Red Hat JBoss Data Grid 6.2

6.2.0 Release Notes

New features, known issues, and resolved issues for Red Hat JBoss Data Grid 6.2
Edition 1

Last Updated: 2017-11-21

Red Hat JBoss Data Grid 6.2 6.2.0 Release Notes

New features, known issues, and resolved issues for Red Hat JBoss Data Grid 6.2
Edition 1

Gemma Sheldon
Red Hat Engineering Content Services
gsheldon@redhat.com

Abstract

The Red Hat JBoss Data Grid 6.2 Release Notes list and provide descriptions for a series of bugzilla bugs. The bugs highlight either issues that are known problems for the relevant release or bugs that have now been resolved.

Table of Contents

CHAPTER 1. INTRODUCTION TO RED HAT JBOSS DATA GRID 6.2 3
1.1. ABOUT RED HAT JBOSS DATA GRID 3
1.2. OVERVIEW 3
1.3. NEW FEATURES AND ENHANCEMENTS 3
1.3.1. Querying in Library Mode 3
1.3.2. Remote Querying Over Hot Rod (Technology Preview) 3
1.3.3. C++ Hot Rod Client 4
1.3.4. Compatibility Mode 4
1.3.5. New High Performance File Cache Store 4
1.3.6. LevelDB Cache Store 4
1.3.6.1. LevelDB Tested Configurations 5
1.3.7. Authentication and Encryption Over Hot Rod 5
1.3.8. Scalability of Cross-Datacenter Replication 5
1.3.9. Rolling Upgrades for REST Clusters 5
1.3.10. Command Line Interface 5
1.3.11. Support on Azul JVM 5

CHAPTER 2. SUPPORTED CONFIGURATIONS 7

CHAPTER 3. COMPONENT VERSIONS 8

CHAPTER 4. KNOWN ISSUES 9

CHAPTER 5. RESOLVED ISSUES 12

APPENDIX A. REVISION HISTORY 14

CHAPTER 1. INTRODUCTION TO RED HAT JBOSS DATA GRID 6.2

Welcome to the Red Hat JBoss Data Grid 6.2. As you become familiar with the newest version of JBoss Data Grid, these Release Notes provide you with information about new features, as well as known and resolved issues. Use this document in conjunction with the entire JBoss Data Grid 6.2 documentation suite, available at the Red Hat Customer Service Portal's JBoss Data Grid page.

1.1. ABOUT RED HAT JBOSS DATA GRID

Red Hat's JBoss Data Grid is an open source, distributed, in-memory key/value data store built from the Infinispan open source software project. Whether deployed in client/server mode or embedded in a Java Virtual Machine, it is built to be elastic, high performance, highly available and to scale linearly.

JBoss Data Grid is accessible for both Java and Non-Java clients. Using JBoss Data Grid, data is distributed and replicated across a manageable cluster of nodes, optionally written to disk and easily accessible using the REST, Memcached and Hot Rod protocols, or directly in process through a traditional Java Map API.

1.2. OVERVIEW

This document contains information about the new features, as well as the known and resolved issues in Red Hat JBoss Data Grid version 6.2.0. Customers are requested to read this documentation prior to installing this version.

Customers can provide feedback or log bugs through their Support account.

1.3. NEW FEATURES AND ENHANCEMENTS

1.3.1. Querying in Library Mode

Red Hat JBoss Data Grid 6.1 featured a technology preview of a Java querying API in Library mode to search for data in the grid using values instead of keys. In JBoss Data Grid 6.2 this API has been improved and is now fully supported. Library Mode Querying provides features such as:

- Keyword, Range, Fuzzy, Wildcard, and Phrase queries
- Combining queries
- Sorting, filtering, and pagination of query results

NOTE

Using JBoss Data Grid as a Lucene Directory is supported only in the context of this Querying functionality.

For more information about this feature, see the JBoss Data Grid *Infinispan Query Guide*.

1.3.2. Remote Querying Over Hot Rod (Technology Preview)

In addition to the Java Querying API in Library mode, JBoss Data Grid 6.2 introduces a Technology Preview for Querying the grid from remote Java Hot Rod clients using a new domain-specific language

(DSL). For more information about this feature, see the Red Hat JBoss Data Grid *Infinispan Query Guide* and the Remote Query quick-start example included in the distribution.

1.3.3. C++ Hot Rod Client

A new C++ Hot Rod client is available in JBoss Data Grid 6.2. This enables C++ applications to read (get) and write (put) data to a remote cluster using the Hot Rod protocol.

The C++ Hot Rod client is supported on following platforms:

- Red Hat Enterprise Linux 5, 64-bit
- Red Hat Enterprise Linux 6, 64-bit

IMPORTANT

The C++ Hot Rod client is available as a Technology Preview on Windows with Visual Studio 2010 (32-bit and 64-bit). Remote Querying is unavailable in the C++ Hot Rod client in this release.

1.3.4. Compatibility Mode

JBoss Data Grid 6.2 features the new Compatibility Mode, which enables data to be read (get) or written (put) regardless of the deployment mode or server endpoint. Some examples of this feature are:

- An entry written by a Java Hot Rod client can be read by a C++ Hot Rod client.
- An entry written by Hot Rod client can be read by a REST client.
- An entry written in Library mode can be read by a Hot Rod client.

1.3.5. New High Performance File Cache Store

JBoss Data Grid 6.2 features a new File cache store which provides high throughput in reads and writes to disk. The new cache store is available on both Library and Client-server modes.

This cache store implementation keeps in-memory an index of the keys for the persisted entries. For use cases in which storing these keys in-memory is not a suitable option, a new LevelDB cache store is also available.

1.3.6. LevelDB Cache Store

JBoss Data Grid 6.2 features a new LevelDB cache store which provides high throughput in reads and writes to disk. This cache store is based on Google's C++ LevelDB implementation with a JNI connector, and is available in both Library and Client-server modes.

The LevelDB cache store is the preferred option when a large number of entries are persisted locally on a node.

The LevelDB library and JNI connector are not part of the JBoss Data Grid 6.2 distribution.

To use the tested version, add the file `leveldbjni-all-1.7.jar` from https://github.com/fusesource/leveldbjni to your JBoss Data Grid deployment.

1.3.6.1. LevelDB Tested Configurations

LevelDB is tested using JBoss Data Grid's Library and Remote Client-Server modes on:

- Red Hat Enterprise Linux 5 (32-bit and 64-bit) with the following JDKs:
  - IBM JDK 1.6 and 1.7
  - OpenJDK 1.6
  - Oracle JDK 1.6 and 1.7
- Red Hat Enterprise Linux 6 (32-bit and 64-bit) with the following JDKs:
  - IBM JDK 1.6 and 1.7
  - OpenJDK 1.6 and 1.7
  - Oracle JDK 1.6 and 1.7

Red Hat supports using LevelDB via this JNI connector as a Cache Store with Red Hat JBoss Data Grid 6.2, and will assist in diagnosing an issue with use of LevelDB with JBoss Data Grid. However, if the issue is determined to be with LevelDB or the JNI connector, Red Hat will be limited in how it can resolve the problem - Red Hat does not distribute, maintain, or patch LevelDB or the JNI connector with JBoss Data Grid 6.2.

1.3.7. Authentication and Encryption Over Hot Rod

Hot Rod server authentication and certificate-based client authentication via the Transport Layer Security (SSL) protocol is available in JBoss Data Grid 6.2. Once the server has been authenticated, the communication with the Hot Rod client is encrypted.

1.3.8. Scalability of Cross-Datacenter Replication

In JBoss Data Grid 6.2, Cross-Datacenter Replication can use multiple site masters (up to the number of nodes in the cluster) to replicate the data from one cluster to the other. This results in a significant improvement in scalability over that of JBoss Data Grid 6.1, which relied on a single site master.

1.3.9. Rolling Upgrades for REST Clusters

This new feature enables you to upgrade the REST cluster from JBoss Data Grid 6.2 to a future version without any downtime.

NOTE

This feature does not enable a rolling upgrade of REST cluster from an earlier version of JBoss Data Grid to JBoss Data Grid 6.2.

1.3.10. Command Line Interface

The command line interface (CLI), which was Technology Preview in Red Hat JBoss Data Grid 6.1, is now fully supported in JBoss Data Grid 6.2 for both Remote Client-Server and Library modes.

1.3.11. Support on Azul JVM

From version 6.2 onwards, JBoss Data Grid is supported on the Azul Zing JVM 5.6.1.

CHAPTER 2. SUPPORTED CONFIGURATIONS

For supported hardware and software configurations, see the Red Hat JBoss Data Grid Supported Configurations reference on the Customer Portal at https://access.redhat.com/site/articles/115883.

CHAPTER 3. COMPONENT VERSIONS

The full list of component versions used in Red Hat JBoss Data Grid is available at the Customer Portal at https://access.redhat.com/site/articles/488833.

CHAPTER 4. KNOWN ISSUES

The following issues are known to exist in Red Hat JBoss Data Grid 6.2 and will be fixed in a subsequent release.

BZ#818863 **- Tests for UNORDERED and LRU eviction strategies fail on IBM JDK**

The LinkedHashMap implementation in IBM's JDK behaves erratically when extended (as is done by the eviction strategy code). This incorrect behavior is exposed by the Red Hat JBoss Data Grid Test Suite. It is recommended, if using eviction, to use LIRS eviction strategy, which is not affected by this issue. Only LRU eviction strategy is affected.

BZ#881080 **- Silence SuspectExceptions**

SuspectExceptions are raised when nodes are shutting down, which is normal and expected, since a suspect node is an unresponsive node. As a result, a SuspectException ERROR is written in the logs. There is currently no fix for this issue. The SuspectExceptions will not be logged in a future version of Red Hat JBoss Data Grid. The SuspectExceptions do not affect data integrity.

BZ#881791 **- Special characters in file path to JDG server are causing problems**

Using special characters in directory paths can cause problems when starting Red Hat JBoss Data Grid server. The JBoss Data Grid server either fails to start or a configuration file for logging capabilities cannot be loaded properly. Special characters include spaces, # (hash sign), ! (exclamation mark), % (percentage sign), $ (dollar sign). This bug causes the JBoss Data Grid server to fail to start properly.Avoiding the use of special characters will stop this problem occurring, allowing JBoss Data Grid server to start properly.

BZ#888429 **- JGroups: SuccessfulResponse not serializable exception on shutdown**

A race condition in the Red Hat JBoss Data Grid shutdown sequence causes an attempt to deserialize an incoming response after the externalizer table, which is used for deserialization of messages, has been stopped. Consequently, a SuccessfulResponse not serializable exception is thrown on shutdown. The issue does not cause data corruption and only happens during shutdown and does not affect availability.

BZ#903308 **- IllegalStateException on shutdown**

Occasionally in Red Hat JBoss Data Grid, while a coordinator node is being shut down, an asynchronous listener thread might attempt to retrieve the state of the cluster and fail with an IllegalStateException.As a result, the IllegalStateException is logged by the coordinator node while it is shutting down.

The IllegalStateException printed by the coordinator node during shutdown can be ignored. The issue does not affect data integrity in any way, and the node will safely shut down.

BZ#987520 **- Putting entries with memcached is ignoring the queue-flush-interval parameter**

The queue-flush-interval and queue-size attributes of a replicated cache are ignored when data is stored remotely from Memcached client. This is caused by the "distribution-based replication" algorithm in JBoss Data Grid: a key's primary owner might not be the local node, so a synchronous put operation is sent to the key owner, thus overriding the async queue.

Consequently, data is replicated immediately and the queue-flush-interval and queue-size attributes are not respected.

BZ#1012036 **- RELAY2 logs error when site unreachable**

When a site is unreachable, JGroups's RELAY2 logs error for each dropped message. Infinispan has configurable fail policies (ignore/warn/abort), and even with ignore policy the log is filled with errors.

_BZ#1013001_ - [ISPN-3375] Map/Reduce jobs with small value sizes fail because of timeouts putting intermediate keys into the cache

When a Map/Reduce task is executed against a cache with many small values, the task may experience TimeoutExceptions while trying to write the intermediate results to the cache.As a result, when these exceptions occur, the task fails.

There are several things the user can do to try and alleviate this issue:

1. Use larger values in the cache where the Map/Reduce task is executed.
2. Use a collator and/or a combiner on the Map/Reduce task if this is possible.
3. Configure the size of the intermediate chunks written to the cache using the state transfer chunkSize parameter.

_BZ#1043434_ - L1 entry added by ST when already invalidated

In distributed cache mode with L1 enabled and `onRehash=true`, when an entry is overwritten during state transfer, the overwrite may be ignored. This node will then report an outdated value of the entry, and after L1 timeout this entry will be completely removed.

As a workaround, use L1 with `onRehash=false`.

_BZ#1043853_ - Concurrent message headers modification causes that message is never sent

Sending a message through JGroups may fail under some race conditions with ArrayOutOfBoundException. The reliability mechanism will try to resend this message again, but all these subsequent attempts will fail with NullPointerException.

Therefore, the message will be never received on the receiver node and will be held in unacknowledged messages buffer on the sender node. If this message is ordered within JGroups, no more ordered messages from this node will be delivered on the receiver node. Some following messages may be buffered as well, eventually leading to OutOfMemoryException.

It is recommended to use UNICAST3 as this dramatically reduces the chance of this bug to happen (it was observed with UNICAST2).

_BZ#1047905_ - REPEATABLE_READ not working for AtomicMap and FineGrainedAtomicMap

When REPEATABLE_READ isolation mode is enabled, reading values from AtomicMap and FineGrainedAtomicMap is not consistent within the same transaction. If another thread changes the value during the transaction, the new value is reflected immediately in the current thread while the thread should still see the original value.

_BZ#1047908_ - WriteSkew not throwing an exception for AtomicMap in REPL and DIST mode

The writeSkewCheck flag does not work correctly for AtomicMap. When JBoss Data Grid runs in REPEATABLE_READ isolation mode for transactions, the writeSkewCheck flag is enabled, and two concurrent transactions read/write data from the same AtomicMap. The WriteSkewException exception is never thrown when committing either of the transactions.

Consequently, the data stored in the AtomicMap might be changed by another transaction, and current transaction will not register this change during commit.

BZ#1050007 **- Fallback to non-preferred IP stacks if host can't be resolved with the preferred one**

By default, the C++ Hot Rod client resolves server host names using IPv4 addresses. If a server is listening on an IPv6 address, the client will not be able to connect to it.

Either specify the server using its IP instead of the name or set the HOTROD_IPSTACK environment variable to IPV4 or IPV6 to force resolution using the preferred stack. The client will connect to the specified server.

CHAPTER 5. RESOLVED ISSUES

The following issues have been resolved in Red Hat JBoss Data Grid 6.2.

BZ#854665 **- Coordinator tries to install new view after graceful shutdown**

In JBoss Data Grid, when gracefully stopping a coordinator node, the node itself would log an attempt to install a new clustering view which would fail. This is harmless, since the new coordinator would in fact perform the proper view installation.

BZ#967984 **- Support For Child Elements of <loader> Missing When Defining Custom Cache Loader**

Red Hat JBoss Data Grid server is not able to parse XML configuration for a custom cache loader if the configuration contains child elements, such as custom properties, inside <loader> element. As a result, JBoss Data Grid server fails to start.

The support for child elements of <loader> element has been implemented. Result: Custom cache loader inside JDG server allows for nested configuration tags and can successfully start with such a configuration.

BZ#989970 **- CLI upgrade command does not work**

When data is being migrated via CLI from a node with old version of JBoss Data Grid to a node with new version, an error occurs during synchronization. This happens when the upgrade command is called on the new node with parameter --synchronize=hotrod. This results in data not properly migrating from the old node to the new one.

BZ#996045 **- ClusterCacheLoader - clusteredGetCommand response value is null**

When an entry is placed in a clustered cache with a ClusterCacheLoader defined, the client gets NullPointerException. As a result, it is not possible to use ClusterCacheLoader reliably.

BZ#1002957 **- Hot Rod client doesn't retry operation on RemoteException**

There are certain types of remote-side exceptions that the Java Hot Rod client will not recover from by retrying, even though it should be possible. This also happens when the org.infinispan.remoting.RemoteException is thrown in the server. The HotRodClientException is thrown on the client side, the operation does not retry and the issue needs to be handled by the user code.

BZ#1006281 **- REST cache store (REST-roll-ups.xml) has problems with modules and finding classes**

The rolling upgrade procedure for the REST client migrates data from a previous version of JBoss Data Grid server ("source") to a new version ("target"). The rest-store configuration element is required for this procedure to succeed. However, it is not possible to successfully start the "target" node with rest-store element defined. The server fails to start with java.lang.ClassNotFoundException: org.apache.commons.codec.EncoderException

As a result, it is not possible to perform rolling upgrades with REST client from JBoss Data Grid 6.1.0.GA to JBoss Data Grid 6.2.0.Beta.

BZ#1011943 **- Read-only attribute of remote-store is ignored**

In Remote Client-Server mode, when a remote-store is defined with a read-only attribute set to true, this attribute is ignored. As a result, data may be stored in the remote store despite it being defined as read-only.

BZ#1013242 **- Service jboss.infinispan.default failed to start on IBM JDK 6, argument type mismatch**

JBoss Data Grid fails to instantiate GlobalComponentRegistry on IBM JDK 1.6. This process fails with IllegalArgumentException: Argument Type Mismatch. As a result, JBoss Data Grid does not work correctly on IBM JDK 1.6. Instead, use IBM JDK version 1.7 instead of version 1.6. JBoss Data Grid works correctly on IBM JDK 1.7.

BZ#1019348 **- InvalidateL1Command during State Transfer should not cancel writing the entry by State Transfer**

In a cluster with L1 enabled, during rebalancing after a node joins or leaves the cluster, an entry may be lost on a node.

BZ#1019742 **- Out of data read after write on node losing ownership**

During the cluster rebalance process, after a node joins or leaves the cluster, a read operation may retrieve data that has already been overwritten. The behavior was observed in transactional mode but may not be limited to it.

BZ#1021362 **- Inconsistent L1 in tx distributed cache**

With L1 enabled, a node may cache an already overwritten entry. Further reads on this node will return out-of-date value.

BZ#1021461 **- In non-tx caches, write operations may not be atomic during rebalance**

In a non-transactional cache, conditional operations executed during cluster rebalance after a node joins or leaves may not test and set the value atomically - two competing commands may both override the value.

BZ#1025258 **- When L1.onRehash is enabled, L1 invalidations should be sent to the previous owners**

With L1 enabled, after a cluster rebalance following a node joining or leaving, L1 may stay outdated on a node - reads from this node will not reflect further writes to the entry until the entry times out from the L1.

BZ#1026862 **- HotRod Rolling Upgrades ClassNotFoundException: org.infinispan.CacheException during recordKnownGlobalKeyset**

The rolling upgrade process for Hot Rod fails with IOException during call of recordKnownGlobalKeyset operation. Consequently, it is not possible to successfully go through rolling upgrade process for Hot Rod client from version JBoss Data Grid 6.1.0.GA to JBoss Data Grid 6.2.0.Beta.

BZ#1027311 **- Client claims that it has L1 intelligence**

The C++ Hot Rod client has a hard-coded level of intelligence (level 1). As a result, the client has only basic intelligence, which means it is not topology aware. Thus, when the client connects to a cluster, and servers join or leave the cluster, the client will not be aware of it.

As a workaround, when new nodes join the cluster or leave it while the C++ client is connected, ensure all clients are restarted to get the new view of the cluster. Another option is to make sure no servers leave or join the cluster during the session.

APPENDIX A. REVISION HISTORY

Revision 6.2-4.2	**Thu Jan 30 2014**	**Misha Husnain Ali**

Updated year.

Revision 6.2-4.1	**Tues Jan 14 2014**	**Gemma Sheldon**

Minor changes to What's New chapter and bug list updated for GA.

Revision 6.2-3.1	**Fri Jan 10 2014**	**Gemma Sheldon**

First draft for GA.

Revision 6.2-2	**Fri Dec 13 2013**	**Gemma Sheldon**

Removed note about DSL from Remote Querying section.

Added Oracle JDK1.6 as tested platform for RHEL 6 and 5 for Level DB.

Revision 6.2-1	**Wed Dec 04 2013**	**Misha Husnain Ali**

Incorporated feedback.

Revision 6.2-0	**Thu Nov 28 2013**	**Misha Husnain Ali**

Added introductory text.