Latest Issues -

Managed Services Resolutions - Cloudera

SPARK

ERROR: "Constant pool for class org.apache.spark.sql.catalyst.expressions.GeneratedClass$SpecificUnsafeProjection has grown past JVM limit of 0xFFFF" when working with wide datasets in Spark jobs

Spark job fails due to JaninoRuntimeException when working with very wide datasets. Spark is affected by the Java constant pool limit of 65536 (0xFFFF). This can manifest in job failures when working with very wide datasets.

Symptom:
Spark job fails with the following error in log files:

org.codehaus.janino.JaninoRuntimeException: Constant pool for class org.apache.spark.sql.catalyst.expressions.GeneratedClass$SpecificUnsafeProjection has grown past JVM limit of 0xFFFF

The error pattern can be: **grows beyond 64 KB** or **grown past JVM limit of 0xFFFF**.

Applies to : Spark 2.2 and lower versions

RCA - This is a Known Issue (SPARK-18016).

Resolution:

To resolve this issue, upgrade to Spark 2.3. Spark 2.3 has the fix for this bug.

2.**Namenode Web UI shows differences between Configured Capacity, Capacity Used, and Live Nodes/Nodes in Service in the Summary and DFS Storage Types sections**

Symptom:

Configured Capacity:	8.32 PB
DFS Used:	5.75 PB (69.16%)
Non DFS Used:	12.1 TB
DFS Remaining:	2.44 PB (29.3%)

DFS Storage Types

Storage Type	Configured Capacity	Capacity Used	Capacity Remaining	Block Pool Used	Nodes In Service
DISK	8.42 PB	6.27 PB (74.5%)	2.02 PB (23.97%)	6.27 PB	294
ARCHIVE	611.63 TB	1.31 MB (0%)	611.63 TB (100%)	1.31 MB	17

Security is on.

Safemode is off.

66,020,070 files and directories, 54,032,254 blocks = 120,052,324 total filesystem object(s).

Heap Memory used 34.82 GB of 59.85 GB Heap Memory. Max Heap Memory is 59.85 GB.

Non Heap Memory used 122.6 MB of 124.84 MB Commited Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity:	8.28 PB
DFS Used:	5.75 PB (69.47%)
Non DFS Used:	11.37 TB
DFS Remaining:	2.4 PB (28.99%)
Block Pool Used:	5.75 PB (69.47%)
DataNodes usages% (Min/Median/Max/stdDev):	36.61% / 74.60% / 78.51% / 8.79%
Live Nodes	273 (Decommissioned: 0, In Maintenance: 0)
Dead Nodes	12 (Decommissioned: 12, In Maintenance: 0)
Decommissioning Nodes	0
Entering Maintenance Nodes	0
Total Datanode Volume Failures	0 (0 B)

Cause :
The issue can manifests itself in two different ways which are documented in HDFS-11896 and HDFS-9034.

1. When a DataNode is marked dead and then re-registers. This can be done by stopping a DataNode for at least 10.5 minutes and then restarting the datanode which can occur during a maintenance outage for a DataNode. A failed decommission and recommission of a DataNode could also cause the issue.
2. Failed disks or storage on DataNodes can also cause the issue.

Resolution:

A Workaround for scenario #1 above would be to perform a rolling restart of the NameNodes after all DataNodes are online and healthy.

A Workaround for scenario #2 above would be to replace all the failed disks on the DataNodes and perform a rolling restart of the NameNodes.

A permanent fix for scenario #1 and #2 would be to upgrade to CDH 6.0.0 and later which contains the fixes for both HDFS-11896 and HDFS-9034.

3.**"Error while compiling statement: FAILED: SemanticException Line 0:-1 Invalid table alias or column reference 'sq_1'" during Hive Query**

Symptoms:

Hive Query fails with error "Invalid table alias or column reference 'sq_1'"

To reproduce the issue,

1.Create two simple tables:

CREATE TABLE a (k1 int, k2, int);

CREATE TABLE b (k1 int, k2, int);

2.Run

SELECT * FROM a WHERE EXISTS (SELECT * FROM b WHERE a.k1=b.k1 and a.k2=b.k2);

3. The following error is displayed

Error: Error while compiling statement: FAILED: SemanticException Line 0:-1 Invalid table alias or column reference 'sq_1':

(possible column names are: _table_or_col b) k2) sq_corr_1)) (tok, (. (tok_table_or_col sq_1) sq_corr_1)) (state=42000,code=400000)

Cause :

This is a Known Issue (HIVE-14719).

Resolution:

To resolve this issue, do any of the following:

1.Enable CBO optimization. This is false in CDH6.x by default. SET hive.cbo.enable=true;

2.Rewrite the query using normal JOIN, as shown:

SELECT a.* FROM a JOIN b ON (a.k1=b.k1 AND a.k2=b.k2);

4.TSB 2019-336: Apache Spark local files left unencrypted

Certain operations in Spark leave local files unencrypted on disk, even when local file encryption is enabled with “spark.io.encryption.enabled”.

Symptoms:

Certain operations in Spark leave local files unencrypted on disk, even when local file encryption is enabled with “spark.io.encryption.enabled”.

This includes cached blocks that are fetched to disk (controlled by spark.maxRemoteBlockSizeFetchToMem) in the following cases:

- In SparkR when parallelize is used
- In Pyspark when broadcast and parallelize are used
- In Pyspark when python udfs is used

User affected: All users who run Spark on CDH and CDS in a multi-user environment.

Date/Time of detection: July 2018

Impact: Unencrypted data accessible.

Resolutions:

Upgrade (recommended)

Update to a version of CDH or CDS.

Workaround

Do not use of pyspark and the fetch-to-disk options.

5.Kudu Tablet server crashes with error message "Cannot receive messages from a leader in leader mode" in tserver log.

Symptom:

The Kudu Tablet Server crashed due to below error.

F1006 09:40:11.971015 7895 time_manager.cc:132] Check failed: mode_ == NON_LEADER (0 vs. 1) Cannot receive messages from a leader in leader mode.

The Tablet Server crash occurs due to KUDU-2274 (Race when tombstoned replica gets copied and becomes leader). The issue have been fixed in CDH 5.14.2 and later.

Resolution:

Restart the affected Tablet Server and upgrade to CDH 5.14.2 and later to prevent further occurrences of the issue.

6.Telemetry publisher (TP) is unable to push logs and error message "Caused by: javax.net.ssl.SSLPeerUnverifiedException: peer not authenticated" is seen in the log. As a result, Workload XM will not receive cluster jobs information.

Symptoms: Users are not able to see MR jobs in WXM. And users notice multiple alerts on Telemetry publisher (TP). Following is the error message from the log file:

2019-11-26 21:02:38,458 WARN com.cloudera.cdx.client.impl.bulk.store.LocalFileSystemRecordStore: Export failed

java.lang.RuntimeException: javax.net.ssl.SSLPeerUnverifiedException: peer not authenticated

at com.google.common.base.Throwables.propagate(Throwables.java:160)

at com.cloudera.cdx.client.impl.bulk.upload.DatabusUploader.export(DatabusUploader.java:111)

at com.cloudera.cdx.client.impl.bulk.store.LocalFileSystemRecordStore.export(LocalFileSystemRecordStore.java:173)

at com.cloudera.cdx.client.impl.bulk.CdxBulkExporter$1.run(CdxBulkExporter.java:103)

at java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:511)

at java.util.concurrent.FutureTask.runAndReset(FutureTask.java:308)

at java.util.concurrent.ScheduledThreadPoolExecutor$ScheduledFutureTask.access$301(ScheduledThreadPoolExecutor.java:180)

at java.util.concurrent.ScheduledThreadPoolExecutor$ScheduledFutureTask.run(ScheduledThreadPoolExecutor.java:294)

at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)

at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)

at java.lang.Thread.run(Thread.java:748)

Caused by: javax.net.ssl.SSLPeerUnverifiedException: peer not authenticated

at sun.security.ssl.SSLSessionImpl.getPeerCertificates(SSLSessionImpl.java:450)

at org.apache.http.conn.ssl.AbstractVerifier.verify(AbstractVerifier.java:126)

at org.apache.http.conn.ssl.SSLSocketFactory.connectSocket(SSLSocketFactory.java:437)

at org.apache.http.impl.conn.DefaultClientConnectionOperator.openConnection(DefaultClientConnectionOperator.java:180)

at org.apache.http.impl.conn.ManagedClientConnectionImpl.open(ManagedClientConnectionImpl.java:294)

at org.apache.http.impl.client.DefaultRequestDirector.tryConnect(DefaultRequestDirector.java:643)

at org.apache.http.impl.client.DefaultRequestDirector.execute(DefaultRequestDirector.java:479)

at org.apache.http.impl.client.AbstractHttpClient.execute(AbstractHttpClient.java:906)

at org.apache.http.impl.client.AbstractHttpClient.execute(AbstractHttpClient.java:805)

at org.apache.http.impl.client.AbstractHttpClient.execute(AbstractHttpClient.java:784)

at com.cloudera.cdx.client.impl.bulk.upload.DatabusUploader.putPayload(DatabusUploader.java:118)

at com.cloudera.cdx.client.impl.bulk.upload.DatabusUploader.export(DatabusUploader.java:108)

... 9 more

2019-11-26 21:02:38,552 WARN com.cloudera.cdx.client.impl.bulk.store.SimpleFileRecordStore: Export failed

Cause:

Telemetry publisher needs to push data over Internet and it uses TLS for connections . Following are the hosts to which it needs to connect with.

- dbusapi.us-west-1.sigma.altus.cloudera.com:443
- cloudera-dbus-prod.s3.amazonaws.com:443

Public certificates offered by above web URLs can be verified by the default cacerts trustStore included with the JVM. However, if a new trustStore is created, but certificates from cacerts aren't included, it could cause the above issues.

Workaround:

1.It's always good to confirm that network connectivity isn't an issue on the telemetry host. Following could be a good way of verifying.

$ curl https://dbusapi.us-west-1.sigma.altus.cloudera.com:443

$ curl https://cloudera-dbus-prod.s3.amazonaws.com:443

2.Add all certificates from JVMs cacert to the used trustStore using a command like following.

$ keytool -importkeystore -srckeystore cacerts -srcstorepass changeit -destkeystore <used_truststore> -deststorepass <truststore_password>

Alternate Way:

You could just import the root and intermediate certificates for the web URLs mentioned in "Cause" section. Please read this documentation for more details:

You could just import the root and intermediate certificates for the web URLs mentioned in "Cause" section. Please read this documentation for more details:

7.NullPointerException seen in DataNode log due to uninitialized DiskBalancer

Symptoms:

The following NullPointerException is seen in the DataNode log on startup.

019-09-23 01:42:17,654 ERROR org.apache.hadoop.jmx.JMXJsonServlet: getting attribute DiskBalancerStatus of Hadoop:service=DataNode,name=DataNodeInfo threw an exception

javax.management.RuntimeMBeanException: java.lang.NullPointerException

Cause :

https://issues.apache.org/jira/browse/HDFS-13721

Workaround :

This issue is harmless and does not affect anything with the DataNode operations and is observed only during startup. However, upgrading to CDH 6.0.1 and later will resolve the issue.

8.ZooKeeper JMX did not support TLS when managed by Cloudera Manager before version 6.1.0. This article explains details and how to fix the issue.

Symptoms:

The ZooKeeper service optionally exposes a JMX port used for reporting and metrics. By default, Cloudera Manager enables this port, but prior to Cloudera Manager 6.1.0, it did not support mutual

TLS authentication on this connection. While JMX has a password-based authentication mechanism that Cloudera Manager enables by default, weaknesses have been found in the authentication mechanism, and Oracle now advises JMX connections to enable mutual TLS authentication in addition to password-based authentication. A successful attack may leak data, cause denial of service, or even allow arbitrary code execution on the Java process that exposes a JMX port. Beginning in Cloudera Manager 6.1.0, it is possible to configure mutual TLS authentication on ZooKeeper's JMX port.

Workaround:

Upgrade Cloudera Manager versions to any of below versions which have fixes:

- 6.1.0 and higher
- 5.16.2 and higher

and enable TLS for the ZooKeeper JMX port by turning on the configuration settings "Enable TLS/SSL for ZooKeeper JMX" and "Enable TLS client authentication for JMX port" on the ZooKeeper service and configuring the appropriate TLS settings.

9. **ERROR: "Argument list too long" in Hadoop commands after setting HADOOP_CLASSPATH**

Adding directory to HADOOP_CLASSPATH, shell commands fail with error "Argument list too long".

Symptoms:

Sometimes, after setting HADOOP_CLASSPATH, Hadoop commands fail.

$ export HADOOP_CLASSPATH=`hadoop classpath`:/opt/cloudera/parcels/CDH/jars/*

$ beeline -d "com.cloudera.impala.jdbc41.Driver" -u "jdbc:impala://impala.exampe.com:21050;AuthMech=1;KrbRealm=EXAMPLE.COM;KrbHostFQDN=_HOST;KrbServiceName=impala;"

///opt/cloudera/parcels/CDH-6.1.x-1.cdh6.1.x.p0.1435386/bin/../lib/hadoop/libexec/hadoop-functions.sh: line 1770: /usr/java/jdk1.8.0_141-cloudera/bin/java:

Argument list too long

///opt/cloudera/parcels/CDH-6.1.x-1.cdh6.1.x.p0.1435386/bin/../lib/hadoop/libexec/hadoop-functions.sh: line 1770: /usr/java/jdk1.8.0_141-cloudera/bin/java: Success

Unable to determine Hadoop version information.

'hadoop version' returned:

///opt/cloudera/parcels/CDH-6.1.x-1.cdh6.1.x.p0.1435386/bin/../lib/hadoop/libexec/hadoop-functions.sh: line 1770: /usr/java/jdk1.8.0_141-cloudera/bin/java:

Argument list too long

///opt/cloudera/parcels/CDH-6.1.x-1.cdh6.1.x.p0.1435386/bin/../lib/hadoop/libexec/hadoop-functions.sh: line 1770: /usr/java/jdk1.8.0_141-cloudera/bin/java: Success

Cause :

This issue occurs when too many jar files are added to CLASSPATH.

From CDH 6, Hadoop scripts go through all jar files included in HADOOP_CLASSPATH and they are added to CLASSPATH one by one. If too many jar files are added to CLASSPATH, the CLASSPATH will be a very long string and it causes all commands launched from shell to fail with error Argument list too long.

Workaround : To resolve this issue, do not add unnecessary files in HADOOP_CLASSPATH. For example, only add the required jar files. You can place the required jar files in a separate directory and add the separate directory into HADOOP_CLASSPATH.

10. **Impala Service Fails to Start | Error: "Failed to find any Kerberos tgt" | Policy File Descrepency**

Resolving Impala start failures in a Kerberized cluster by configuring the "Maximum renewable life" property.

Symptoms:

In a Kerberos enabled cluster, Impala sometimes fails to start. We can see the following log, for example:

impalad.ERROR

```
 E0530 15:04:48.303849 20917 UserGroupInformation.java:1411]
PriviledgedActionException as:impala (auth:KERBEROS)
cause:java.io.IOException: javax.security.sasl.SaslException: GSS
initiate failed [Caused by GSSException: No valid credentials provided
(Mechanism level: Failed to find any Kerberos tgt)]

E0530 15:04:48.304170 20917 impala-server.cc:209] Could not read the HDFS
root directory at hdfs://impalad1.host. Error was:

Failed on local exception: java.io.IOException:
javax.security.sasl.SaslException: GSS initiate failed [Caused by
GSSException: No valid credentials provided (Mechanism level: Failed to
find any Kerberos tgt)]; Host Details : local host is:
"impalad1/ip.address"; destination host is: "impalad1.host":8020;

E0530 15:04:48.304186 20917 impala-server.cc:211] Aborting Impala Server
startup due to improper configuration

catalogd.ERROR

E0530 16:15:22.172746 836 TSaslTransport.java:296] SASL negotiation
failure

Java exception follows:
```

```
javax.security.sasl.SaslException: GSS initiate failed [Caused by
GSSException: No valid credentials provided (Mechanism level: Failed to
find any Kerberos tgt)]

...

Caused by: GSSException: No valid credentials provided (Mechanism level:
Failed to find any Kerberos tgt)

at
com.sun.security.sasl.gsskerb.GssKrb5Client.evaluateChallenge(GssKrb5Clie
nt.java:193)
```

Cause : TGT principals not nonrenewable.

Kerberos misconfiguration

Workaround :

1. Confirm the "Maximum renewable life" is "0 days 00:00:00" by running the following commands as root on the KDC:

```
[root@kdc ~]# kadmin.local -q "getprinc krbtgt/<REALM>@<REALM>"

Authenticating as principal root/admin@<REALM> with password.

Principal: krbtgt/<REALM>@<REALM>

Expiration date: [never]

Last password change: [never]

Password expiration date: [none]

Maximum ticket life: 1 day 00:00:00

Maximum renewable life: 0 days 00:00:00
```

2. Set the following entries to the realm definition in `/var/kerberos/krb5kdc/kdc.conf`

```
max_life = 1d

max_renewable_life = 7d
```

3. Restart the KDC (to set the default for the new principals).

```
# service krb5kdc restart

# service kadmin restart
```

4. Modify the existing principals by running the following command as root on the KDC:

```
# for i in `kadmin.local -q "listprincs" | egrep -v
"K/M|kadmin|Authenticating"`; do kadmin.local -q "modprinc
-maxrenewlife 7day $i"; done
```

5. Confirm renewable life is 7 days by running command, "getprinc":

```
# for i in `kadmin.local -q "listprincs impala*" | grep -v
Authenticating`; do kadmin.local -q "getprinc $i"; done
```

6. Restart the Impala service and test.

If the issue is not resolved, ensure the impalad1.host (the host name in the above example) has the correct JCE policy file per the following steps:

1. Compare the file size in the impalad1.host to the other host impalad is running.

```
# ls -lrt
$JAVA_HOME/jre/lib/security/{US_export_policy.jar,local_policy.jar}
```

If there is a discrepancy between hosts, copy the policy file from the good host to the impalad1.host and ensure each host has the same policy files.

2. Restart the Impala service and see if the issue is resolved.

11.CDSW service shutdown takes a long time to stop

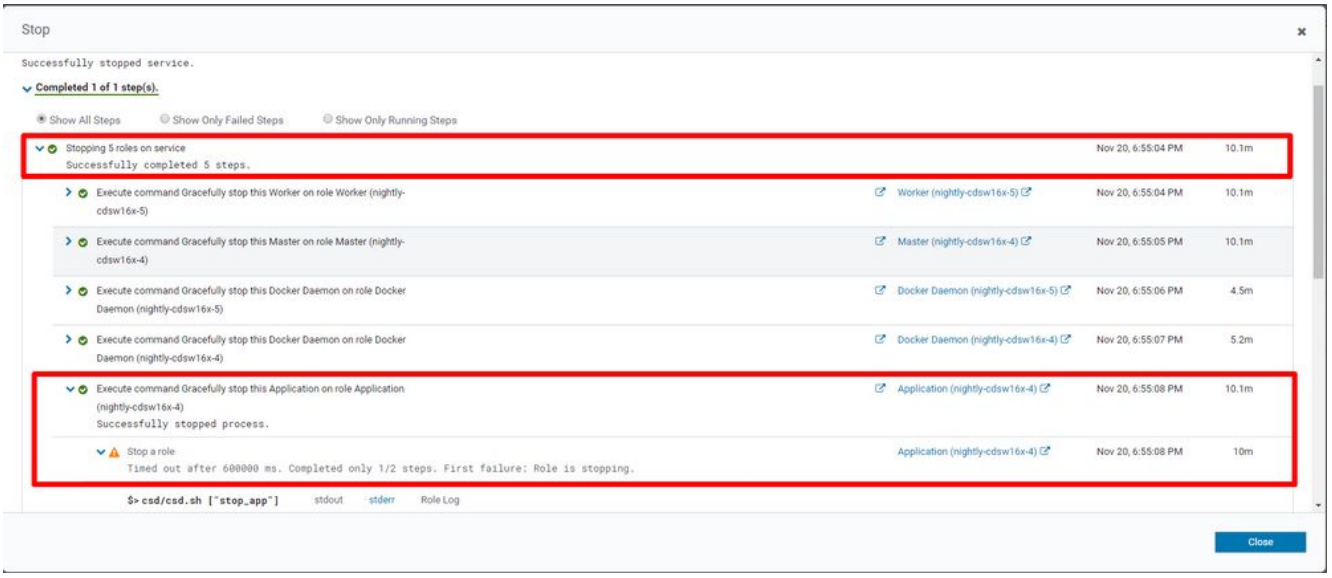
In a normal scenario, a CDSW shutdown completes in around 2 minutes. However, in some cases, it can take upto 10 minutes long. This KB discusses why this happens and what we can do to avoid this.

Symptoms:

In a normal scenario, a CDSW shutdown completes in around 2 minutes. However, in some cases, it can take upto 10 minutes long.

1. Go to **Cloudera Manager** > **CDSW** > **Instances** > Select **Select All Rows** next to Role Type to select all instances within CDSW.
2. Click on the **Actions for Selected** drop down.
3. Click **Stop**.

CDSW roles Application, Master, and Worker may take 10 minutes or more to stop:



This issue occurs when the setting **Automatically Restart Process** is enabled.

To confirm this, go to Cloudera Manager > CDSW > Configuration > Enable Automatically Restart Process.

Workaround:

To resolve this issue, disable the setting.

Go to Cloudera Manager > CDSW > Configuration > Disable **Automatically Restart Process**.

Restart CDSW even if a restart icon is not displayed on Cloudera Manager. (This may not take effect otherwise).

Note: This setting is disabled by default for CDSW and should not be enabled until such time.

12.CDSW Disk Storage Per Project

A feature request to fetch disk usage per project has raised and tracked internally by JIRA DSE-6257. As a workaround, we can fetch this information by executing a Shell Script on one of the CDSW Master Node

Workaround:

Step 1: Download the "CDSW-Project-Disk-Usage.sh" file and move this file to one of the CDSW Master Node using the mentioned command.

$ scp /<local_path>/CDSW-Project-Disk-Usage.sh user@hostname:/tmp

Step 2: Execute this shell script using mentioned command:

$ bash CDSW-Project-Disk-Usage.sh

>> Mentioned output can be observed.

[user@hostname tmp]# bash CDSW-Project-Disk-Usage.sh

**********CDSW Disk Storage Usage Per Project**********

Project Name: Demo_Project

Project Path: /var/lib/cdsw/current/projects/projects/0/1 | Size: 12K

Project Name: Demo_2

Project Path: /var/lib/cdsw/current/projects/projects/0/2 | Size: 24K

****

[user@hostname tmp]#

The disk usage for each project can be reviewed.

13.TSB 2019-348: Impala/Sentry security roles mismatch after Catalog Server restart

An impalad's metadata cache can contain old id + role pairs, and when it is updated with privileges with new role ids from the catalog, the privilege will be added to the wrong role; the one that previously had the same role id.

Symptoms:

This issue occurs when Impala's Catalog Server is restarted without also restarting all the Impala Daemons.

Impala uses generated numeric identifiers for roles. These identifiers are regenerated during catalogd restarts, and the same role can get a different identifier, possibly used by a different role before restart. An impalad's metadata cache can contain old id + role pairs, and when it is updated with privileges with new role ids from the catalog, the privilege will be added to the wrong role; the one that previously had the same role id.

Workaround:

Update to a version of CDH containing the fix.

Restart all impalad if catalogd was restarted.

14.ERROR:"Fatal error during KafkaServerStartable startup. Prepare to shutdown kafka.common.InvalidOffsetException: Attempt to append an offset (30161755101) to position 14656 no larger than the last offset appended" when starting Kafka brokers

Symptoms:

Unable to start Kafka brokers.

The following exception in broker logs:

FATAL kafka.server.KafkaServerStartable: Fatal error during KafkaServerStartable startup.

Prepare to shutdown kafka.common.InvalidOffsetException: Attempt to append an offset (30161755101) to position 14656

no larger than the last offset appended (30161893150) to /data/01/kafka/data/__consumer_offsets-

Cause:

This is a known issue tracked in KAFKA-3919.

This may occur due to the following:

The cluster could have been destabilized while there was an intermittent connectivity issue with ZK for a few minutes. This results in a quick succession of Controller elections and changes, and ISR shrinks and grows. During the ISR shrink and expansion, some brokers seem to have got themselves into the same inconsistent state as above and halted.

KAFKA-3919 is fixed by KAFKA-1211.

Fix: Upgrade to latest version of CDK which is less prone to this issue. (CDK3.0.0 which has the fix for this bug)**.**

Workaround:

To recover the brokers, do the following:

1.Manually delete the corrupted log segments from the topic partitions of the stopped problematic broker, leaving the topic partition directories at the top level as is.

a) The problematic log segment and the partition can be found in the exception:

FATAL kafka.server.KafkaServerStartable: Fatal error during KafkaServerStartable startup.

Prepare to shutdown kafka.common.InvalidOffsetException:

Attempt to append an offset (30161755101) to position 14656 no larger than the last offset appended (30161893150) to

/data/01/kafka/data/__consumer_offsets-40/00000000010739377922.index.

b) In the above exception, the corrupted log segment is 00000000010739377922.index and the partition is __consumer_offsets-40 on this broker.

c)There will be 3 files related to each segment: .log, .timindex and .index files with the same log number in the partition directory on this broker.

d)In the above case 00000000010739377922.index, 00000000010739377922.timeindex, and 00000000010739377922.log files in the same path:

/data/01/kafka/data/__consumer_offsets-40/

e)Remove these 3 files and if needed take a back up under /tmp path.

f)Do not place any back up files in the partition directories otherwise the broker will not be able to start.

2)Verify the current state of this topic/partition using kafka-topics --describe command.

3)Ensure that this broker is not the current leader for this partition (this issue does not happen with leader in most cases).

4)Start the broker and it should be able to recover and replicate the data for these problematic partitions from the current leaders.

5)If you run into same issue again with another partition, follow the same steps to recover the broker from issues with other partition as well.