispan CR not idempotent when removing "expose" configuration.
CHAPTER 3. TECHNOLOGY PREVIEWS

Data Grid releases offer technology preview features. Find out more about Red Hat support for these capabilities.

3.1. TECHNOLOGY PREVIEW FEATURES

Technology preview features or capabilities are not supported with Red Hat production service-level agreements (SLAs) and might not be functionally complete.

Red Hat does not recommend using technology preview features or capabilities for production. These features provide early access to upcoming product features, which enables you to test functionality and provide feedback during the development process.

For more information, see Red Hat Technology Preview Features Support Scope.