Red Hat JBoss Data Grid 6.5

6.5.1 Release Notes

Known and resolved issues for Red Hat JBoss Data Grid 6.5.1

Last Updated: 2017-11-13

Red Hat JBoss Data Grid 6.5 6.5.1 Release Notes

Known and resolved issues for Red Hat JBoss Data Grid 6.5.1

Christian Huffman
Red Hat Engineering Content Services
chuffman@redhat.com

Legal Notice

Abstract

The Red Hat JBoss Data Grid 6.5.1 Release Notes list and provide descriptions for a series of bugzilla bugs. The bugs highlight issues that are known problems and resolved issues for the release.

Table of Contents

CHAPTER 1. INTRODUCTION TO RED HAT JBOSS DATA GRID 6.5.1

Welcome to Red Hat JBoss Data Grid 6.5.1. 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 documentation suite, available at the Red Hat Customer Service Portal's JBoss Data Grid documentation page.

Report a bug

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.

Report a bug

1.2. OVERVIEW

This document contains information about the known and resolved issues of Red Hat JBoss Data Grid version 6.5.1. Customers are requested to read this documentation prior to installing this version.

Report a bug

CHAPTER 2. SUPPORTED CONFIGURATIONS

2.1. 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.

Report a bug

CHAPTER 3. COMPONENT VERSIONS

3.1. 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.

Report a bug

CHAPTER 4. KNOWN AND RESOLVED ISSUES

4.1. KNOWN ISSUES

BZ-1204813 - JSR-107 Support for cacheResolverFactory annotation property

JCache annotations provides a way to define a custom `CacheResolverFactory`, used to produce `CacheResolver`; this class's purpose is to decide which cache is used for storing results of annotated methods; however, the support for specifying a `CacheResolver` is not provided yet.

Define a CDI ManagedCacheResolver which will be used instead.

BZ-1158839 - Clustered cache with FileStore (shared=false) is inconsistent after restarting one node if entries are deleted during restart

In Red Hat JBoss Data Grid, when a node restarts, it does not automatically purge entries from its local cache store. As a result, the Administrator starting the node must change the node configuration manually to set the cache store to be purged when the node is starting. If the configuration is not changed, the cache may be inconsistent (removed entries can appear to be present).

This is a known issue in JBoss Data Grid 6.5.1 and no workaround is currently available for this issue.

BZ-1114080 - HR client SASL MD5 against LDAP fails

In Red Hat JBoss Data Grid, the server does not support pass-through MD5 authentication against LDAP. As a result, the Hot Rod client is unable to authenticate to the JBoss Data Grid server via MD5 is the authentication is backed by the LDAP server.

This is a known issue in JBoss Data Grid 6.5.1 and a workaround is to use the PLAIN authentication over end-to-end SSL encryption.

BZ-1024373 - Default optimistic locking configuration leads to inconsistency

In Red Hat JBoss Data Grid, transactional caches are configured with optimistic locking by default. Concurrent `replace()` calls can return true under contention and transactions might unexpectedly commit.

Two concurrent commands, `replace(key, A, B)` and `replace(key, A, C)` may both overwrite the entry. The command which is finalized later wins, overwriting an unexpected value with new value.

This is a known issue in JBoss Data Grid 6.5.1. As a workaround, enable write skew check and the ***REPEATABLE_READ*** isolation level. This results in concurrent replace operations working as expected.

BZ-1200822 - JSR-107 Support for clustered caches in HotRod implementation

When creating a new cache (which is not defined in server configuration file) in HotRod implementation of JSR-107, the cache is created as `local` only in one of the servers. This behavior requires class `org.jboss.as.controller.client.ModelControllerClient` to be present on the classpath.

As a workaround use a clustered cache defined in the server configuration file. This still requires `cacheManager.createCache(cacheName, configuration)` to be invoked before accessing the cache for the first time.

BZ-1223290 - JPA Cache Store not working properly on Weblogic

A JPA Cache Store deployed to WebLogic servers throws a **`NullPointerException`** after the following error message:

```
Entity manager factory name (org.infinispan.persistence.jpa) is already
registered
```

This is a known issue in JBoss Data Grid 6.5.1, and no workaround exists at this time.

BZ-1259914 - RemoteCacheManager.getCache may ignore the forceReturnValue flag

The **`getCache`** operation on **`RemoteCacheManager`** may ignore the ***`forceReturnValue`*** flag in the **`manager.getCache("foo", true)`** call. This behavior results in a cache with the ***`forceReturnValue`*** flag set to false instead of the expected value.

This is a known issue in JBoss Data Grid 6.5.1, and no workaround exists at this time.

Report a bug

4.2. RESOLVED ISSUES

BZ-1244133 - NearCache: it is possible to read stale data with a put/remove followed by a get

When using a near cache in **LAZY** mode there was a possibility that a read operation following a write operation on the same key within the same thread would return outdated data. For example, the below calls from the same thread could result in **`v1`** being returned:

```
put(k, v1)
put(k, v2)
get(k)
```

This issue is resolved as of Red Hat JBoss Data Grid 6.5.1.

BZ-1243671 - Null is returned for a not expired entry in Hot Rod client

When using the HotRod client to update the lifespan of an expirable entry so that it will be immortal (i.e. performing **`put`** with ***`lifespan`*** set to -1), the entry was unexpectedly removed from the cache.

This issue is resolved as of Red Hat JBoss Data Grid 6.5.1.

BZ-1235140 - Async stores can lose updates

`AsyncCacheWriter` could drop updates if it was unable to write **`modificationQueueSize`** entries within the ***`shutdownTimeout`***, as older changes overwrote newer ones in the back-end store. This issue was primarily caused by a large queue size or a slow back-end store with a relatively small **`AsyncStore`** ***`shutdownTimeout`*** (25 seconds by default).

Now **`AsyncStore`** will write everything to the backing store before shutting down, and the ***`shutdownTimeout`*** has been removed.

This issue is resolved as of Red Hat JBoss Data Grid 6.5.1.

BZ-1235134 - AsyncCacheLoader.load() may return stale data

A race condition existed that allowed `AsyncCacheLoader.load()` / `State.get()` to return stale data. As `State.get()` scans the states of `AsyncCacheWriter` from head to tail, while the `AsyncStoreCoordinator` may move conflicting entries from the current state to the head state. If `State.get()` / `AsyncCacheLoader.load()` is preempted after checking the head state then it may not see modifications moved by the coordinator thread and load stale data instead.

This issue is resolved as of Red Hat JBoss Data Grid 6.5.1; the `AsyncStoreCoordinator` was redesigned so that conflicting keys are collected in a separate list which is processed after the `AsyncStoreProcessors` complete.

BZ-1234927 - HotRod size method doesn't check BULK_READ permission

When `size()` is performed using the HotRod client, a MapReduce task is invoked internally to determine the total number of entries present in the cache. Previously this operation did not require any permissions; however, a check has been added to ensure the role invoking `size()` has the BULK_READ permission.

This issue is resolved as of Red Hat JBoss Data Grid 6.5.1.

BZ-1229875 - The server erroneously prints 'Ignoring XML attribute isolation, please remove from configuration file'

The only isolation level supported by server configuration in the `<locking>` element of cache configuration was READ_COMMITED. Due to this, the server incorrectly printed 'Ignoring XML attribute isolation, please remove from configuration file' log message, even if isolation attribute was not present in the `<locking>` element.

This issue is resolved as of Red Hat JBoss Data Grid 6.5.1 as the server currently supports specifying other isolation levels, resulting in the log message no longer being displayed.

BZ-1229791 - Cluster listener state can be invalid due to exception

When a clustered listener is registered and it throws an exception which is not caught, the listener is put in an inconsistent state. This might result in further exceptions when processing the next events, even if those events should generate no error or exception.

This issue is resolved as of Red Hat JBoss Data Grid 6.5.1.

BZ-1228676 - Allocated buffer in key and value converter should be released

The Hot Rod decoder was not reporting all errors produced, resulting in a combination of errors not being logged and byte buffers being leaked as these errors accumulated. As these buffers grew they could potentially cause OutOfMemory errors.

This issue is resolved as of Red Hat JBoss Data Grid 6.5.1, as the Hot Rod decoder extends a different Netty decoder that handles all potential failures.

BZ-1228026 - Invalid message id under high load

The C++ HotRod client was occasionally logging the following exception when there were many concurrent `Put` and `Get` operations:

```
"Invalid message id. Expected $OLD_MSGID and received $NEW_MSGID"
```

This issue was caused by the message id counter not being thread safe.

This issue is resolved as of Red Hat JBoss Data Grid 6.5.1.

BZ-1255783 - Server should enable writeSkew for some configurations by default

The server configuration did not allow users to specify write skew check option. This behavior could lead to inconsistencies on failure scenarios when using conditional operations with optimistic locking caches.

This issue is resolved as of Red Hat JBoss Data Grid 6.5.1, where write skew check is automatically enabled when using a combination of optimistic locking, synchronous cache mode, and **REPEATABLE_READ** isolation level.

BZ-1255213 - Retry failed saying "Failed to connect" under high load

The HotRod C++ client's failover was failing sporadically under high load. This was caused by the `Transport` object throwing a `TransportException` during an operation, resulting in a `ConnectionPool::invalidateObject` which removed the `Transport` from the busy queue, destroyed the `Transport`, and then added a new `Transport` to the idle queue. When the new `Transport` is created, it tries to connect to the server socket which fails. This prevents retry attempts and client failover from happening.

This issue is resolved as of Red Hat JBoss Data Grid 6.5.1.

BZ-1254321 - ConcurrentTimeoutException if a HotRod client uses getAll(...) and the owners < numOfNodes

When compatibility mode was enabled in distributed mode and the number of key owners was less than the number of nodes in the cluster, the `getAll` operation on the HotRod client performed a local `getAll`, which sometimes resulted in not returning all entries.

This issue is resolved as of Red Hat JBoss Data Grid 6.5.1.

BZ-1252986 - putIfAbsentAsync not enforcing withFlags(Flag.FORCE_RETURN_VALUE)

The call to `putIfAbsentAsync` on a remote HotRod client and using `withFlags(Flag.FORCE_RETURN_VALUE)` did not work as expected. The previous value was not returned as expected unless the flag was also configured in HotRod client properties file.

This issue is resolved as of Red Hat JBoss Data Grid 6.5.1.

BZ-1223395 - Operation [Registering Deployed Cache Store service...] happens too late on slower machines

While using a deployed Custom Cache Store for a particular cache, there was a potential race condition in the JBoss Data Grid server. It was possible that the cache start-up would occur before the Custom Cache Store library registration was successfully finished, resulting in the cache being unable to find the requested resources during the start-up process and failing to start.

This issue is resolved as of Red Hat JBoss Data Grid 6.5.1.

Report a bug

APPENDIX A. REVISION HISTORY

Revision 6.5.1-0 **Mon 24 Aug 2015** **Christian Huffman**

Created document.