Troubleshooting DataStax Enterprise

Updated: 2019-01-19-08:00

Table of Contents

Starting and installing DSE.......... 5

- Problems with starting DSE.......... 5
- Error messages when libaio not installed (DSE 6.0 and 6.7).......... 7
- DataStax Enterprise times out when starting.......... 8
- MX4J warning message during DataStax Enterprise installation.......... 9
- DataStax Enterprise fails to start after configuring authentication.......... 9
- JNA fails to initialize.......... 10
- Cannot write to tmp directory.......... 10

Linux related problems.......... 12

- Peculiar Linux kernel performance problem on NUMA systems.......... 12
- Nodes appear unresponsive due to a Linux futex_wait() kernel bug.......... 12
- Reads are getting slower while writes are still fast.......... 13
- Nodes seem to freeze after some period of time.......... 15
- Nodes are dying with OOM errors.......... 16
- Nodetool or JMX connections failing on remote nodes.......... 16
- Handling schema disagreements.......... 17
- Garbage collection pauses.......... 17
- Java reports an error saying there are too many open files.......... 18
- Insufficient user resource limits errors.......... 18
- No DSE processing but high CPU usage.......... 20
- Cannot initialize class org.xerial.snappy.Snappy.......... 21
- Firewall idle connection timeout causing nodes to lose communication.......... 22

Using nodetool sjk.......... 23

DSE Graph.......... 24

- Inconsistent handling of string vertex/edge ids collections in DSE Graph steps.......... 24
- Directly querying Graph data with user-defined vertex ids using solr_query results in null returns.......... 24

Dropping a graph or a portion of a graph.......... 25

Gremlin console hangs or behaves erratically.......... 25

Queries sporadically fail with LOCAL_ONE/LOCAL_QUORUM.......... 26

Consistency level and graph.addVertex().......... 26

Issues creating a graph cluster using Lifecycle Manager (LCM).......... 26

Shutting down Studio Gremlin process.......... 28

Dropping edge property drops edges.......... 28

Creating graph with options while unified authentication is enabled fails.......... 28

Vertex label not specified in DSE Graph snapshot.......... 29

DSE Analytics.......... 30

DSE Search.......... 31

DSE Search node start.......... 31

Handling inconsistencies in query results.......... 35

Tracing Solr HTTP requests.......... 36

Using the ShardRouter MBean.......... 37

Using Solr MBeans.......... 38

Security.......... 39

DSE fails to start.......... 39

SSL certificate doesn't match.......... 39

SSL exceptions occur on start up or no connections.......... 40

Connection errors with cqlsh and other DSE tools.......... 41

Studio.......... 42

Other troubleshooting.......... 45

Ring view differences.......... 45

sstableloader missing rows.......... 45

Prepared statements discarded.......... 45

Purging gossip state on a node.......... 46

Recovering expired data caused by TTL year 2038 problem.......... 47

Troubleshooting starting and installing DataStax Enterprise

The DataStax Enterprise Help Center also provides troubleshooting information.

Troubleshooting starting DataStax Enterprise

The DataStax Enterprise Help Center also provides troubleshooting information.

To identify and resolve start problem, try these steps before contacting DataStax Support.

Start DataStax Enterprise

Start DataStax Enterprise:

- Package installations:

  ```
  $ sudo service dse start
  ```

  For options see Starting DataStax Enterprise as a service.

- Tarball installations:

  ```
  $ install_location/dse cassandra option
  ```

  For options see Starting DataStax Enterprise as a stand-alone process.

Verify DataStax Enterprise status

- Package installations:

  ```
  $ sudo service dse status
  ```

- Tarball installations:

  ```
  $ install_location/nodetool status
  ```

Review the log files

If DataStax Enterprise is not running, or starts running and then stops, look for errors at the end of the system log file:

```
$ cat /var/log/cassandra/system.log
```

Discover the process ID

It is possible that an active DSE or Cassandra process is interfering with starting DataStax Enterprise. For example, on a local development node:

1. Verify the node status:
   - Package installations:

     ```
     $ nodetool status
     ```

   - Tarball installations:

     ```
     $ install_location/nodetool status
     ```

   ```
   $ Datacenter: Cassandra
   =====================
   Status=Up/Down
   |/ State=Normal/Leaving/Joining/Moving
   --  Address    Load       Owns  Host ID
    Token                 Rack
   UN  127.0.0.1  206.23 KB  54%   392b0ca3-3090-440f-8a17-93702234aeec
    1093280155279701211  rack1
   ```

2. However, there can be a situation where Cassandra is running, but DataStax Enterprise is not. For example, if your environment had earlier product versions installed or the product was not shut down gracefully, look for the Cassandra process ID and terminate the Cassandra process:

   ```
   $ pgrep -f cassandra
   29474
   sudo kill -9 29474
   ```

Subprocesses not killed when DataStax Enterprise is shut down improperly

To prevent this problem, avoid using `kill 9` and shut down DataStax Enterprise properly.

- Package installations:

  ```
  $ nodetool status
  ```

- Tarball installations:

  ```
  $ install_location/nodetool status
  ```

If DataStax Enterprise is shut down with `kill -9`, you must restart the node or manually kill any remaining sub-processes:

- Package installations:

  For example, if DataStax Enterprise was started using `sudo services dse start` or `sudo /etc/init.d/dse start` and the main process was killed using ``kill -9 `cat /var/run/dse/dse.pid` ``:

  1. To view the subprocesses left behind (all DSE processes run under user *cassandra* [default]):

```
$ pgrep -c -ucassandra >/dev/null && ps -o pid,ppid,user,args
 `pgrep -ucassandra`
```

2. To shut down the subprocesses:

   ```
   $ sudo pkill -ucassandra
   ```

- Tarball installations:

  For example, if DataStax Enterprise was started using `sudo dse cassandra -k -t` and the main process was killed using ``sudo kill -9 `cat /var/run/dse/dse.pid` `` or `sudo pkill -9 -f jmxremote.port=7199`:

  1. To view the subprocesses left behind:

     ```
     $ pgrep -c -f dse-5.1 >/dev/null && ps -o pid,ppid,user,args `pgrep
      -f dse-5.0
     ```

     All DataStax Enterprise processes run under user `cassandra`.

  2. To shut down the subprocesses:

     ```
     $ sudo pkill -f dse-5.1
     ```

Note: The `kill` command (SIGTERM) shuts down the subprocesses.

Error messages when libaio not installed (DSE 6.0 and 6.7)

DSE 6.7 and 6.0 require installing the libaio package. If the libaio package is not installed, an error like this is logged:

```
WARN  [main] 2018-04-10 03:09:20,412 StartupChecks.java:633 - Epoll
 doesn't seem to be available: this may result in subpar performance.
 Libaio doesn't appear to be installed.
java.lang.IllegalArgumentException: Failed to load any of the given
 libraries: [netty-transport-native-epoll, netty_transport_native_epoll]
        at
 io.netty.util.internal.NativeLibraryLoader.loadFirstAvailable(NativeLibraryLoader.java:18
        at
 io.netty.channel.epoll.Native.loadNativeLibrary(Native.java:295)
        at io.netty.channel.epoll.Native.<clinit>(Native.java:60)
        at io.netty.channel.epoll.Epoll.<clinit>(Epoll.java:33)
        at org.apache.cassandra.concurrent.TPC.<clinit>(TPC.java:80)
        at
 org.apache.cassandra.service.CassandraDaemon.initializeTPC(CassandraDaemon.java:180)
```

```
        at
org.apache.cassandra.service.CassandraDaemon.activate(CassandraDaemon.java:648)
        at
org.apache.cassandra.service.CassandraDaemon.main(CassandraDaemon.java:797)
```

This part of the error message `Libaio doesn't appear to be installed.` indicates that libaio was not installed during installation. In earlier versions of DSE, the libaio package was not required.

To resolve this error:

1. Install libaio1:
   - For Debian-based systems:

     ```
     sudo apt-get install libaio1
     ```
   - For Red Hat-based systems:

     ```
     sudo yum install libaio1
     ```
2. Restart DSE.

DataStax Enterprise times out when starting

The DataStax Enterprise Help Center also provides troubleshooting information.

When starting DataStax Enterprise (DSE) as a service, a script sets up the environment and launches the service. After the DSE service is launched, the script verifies if the service is running. The service takes a few seconds to start, and might display:

```
WARNING: Timed out while waiting for DSE to start.
```

This error does not necessarily mean that the DSE service failed to start. Verify by checking the log files in `/var/log/cassandra/system.out`.

The start script checks if the DSE service is running once per second, so the number of checks is equal to the number of seconds.

To increase the time until the service is declared not to launch successfully, uncomment and edit the WAIT_FOR_START option in the `/etc/default/dse` file, and then restart the DataStax Enterprise:

```
# Uncomment if you want longer/shorter waits checking if the service is up
```

```
WAIT_FOR_START=14
```

MX4J warning message during DataStax Enterprise installation

The DataStax Enterprise Help Center also provides troubleshooting information.

When DataStax Enterprise loads, you may notice a message that MX4J will not load and that `mx4j-tools.jar` is not in the classpath.

You can ignore this message. MX4j provides an HTML and HTTP interface to JMX and is not necessary to run DataStax Enterprise.

DataStax recommends using DSE OpsCenter. It has more monitoring capabilities than MX4J.

DataStax Enterprise fails to start after configuring authentication

The DataStax Enterprise Help Center also provides troubleshooting information.

Authorizer requires Authenticator

Problem description

When settings do not match DataStax Enterprise fails to start. For example, dse.yaml with `authentication_options.enabled: false` and `authorization_options.enabled: true`, prevents DSE from starting and has the following errors in `cassandra/system.log`:

```
Caused by: org.apache.cassandra.exceptions.ConfigurationException:
 com.datastax.bdp.cassandra.auth.DseAuthenticator does not
 currently require authentication, so it can't be used with
 com.datastax.bdp.cassandra.auth.DseAuthorizer which does currently
 require authorization. You need to either choose new classes or update
 their configurations so they are compatible.
```

Solution

Set both `authentication_options` and `authorization_options` enabled to the same setting in the dse.yaml file.

External authentication services unreachable

Problem description

DataStax Enterprise start up fails when `authentication_options.enabled: true` and a scheme is not configured or the corresponding service is unavailable that is configured in the dse.yaml:

- `role_manager.mode`
- `authentication_options.default_scheme`

- `authentication_options.other_scheme`

For example, when the default_schema is set to kerberos and the KDS server defined in the `kerberos_options` is unavailable, the `cassandra/system.log` shows the following start up error:

```
1) An exception was caught and reported. Message: The dse
 service keytab at this location resources/dse/conf/dse.keytab
 either doesn't exist or cannot be read by the dse service at
 com.datastax.bdp.DseModule.configure(Unknown Source)
```

Solution

Ensure that the configured KDC or LDAP host is available or remove the scheme setting from the DSE Authenticator or Role Manager options.

JNA fails to initialize

The DataStax Enterprise Help Center also provides troubleshooting information.

DataStax Enterprise fails to start because the JNA temporary directory is not available. If the tmp directory is inaccessible the following error appears in the system log:

```
tail -3 /var/log/cassandra/system.log
```

The error is similar to:

```
ERROR main 2015-12-18 09:57:00,879 CassandraDaemon.java:213 - JNA failing
 to initialize properly. Use -Dcassandra.boot_without_jna=true to
 bootstrap even so.
INFO Thread-2 2015-12-18 09:57:00,880 DseDaemon.java:418 - DSE shutting
 down...
INFO Thread-2 2015-12-18 09:57:00,881 PluginManager.java:103 - All plugins
 are stopped.
```

Configure the database JNA temporary path to an executable directory.

- In cassandra-env.sh, add the line:

  ```
  JVM_OPTS="$JVM_OPTS -Djna.tmpdir=/path_to_directory"
  ```

  Where *`path_to_directory`* is the absolute path to a directory to which the database user has read, write, and execute permissions.

Unable to write to the standard tmp directory for JNA in DataStax Enterprise

The DataStax Enterprise Help Center also provides troubleshooting information.

Set JNA temporary path to be executable. In cassandra-env.sh, add the line:

```
JVM_OPTS="$JVM_OPTS -Djna.tmpdir=/var/tmp"
```

Troubleshooting Linux related problems

The DataStax Enterprise Help Center also provides troubleshooting information.

Peculiar Linux kernel performance problem on NUMA systems

The DataStax Help Center also provides troubleshooting information.

Problems due to zone_reclaim_mode.

The Linux kernel can be inconsistent in enabling/disabling zone_reclaim_mode. This can result in odd performance problems:

- Random huge CPU spikes resulting in large increases in latency and throughput.
- Programs hanging indefinitely apparently doing nothing.
- Symptoms appearing and disappearing suddenly.
- After a reboot, the symptoms generally do not show again for some time.

To ensure that zone_reclaim_mode is disabled:

```
$ echo 0 > /proc/sys/vm/zone_reclaim_mode
```

Nodes appear unresponsive due to a Linux futex_wait() kernel bug

The DataStax Enterprise Help Center also provides troubleshooting information.

Nodes randomly freeze and become unresponsive for an unknown reason.

The bug exists in RHEL 6.6, CentOS 6.6 and above.

Nodes affected by this bug have the following characteristics:

- No garbage collection activity in the logs.
- No compactions in progress.
- Unable to run nodetool commands.
- No response on native transport, Thrift or JMX ports.
- Low or close to zero CPU utilization.
- High CPU utilization which eventually leads to the node becoming unresponsive.

A thread dump on the node might show:

```
Thread 104823: (state = BLOCKED)
 - sun.misc.Unsafe.park(boolean, long) @bci=0 (Compiled frame; information
 may be imprecise)
 - java.util.concurrent.locks.LockSupport.parkNanos(java.lang.Object,
 long) @bci=20, line=226 (Compiled frame)
 - java.util.concurrent.locks.AbstractQueuedSynchronizer
$ConditionObject.awaitNanos(long) @bci=68, line=2082 (Compiled frame)
 - java.util.concurrent.LinkedBlockingQueue.poll(long,
 java.util.concurrent.TimeUnit) @bci=62, line=467 (Compiled frame)
 - java.util.concurrent.ThreadPoolExecutor.getTask() @bci=141, line=1068
 (Compiled frame)
 -
 java.util.concurrent.ThreadPoolExecutor.runWorker(java.util.concurrent.ThreadPoolExecutor
$Worker) @bci=26, line=1130 (Compiled frame)
 - java.util.concurrent.ThreadPoolExecutor$Worker.run() @bci=5, line=615
 (Interpreted frame)
 - java.lang.Thread.run() @bci=11, line=745 (Interpreted frame)
```

Cause

This problem is caused by a Linux `futex_wait()` bug that causes user processes to deadlock and hang. A `futex_wait()` call (and any processes making this call) can stay blocked forever. JVM synchronization method calls such as `lock()`, `park()` and `unpark()` all make `futex_wait()` calls at some point and can trigger the unresponsiveness caused by this bug.

Solution

Upgrade to Linux kernels containing the get_futex_key_refs() fix, such as RHEL 6.6.z and CentOS 6.6.z.

Use the following command to check for the installed patches on a RHEL server:

```
$ sudo rpm -q --changelog kernel-`uname -r` | grep futex | grep ref
```

Sample output from this command:

```
- [kernel] futex: Mention key referencing differences between shared and
 private futexes (Larry Woodman) [1167405]
- [kernel] futex: Ensure get_futex_key_refs() always implies a barrier
 (Larry Woodman) [1167405]
```

If the patch had not been installed, the `rpm` command would show nothing.

For further information on distributions that contain the fix, consult the relevant vendor or distributor of the operating system.

Reads are getting slower while writes are still fast

The DataStax Enterprise Help Center also provides troubleshooting information.

Too many SSTables can cause slow reads. Take the following steps to determine and correct slow reads:

Determine the total number of SSTables for each table.

Check this number with nodetool tablestats.

Get the number of SSTables consulted for each read.

Check this number with nodetool tablehistograms. A median value over 2 or 3 is likely causing problems.

Make sure SSTables are not flushing too frequently

Check `debug.log` for *enqueuing flush* messages and note the size and frequency of flushes:

- If the `SlabPoolCleaner` thread is frequently enqueueing many small flushes, increase memtable_cleanup_threshold.

  The value of memtable_cleanup_threshold should be inversely proportional to the number of tables that receive heavy writes.

- If spare memory is available, consider increasing memtable_heap_space_in_mb (deprecated) or memtable_offheap_space_in_mb (deprecated).
- If the `COMMIT-LOG-ALLOCATOR` thread is frequently enqueuing flushes, increase commitlog_total_space_in_mb.

  The number of flushes enqueued by `COMMIT-LOG-ALLOCATOR` should be a small minority of the total flushes enqueued.

- Check the pending compactions using nodetool compactionstats.

  Even if flushes are not excessively frequent, compactions might not be able to keep up. If a high number of pending compactions exist, compactions are not keeping up. Note that this number is only an estimate for LeveledCompactionStrategy.

Make sure the compaction_throughput_mb_per_sec is set appropriately for your storage.

The default value of 16 MB/sec for compaction_throughput_mb_per_sec is chosen for spinning disks; SSDs can use a much higher setting such as 128 MB/sec or more.

1. Temporarily adjust the value using nodetool setcompactionthroughput.
2. Watch the I/O utilization using `iostat -x -t 10`, which shows the averages for 10 second intervals and prints timestamps:
   - `%iowait` over 1 indicates that the node is starting to get I/O bound.
   - The acceptable bounds for `await` (Average Wait in Milliseconds) are:
     - # Most SSDs: below 10 ms.
     - # Most 7200 RPM spinning disks: below 200 ms.
3. Once you have found a good throughput for your system, set it permanently in `cassandra.yaml`.

4. If your I/O is not able to keep up with the necessary compaction throughput, you probably need to get faster disks or add more nodes.

If you have set a high compaction throughput but I/O utilization is low and compactions are still not keeping up, the compactions may be CPU-bound.

Check the per-core CPU utilization of CompactionExecutor threads.

If the threads are utilizing 100% of a single core, the compaction may be CPU bound. Increasing concurrent_compactors will allow multiple concurrent compactions of different sets of SSTables, but compaction of each set of SSTables is inherently single-threaded. If you are using LeveledCompactionStrateg (LCS), you need to either switch to SizeTieredCompactionStrategy (STCS) or add more nodes to spread compaction load.

Note: Increasing concurrent compactors beyond the number of physical CPU cores (not Hyperthreaded cores) can be counter productive. Using all available CPU for compaction means no CPUs remain to handle reads and writes. If you need to continue to service requests while catching up on compactions, be sure to leave 1 or 2 physical CPUs free for reads and/or writes.

Make sure that there is enough free memory for file cache (page cache).

The `free -m` command shows the amount of memory available for caches. You should have enough memory available for file cache to hold your hot working set in memory.

Switch to SizeTieredCompactionStrategy.

If using LeveledCompactionStrategy (LCS) and the above steps haven't worked, consider switching to SizeTieredCompactionStrategy (STCS). LCS uses more resources to compact than STCS. Often nodes that are falling behind while compacting with LCS can easily keep up using STCS.

Nodes seem to freeze after some period of time

The DataStax Help Center also provides troubleshooting information.

Some portion of the JVM is being swapped out by the Linux operating system (OS).

Linux

Check your system.log for messages from the `GCInspector`. If the `GCInspector` is indicating that either the `ParNew` or `ConcurrentMarkSweep` collectors took longer than 15 seconds, there is a high probability that some portion of the JVM is being swapped out by the OS.

DataStax strongly recommends disabling swap entirely (`sudo swapoff --all`). Because the database has multiple replicas and transparent failover, it is preferable for a replica to be killed immediately when memory is low rather than go into swap. This allows traffic to be immediately redirected to a functioning replica instead of continuing to hit the replica that has high latency due to swapping. If your system has a lot of DRAM, swapping still lowers performance significantly because the OS swaps out executable code so that more DRAM is available for caching disks.

```
$ sudo swapoff --all
```

To make this change permanent, remove all swap file entries from `/etc/fstab`.

> **Note:** If you insist on using swap, you can set `vm.swappiness=1`. This allows the kernel swap out the absolute least used parts.

If the GCInspector isn't reporting very long GC times, but is reporting moderate times frequently (`ConcurrentMarkSweep` taking a few seconds very often) then it is likely that the JVM is experiencing extreme GC pressure and will eventually OOM. See Nodes are dying with OOM errors *(page 16)*.

Nodes are dying with OOM errors

The DataStax Enterprise Help Center also provides troubleshooting information.

Nodes are dying with OutOfMemory exceptions.

Check for these typical causes:

Row cache is too large, or is caching large rows
: Row cache is generally a high-end optimization. Try disabling it and see if the OOM problems continue.

There is a large user query running on the node which takes up all the heap
: In production, understand and test all queries upfront to avoid arbitrary query patterns. Test to discover each query's max response size. Paging in CQL can often prevent a query from pulling too much data at once.

If none of these apply to your situation, try loading the heap dump in MAT and see which class is consuming the bulk of the heap for clues.

Nodetool or JMX connections failing on remote nodes

The DataStax Enterprise Help Center also provides troubleshooting information.

Nodetool commands can be run locally but not on other nodes in the cluster.

If you can run nodetool commands locally but not on other nodes in the ring, it might be common JMX connection problem. Add an entry like the following in cassandra-env.sh on each node:

```
JVM_OPTS = "$JVM_OPTS -Djava.rmi.server.hostname=public name"
```

The default settings start up JMX only on the local node. In `cassandra-env.sh` or `cassandra-env.ps1`, add JMX authentication to contact remote nodes. See Jmx Security for details.

If you still cannot run `nodetool` commands remotely after making this configuration change, do a full evaluation of your firewall and network security. The `nodetool` utility communicates through JMX on port 7199.

Handling schema disagreements

The DataStax Enterprise Help Center also provides troubleshooting information.

Check for and resolve schema disagreements on Linux platforms.

In the event that a schema disagreement occurs, check for and resolve schema disagreements as follows:

1. Run the `nodetool describecluster` command.

   ```
   $ nodetool describecluster
   ```

   If any node is UNREACHABLE, the output looks like this:

   ```
   Snitch: org.apache.cassandra.locator.DynamicEndpointSnitch
   Partitioner: org.apache.cassandra.dht.Murmur3Partitioner
   Schema versions:
     UNREACHABLE: 1176b7ac-8993-395d-85fd-41b89ef49fbb:
    [10.202.205.203]
                  9b861925-1a19-057c-ff70-779273e95aa6: [10.80.207.102]
                  8613985e-c49e-b8f7-57ae-6439e879bb2a: [10.116.138.23]
   ```

2. Restart unreachable nodes.

3. Repeat steps 1 and 2 until `nodetool describecluster` shows that all nodes have the same schema version number—only one schema version appears in the output.

Garbage collection pauses

The DataStax Enterprise Help Center also provides troubleshooting information.

Troubleshooting GC pauses of more than a second, or multiple pauses within a second that add up to a large fraction of that second.

Garbage collection (GC) is the process by which Java removes data that is no longer needed from memory. A garbage collection pause, also known as a stop-the-world event, happens when a region of memory is full and the JVM requires space to continue. During a pause all operations are suspended. Because a pause affects networking, the node can appear as down to other nodes in the cluster. Additionally, any Select and Insert statements will wait, which increases read and write latencies. Any pause of more than a second, or multiple pauses within a second that add to a large fraction of that second, should be avoided. The basic cause of the problem is that the rate of data stored in memory outpaces the rate at which data can be removed.

The two most common log messages that indicate excessive pausing is occurring are:

```
INFO [ScheduledTasks:1] 2013-03-07 18:44:46,795 GCInspector.java (line
 122) GC for ConcurrentMarkSweep: 1835 ms for 3 collections, 2606015656
 used; max is 10611589120
INFO [ScheduledTasks:1] 2013-03-07 19:45:08,029 GCInspector.java (line
 122) GC for ParNew: 9866 ms for 8 collections, 2910124308 used; max is
 6358564864
```

Causes of garbage collection pause include:

- If the problem is recent, check for any recent applications changes.
- Excessive tombstone activity: often caused by heavy delete workloads.
- Large row updates or large batch updates: reduce the size of the individual write below 1 Mb (at the most).
- Extremely wide rows: manifests as problems in repairs, selects, caching, and elsewhere.

Server side factors include:

- Missing or strange JVM parameters. Compare those set to the default settings shipped with latest product version.
- JNA not found.
- Swap enabled.

For more information, see Tuning Java Virtual Machine.

Java reports an error saying there are too many open files

The DataStax Enterprise Help Center also provides troubleshooting information.

Java may not have enough open file descriptors on Linux platforms.

DataStax Enterprise generally needs more than the default (1024) amount of file descriptors. To increase the number of file descriptors, change the security limits on your nodes as described in the Recommended production settings.

Another, much less likely possibility, is a file descriptor leak. Run `lsof -n | grep java` to check that the number of file descriptors opened by Java is reasonable and report the error if the number is greater than a few thousand.

Insufficient user resource limits errors

The DataStax Help Center also provides troubleshooting information.

Insufficient resource limits may result in a number of errors in DataStax Enterprise on Linux platforms.

Errors

Insufficient as (address space) or memlock setting

```
ERROR [SSTableBatchOpen:1 ] 2012-07-25 15:46:02,913
 AbstractCassandraDaemon.java  (line 139)
Fatal exception in thread Thread [SSTableBatchOpen:1,5,main ]
java.io.IOError: java.io.IOException: Map failed  at ...
```

Insufficient memlock settings

```
WARN [main ] 2011-06-15 09:58:56,861 CLibrary.java  (line 118)
 Unable to lock JVM memory  (ENOMEM).
This can result in part of the JVM being swapped out, especially
 with mmapped I/O enabled.
Increase RLIMIT_MEMLOCK or run Cassandra as root.
```

Insufficient nofiles setting

```
WARN 05:13:43,644 Transport error occurred during acceptance of
 message.
org.apache.thrift.transport.TTransportException:
 java.net.SocketException:
Too many open files ...
```

Insufficient nproc setting

```
ERROR [MutationStage:11 ] 2012-04-30 09:46:08,102
 AbstractCassandraDaemon.java  (line 139)
Fatal exception in thread Thread [MutationStage:11,5,main ]
java.lang.OutOfMemoryError: unable to create new native thread
```

Recommended settings

Use the `ulimit -a` command to view the current limits. Although limits can also be temporarily set using this command, DataStax recommends making the changes permanent:

Package installations:

Ensure that the following settings are included in the `/etc/security/limits.d/cassandra.conf` file:

```
<cassandra_user> - memlock unlimited
<cassandra_user> - nofile 100000
<cassandra_user> - nproc 32768
<cassandra_user> - as unlimited
```

Tarball installations:

In RHEL version 6.x, ensure that the following settings are included in the `/etc/security/limits.conf` file:

```
<cassandra_user> - memlock unlimited
<cassandra_user> - nofile 100000
```

```
<cassandra_user> - nproc 32768
<cassandra_user> - as unlimited
```

If you run DataStax Enterprise as root, some Linux distributions, such as Ubuntu, require setting the limits for root explicitly instead of using *`cassandra_user`*:

```
root - memlock unlimited
root - nofile 100000
root - nproc 32768
root - as unlimited
```

For RHEL 6.x-based systems, also set the **nproc** limits in `/etc/security/limits.d/90-nproc.conf`:

```
cassandra_user - nproc 32768
```

For all installations, add the following line to `/etc/sysctl.conf`:

```
vm.max_map_count = 1048575
```

For installations on Debian and Ubuntu operating systems, the pam_limits.so module is not enabled by default. Edit the `/etc/pam.d/su` file and uncomment this line:

```
session    required   pam_limits.so
```

This change to the PAM configuration file ensures that the system reads the files in the `/etc/security/limits.d` directory.

To make the changes take effect, reboot the server or run the following command:

```
$ sudo sysctl -p
```

To confirm the limits are applied to the DataStax Enterprise process, run the following command where *`pid`* is the process ID of the currently running DataStax Enterprise process:

```
$ cat /proc/pid/limits
```

No DataStax Enterprise processing but high CPU usage

The DataStax Enterprise Help Center also provides troubleshooting information.

Extremely high CPU usage but no DataStax Enterprise processing on Linux platforms.

Check the CPU usage for the process `khugepaged`. It may run as high as 100%, blocking other processes.

Cause:

Many modern Linux distributions ship with Transparent Hugepages enabled by default. When Linux uses Transparent Hugepages, the kernel tries to allocate memory in large chunks (usually 2 MB), rather than 4K. This can improve performance by reducing the number of pages the CPU must track. However, some applications still allocate memory based on 4K pages. This can cause noticeable performance problems when Linux tries to defrag 2 MB pages. For more information, see Cassandra Java Huge Pages and this RedHat bug report.

Possible solutions:

- A temporary fix: drop caches by entering:

  ```
  sync && echo 3 > /proc/sys/vm/drop_caches
  ```

- A better solution: disable defrag for hugepages by entering:

  ```
  echo never | sudo tee /sys/kernel/mm/transparent_hugepage/defrag
  ```

- Another alternative: add `-XX:+AlwaysPreTouch` to the `jvm.options` file. This change should be tested carefully before being put into production. For details, see Tuning Java Virtual Machine and blog post.

Cannot initialize class org.xerial.snappy.Snappy

The DataStax Enterprise Help Center also provides troubleshooting information.

On Linux platforms, an error may occur when Snappy compression/decompression is enabled although its library is available from the classpath.

```
java.util.concurrent.ExecutionException: java.lang.NoClassDefFoundError:
    Could not initialize class org.xerial.snappy.Snappy
...
Caused by: java.lang.NoClassDefFoundError: Could not initialize class
 org.xerial.snappy.Snappy
   at
 org.apache.cassandra.io.compress.SnappyCompressor.initialCompressedBufferLength
       (SnappyCompressor.java:39)
```

The native library `snappy-1.0.4.1-libsnappyjava.so` for Snappy compression is included in the `snappy-java-1.0.4.1.jar` file. When the JVM initializes the JAR, the library is added to the default temp directory. If the default temp directory is mounted with a noexec option, it results in the above exception.

One solution is to specify a different temp directory that has already been mounted without the noexec option, as follows:

- If you use the dse start command `$_BIN/dse cassandra` or `$_BIN/cassandra`, simply append the command line:

  ```
  $ bin/dse cassandra -t -Dorg.xerial.snappy.tempdir=/path/to/newtmp
  ```

- If starting from a package using service dse start or service cassandra start, add a system environment variable *`JVM_OPTS`* with the value:

```
JVM_OPTS=-Dorg.xerial.snappy.tempdir=/path/to/newtmp
```

The default cassandra-env.sh looks for the variable and appends to it when starting the JVM.

Firewall idle connection timeout causing nodes to lose communication during low traffic times on Linux

The DataStax Enterprise Help Center also provides troubleshooting information.

During low traffic intervals, a firewall configured with an idle connection timeout can close connections to local nodes and nodes in other data centers. The default idle connection timeout is usually 60 minutes and configurable by the network administrator.

To prevent connections between nodes from timing out, set the TCP keep alive variables:

1. Get a list of available kernel variables:

   ```
   $ sysctl -A | grep net.ipv4
   ```

   The following variables should exist:

   - *`net.ipv4.tcp_keepalive_time`*

     Time of connection inactivity after which the first keep alive request is sent.

   - *`net.ipv4.tcp_keepalive_probes`*

     Number of keep alive requests retransmitted before the connection is considered broken.

   - *`net.ipv4.tcp_keepalive_intvl`*

     Time interval between keep alive probes.

2. To change these settings:

   ```
   $ sudo sysctl -w net.ipv4.tcp_keepalive_time=60
    net.ipv4.tcp_keepalive_probes=3 net.ipv4.tcp_keepalive_intvl=10
   ```

   This sample command changes TCP keepalive timeout to 60 seconds with 3 probes, 10 seconds gap between each. This setting detects dead TCP connections after 90 seconds (60 + 10 + 10 + 10). There is no need to be concerned about the additional traffic as it's negligible and permanently leaving these settings shouldn't be an issue.

Using nodetool sjk

Use nodetool sjk mx to gather database information from MBeans. See Using nodetool sjk 6.7 | 6.0 | 5.1.

DSE Graph troubleshooting

The DataStax Enterprise Help Center also provides troubleshooting information.

Inconsistent handling of string vertex/edge ids collections in DSE Graph steps

The DataStax Enterprise Help Center also provides troubleshooting information.

The string formatting for vertices with user-defined vertex ids has changed. Invoking `toString` on a user-defined vertex id containing a text property, or on an edge id that is incident upon a vertex with a user-defined vertex id now returns a value that double-quotes that text property value and escapes the value's internal double-quotes. This change avoids cases where the old format could lead to irresolvable parsing ambiguity. For example, examine the input value and returned value for the following graph statements:

```
gremlin> graph.addVertex(T.label, "v", "p", "a\"b")
==>v[{~label=v, p="a""b"}]
gremlin> g.V().id().toList()
==>{~label=v, p="a""b"}
gremlin> g.V().id().next().toString()
==>{~label=v, p="a""b"}
gremlin> g.V().toList()
==>v[{~label=v, p="a""b"}]
```

Directly querying Graph data with user-defined vertex ids using solr_query results in null returns

The DataStax Enterprise Help Center also provides troubleshooting information.

The valid way to issue a direct CQL query that includes a solr_query is to retrieve only the user-defined vertex id fields in the select statement. Moving from DSE 5.0 to 6.0 may cause issues with previously created queries.

Because of these complexities, DataStax currently does not recommend querying the underling Graph generated CQL tables or SOLR Indexes. Such queries will be required to rewrite their queries in a future DSE Graph release.

In DSE 50. and DSE 5.1, when someone issues a CQL statement using the SOLR API in the WHERE clause against a graph table that contains a DSE Search index (assuming search index was created by DSE Graph)

- The end user will see unexpected results like the following:

  ```
  cassandra_admin@cqlsh:ecdc_graph> select first_name, last_name,
   record_type from contact_p where solr_query = 'last_name:Abbott';
  ```

```
 first_name | last_name | record_type
------------+-----------+-------------
       null |      null |        nullnull |      null |
 nullnull |      null |        nullnull |      null |        nullnull
 |      null |        nullnull |      null |        nullnull |
 null |        nullnull |      null |           G
       null |      null |        nullnull |      null |
 nullnull |      null |        null
      Lacey |      null |        nullnull |      null |
 nullnull |    Abbott |        null
```

In DSE 6.0, the user will get an error message telling them that they cannot do that query.

Dropping a graph or a portion of a graph

The DataStax Enterprise Help Center also provides troubleshooting information.

Dropping a graph or a portion of a graph such as some vertices can hang based on errors in logged/unlogged or causes error depending on logged/unlogged DSE database batches. A method to resolve the problem is to DROP TABLE or TRUNCATE the underlying DSE database tables storing graph data.

The data for a graph is stored in `<graph_name>.<vertex_label>_p` and `<graph_name>.<vertex_label>_e`. For example, `recipe` data stored in a graph `food` will be `food.recipe_p` and `food.recipe_e`.

In some cases, additional steps must be taken to delete a graph. A graph consists of three DSE database keyspaces: `<graph_name>`, `<graph_name>_system`, and `<graph_name>_pvt`. All three keyspaces must be deleted, with `cqlsh` if necessary, to completely delete the graph.

If a graph hangs during provisioning, use the following `cqlsh` commands:

```
cqlsh> delete from dse_system.shared_data where dataspace = 'Cluster'
 and valid_until = 13814000-1dd2-11b2-0000-000000000000
 and namespace = 'system' and name = '<graph_name>';
cqlsh> update dse_system.shared_data set last_updated = now() where
 dataspace = 'Cluster';
```

Warning: `Shared_data` is not normally manually updated. However, this procedure can be used in the case of a node failure during a graph provisioning operation.

Gremlin console hangs or behaves erratically

The DataStax Enterprise Help Center also provides troubleshooting information.

The Gremlin console can quit responding if a bad query is entered. Here are some hints for how to regain control:

:clear

This command will clear the current buffer and reset the prompt counter.

:remote config alias reset

This command will reset the graph traversal *g* to no graph. Use this command before running system commands.

:remote config alias g some_graph.g

This command will reset the graph traversal *g* to another graph. If queries are hanging in the current graph, switching to another graph allows the console to respond.

Queries sporadically fail with LOCAL_ONE/ LOCAL_QUORUM

The DataStax Enterprise Help Center also provides troubleshooting information.

The DataStax Enterprise database will sporadically fail in a multi-datacenter cluster using `SimpleStrategy` if a query uses `LOCAL_ONE` or `LOCAL_QUORUM`. This affects DSE Graph queries in a multi-datacenter cluster as well. If you use DataStax Studio to auto-create graphs in Development mode, this issue is likely to appear.

To avoid this issue, in or multi-datacenter clusters, create graphs with appropriate DSE database settings.

Consistency level and graph.addVertex()

The DataStax Enterprise Help Center also provides troubleshooting information.

Vertex id allocation requires `QUORUM` consistency to ensure that allocated ids are unique. Vertex ids are allocated in blocks. If a block runs out, write failures can occur and a consistency level of `QUORUM` cannot be met. Graph clusters can operate while falling below `QUORUM`, but it is possible for queries involving adding a vertex can fail if a consistency level of `ONE` is used.

Do not use a consistency level of `ONE` for operations that add vertex data.

Issues creating a graph cluster using Lifecycle Manager (LCM)

The DataStax Enterprise Help Center also provides troubleshooting information.

Currently, OpsCenter LCM is blocking proper setup and usage of DSE Graph. Some steps can be taken to fix the problems until changes are made in DSE 5.1.3.

The default serializers in dse.yaml file must be updated to match the following for the `gremlin_server` settings:

```
gremlin_server:
    maxContentLength: 65536000
    maxChunkSize: 4096000
    port: 8182
    serializers:
      - { className:
 org.apache.tinkerpop.gremlin.driver.ser.GryoMessageSerializerV1d0,
 config: { ioRegistries:
 [org.apache.tinkerpop.gremlin.tinkergraph.structure.TinkerIoRegistry],
 classResolverSupplier:
 com.datastax.bdp.graph.impl.tinkerpop.io.DseClassResolverProvider }}
      - { className:
 org.apache.tinkerpop.gremlin.driver.ser.GryoLiteMessageSerializerV1d0,
 config: { ioRegistries:
 [org.apache.tinkerpop.gremlin.tinkergraph.structure.TinkerIoRegistry],
 classResolverSupplier:
 com.datastax.bdp.graph.impl.tinkerpop.io.DseClassResolverProvider }}
      - { className:
 org.apache.tinkerpop.gremlin.driver.ser.GryoMessageSerializerV1d0,
 config: { serializeResultToString: true }}
      - { className:
 org.apache.tinkerpop.gremlin.driver.ser.GraphSONMessageSerializerGremlinV1d0,
 config: { ioRegistries:
 [org.apache.tinkerpop.gremlin.tinkergraph.structure.TinkerIoRegistry] }}
      - { className:
 org.apache.tinkerpop.gremlin.driver.ser.GraphSONMessageSerializerV1d0,
 config: { ioRegistries:
 [org.apache.tinkerpop.gremlin.tinkergraph.structure.TinkerIoRegistry] }}
    scriptEngines:
      gremlin-groovy:
        config:
          compilerCustomizerProviders:

 "org.apache.tinkerpop.gremlin.groovy.jsr223.customizer.ThreadInterruptCustomizerProvider"
 []

 "org.apache.tinkerpop.gremlin.groovy.jsr223.customizer.InterpreterModeCustomizerProvider"
 []
```

The `rpc_address` setting must be set to `0.0.0.0` in the dse.yaml file.

LCM does not set the default log level to `INFO`. Change the setting in the `logback-gremlin-server.xml` file found in *`install_location`*`/resources/graph/conf`:

```
<appender name="SYSTEMLOG"
 class="ch.qos.logback.core.rolling.RollingFileAppender">
    <filter class="ch.qos.logback.classic.filter.ThresholdFilter">
      <level>INFO</level>
    </filter>
```

To add the `gremlin` log reference, insert the following after the `SPARK_SERVER_LOGBACK_CONF_FILE` reference:

```
<include file="${GREMLIN_SERVER_LOGBACK_CONF_FILE}"/>
```

and the following in the root tag definition after the `SparkWorkerFileAppender`:

```
<appender-ref ref="GremlinServerFileAppender" />
```

Shutting down Studio Gremlin process

The DataStax Enterprise Help Center also provides troubleshooting information.

It is possible for a server-side execution in DataStax Studio to become unresponsive. The symptom of this issue is a never-ending progress spinner in the cell where the command was executed. To fix this issue two steps should be taken.

1. Copy the cell's command to a new cell, delete the hung cell, and restart DataStax Studio.
2. Restart DataStax Enterprise, since DataStax Studio cell's status will most likely still be out of sync with DSE and the issue will not resolve.

This issue will be fixed in DataStax Studio 1.1.0.

Dropping edge property drops edges

The DataStax Enterprise Help Center also provides troubleshooting information.

For existing graph data created prior to DSE 5.0.5, dropping an edge property might drop the edge, depending on the method used to create the edge property.

- If the edge property was created at the same time as the edge label in previous DSE versions, dropping the edge property will drop the edge entirely.
- If the edge property is created separately from the edge label in previous DSE versions, dropping the edge property will not drop the edge.

This issue is fixed in DSE 5.0.5, but data created prior to that version will continue to drop edges in an unexpected manner.

Creating graph with options while unified authentication is enabled fails

The DataStax Enterprise Help Center also provides troubleshooting information.

If unified authentication is enabled, the following graph creation will fail with a `NullPointerException`:

```
system.graph('food3').
 replication("{'class' : 'NetworkTopologyStrategy', 'PSC' : 3 }").
 systemReplication("{'class' : 'NetworkTopologyStrategy', 'PSC' : 3 }").
 option("graph.schema_mode").set("Production").
 create()
```

The workaround until this issue is resolved is to create graphs without options, and set them after creation:

```
// Create graph
system.graph('food3').
 replication("{'class' : 'NetworkTopologyStrategy', 'PSC' : 3 }").
 systemReplication("{'class' : 'NetworkTopologyStrategy', 'PSC' : 3 }")
// Set options for graph created
schema.config().option("graph.schema_mode").set("Production")
```

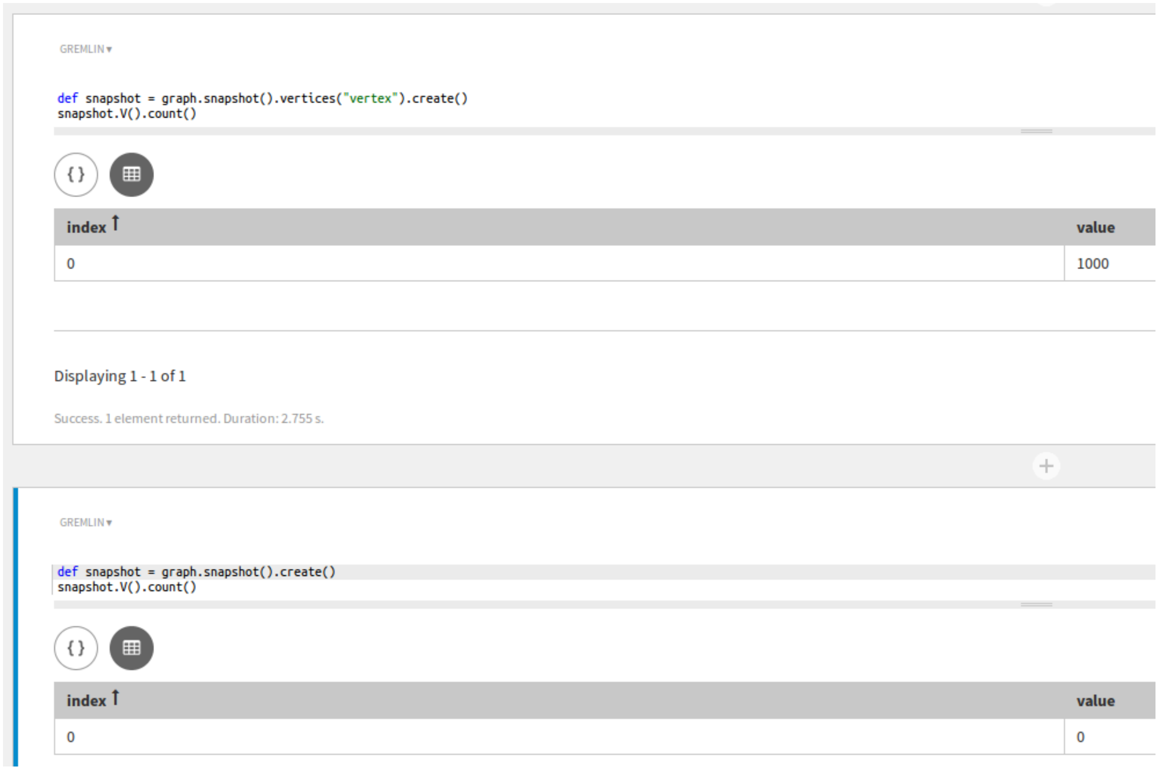
DSP-12174 fixes this issue in DSE 5.0.6 and later.

Vertex label not specified in DSE Graph snapshot

The DataStax Enterprise Help Center also provides troubleshooting information.

If a vertex label is not specified for `graph.snapshot().create()`, then an empty snapshot will be created.

The result can be seen in Studio by looking at the count value for a snapshot:



```
def snapshot = graph.snapshot().vertices("vertex").create()
snapshot.V().count()
```

index	value
0	1000

Displaying 1 - 1 of 1

Success. 1 element returned. Duration: 2.755 s.

```
def snapshot = graph.snapshot().create()
snapshot.V().count()
```

index	value
0	0

This issue is fixed in DSE 5.0.6 and later.

Adding or removing a Spark application, driver, or worker fails

The DataStax Enterprise Help Center also provides troubleshooting information.

Adding or removing a Spark application, driver, or worker fails if it cannot be written to recovery storage.

When the Spark Master first starts it updates its transient state and begins process management tasks, like communicating with workers. Then the Master tries to store changes in the persistence engine. If that task fails, the Master is restarted and its transient state is rebuilt from recovery storage, from the last state that was successfully persisted. To avoid an inconsistent state between these stages, any registration or removal of Spark applications, drivers, or workers will fail until the Master can write the data to recovery storage.

If you encounter a failure, wait until the Master is running and has successfully written to recovery storage before adding or removing applications, drivers, or workers.

DSE Search troubleshooting

The DataStax Enterprise Help Center also provides troubleshooting information.

DSE Search node start

Slow startup on nodes with large encrypted indexes

Slow startup on nodes with large encrypted indexes is resolved in 6.0.0 and later, 5.1.6 and later, and 5.0.12 and later. However, action is required to realize the performance gains.

First upgrade the nodes and then do a full reindex of all encrypted search indexes on each node in your cluster. Plan sufficient time after the upgrade is complete to reindex with deleteAll=true in a rolling fashion. For example:

```
$ dsetool reload_core keyspace_name.table_name distributed=false
 reindex=true deleteAll=true
```

DSE Search nodes fail to start

DSE 5.1.11 and later and DSE 6.0.3 and later refuse to start when required Tomcat files are not present.

When the `tomcat/conf` directory is missing, a WARN message like this appears in the system.log:

```
WARN  [localhost-startStop-2] 2018-07-26 11:55:41,104
 DirectJDKLog.java:182 - Failed to scan [file:/usr/local/
dse/dse-5.1.9/resources/cassandra/lib/apache-cassandra-
thrift-3.11.1.2261.jar] from classloader hierarchy
java.io.FileNotFoundException: /usr/local/dse/dse-5.1.9/resources/
cassandra/lib/apache-cassandra-thrift-3.11.1.2261.jar (No such file
 or directory)
        at java.util.zip.ZipFile.open(Native Method)
 ~[na:1.8.0_181]
        at java.util.zip.ZipFile.<init>(ZipFile.java:225)
 ~[na:1.8.0_181]
        at java.util.zip.ZipFile.<init>(ZipFile.java:155)
 ~[na:1.8.0_181]
        at java.util.jar.JarFile.<init>(JarFile.java:166)
 ~[na:1.8.0_181]
        at java.util.jar.JarFile.<init>(JarFile.java:130)
 ~[na:1.8.0_181]
        at
 org.apache.tomcat.util.scan.JarFileUrlJar.<init>(JarFileUrlJar.java:60)
 ~[tomcat-util-scan-8.0.47.jar:8.0.47]
        at
 org.apache.tomcat.util.scan.JarFactory.newInstance(JarFactory.java:49)
 ~[tomcat-util-scan-8.0.47.jar:8.0.47]
```

```
        at
 org.apache.tomcat.util.scan.StandardJarScanner.process(StandardJarScanner.java:334
 ~[tomcat-util-scan-8.0.47.jar:8.0.47]
        at
 org.apache.tomcat.util.scan.StandardJarScanner.scan(StandardJarScanner.java:284)
 ~[tomcat-util-scan-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.startup.ContextConfig.processJarsForWebFragments(ContextConfig
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.startup.ContextConfig.webConfig(ContextConfig.java:1131)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.startup.ContextConfig.configureStart(ContextConfig.java:783)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.startup.ContextConfig.lifecycleEvent(ContextConfig.java:307)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.util.LifecycleSupport.fireLifecycleEvent(LifecycleSupport.java
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.util.LifecycleBase.fireLifecycleEvent(LifecycleBase.java:90)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.core.StandardContext.startInternal(StandardContext.java:5213)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.util.LifecycleBase.start(LifecycleBase.java:145)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.core.ContainerBase.addChildInternal(ContainerBase.java:753)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at org.apache.catalina.core.ContainerBase.access
$000(ContainerBase.java:131) [tomcat-embed-core-8.0.47.jar:8.0.47]
        at org.apache.catalina.core.ContainerBase
$PrivilegedAddChild.run(ContainerBase.java:153) [tomcat-embed-
core-8.0.47.jar:8.0.47]
        at org.apache.catalina.core.ContainerBase
$PrivilegedAddChild.run(ContainerBase.java:143) [tomcat-embed-
core-8.0.47.jar:8.0.47]
        at java.security.AccessController.doPrivileged(Native
 Method) [na:1.8.0_181]
        at
 org.apache.catalina.core.ContainerBase.addChild(ContainerBase.java:727)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.core.StandardHost.addChild(StandardHost.java:717)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.startup.HostConfig.deployDirectory(HostConfig.java:1129)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
```

```
        at org.apache.catalina.startup.HostConfig
$DeployDirectory.run(HostConfig.java:1871) [tomcat-embed-
core-8.0.47.jar:8.0.47]
        at java.util.concurrent.Executors
$RunnableAdapter.call(Executors.java:511) [na:1.8.0_181]
        at java.util.concurrent.FutureTask.run(FutureTask.java:266)
 [na:1.8.0_181]
        at
 java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
 [na:1.8.0_181]
        at java.util.concurrent.ThreadPoolExecutor
$Worker.run(ThreadPoolExecutor.java:624) [na:1.8.0_181]
        at java.lang.Thread.run(Thread.java:748) [na:1.8.0_181]
```

Followed by reference to the missing locale Tomcat JAR files:

```
WARN  [localhost-startStop-2] 2018-07-26 11:55:41,106
 DirectJDKLog.java:182 - Failed to scan [file:/usr/local/dse/
dse-5.1.9/resources/spark/lib/derbyLocale_cs.jar] from classloader
 hierarchy
java.io.FileNotFoundException: /usr/local/dse/dse-5.1.9/resources/
spark/lib/derbyLocale_cs.jar (No such file or directory)
        at java.util.zip.ZipFile.open(Native Method)
 ~[na:1.8.0_181]
        at java.util.zip.ZipFile.<init>(ZipFile.java:225)
 ~[na:1.8.0_181]
        at java.util.zip.ZipFile.<init>(ZipFile.java:155)
 ~[na:1.8.0_181]
        at java.util.jar.JarFile.<init>(JarFile.java:166)
 ~[na:1.8.0_181]
        at java.util.jar.JarFile.<init>(JarFile.java:130)
 ~[na:1.8.0_181]
        at
 org.apache.tomcat.util.scan.JarFileUrlJar.<init>(JarFileUrlJar.java:60)
 ~[tomcat-util-scan-8.0.47.jar:8.0.47]
        at
 org.apache.tomcat.util.scan.JarFactory.newInstance(JarFactory.java:49)
 ~[tomcat-util-scan-8.0.47.jar:8.0.47]
        at
 org.apache.tomcat.util.scan.StandardJarScanner.process(StandardJarScanner.java:334
 ~[tomcat-util-scan-8.0.47.jar:8.0.47]
        at
 org.apache.tomcat.util.scan.StandardJarScanner.scan(StandardJarScanner.java:284)
 ~[tomcat-util-scan-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.startup.ContextConfig.processJarsForWebFragments(ContextConfig
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.startup.ContextConfig.webConfig(ContextConfig.java:1131)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.startup.ContextConfig.configureStart(ContextConfig.java:783)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
```

```
        at
 org.apache.catalina.startup.ContextConfig.lifecycleEvent(ContextConfig.java:307)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.util.LifecycleSupport.fireLifecycleEvent(LifecycleSupport.java
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.util.LifecycleBase.fireLifecycleEvent(LifecycleBase.java:90)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.core.StandardContext.startInternal(StandardContext.java:5213)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.util.LifecycleBase.start(LifecycleBase.java:145)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.core.ContainerBase.addChildInternal(ContainerBase.java:753)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at org.apache.catalina.core.ContainerBase.access
$000(ContainerBase.java:131) [tomcat-embed-core-8.0.47.jar:8.0.47]
        at org.apache.catalina.core.ContainerBase
$PrivilegedAddChild.run(ContainerBase.java:153) [tomcat-embed-
core-8.0.47.jar:8.0.47]
        at org.apache.catalina.core.ContainerBase
$PrivilegedAddChild.run(ContainerBase.java:143) [tomcat-embed-
core-8.0.47.jar:8.0.47]
        at java.security.AccessController.doPrivileged(Native
 Method) [na:1.8.0_181]
        at
 org.apache.catalina.core.ContainerBase.addChild(ContainerBase.java:727)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.core.StandardHost.addChild(StandardHost.java:717)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at
 org.apache.catalina.startup.HostConfig.deployDirectory(HostConfig.java:1129)
 [tomcat-embed-core-8.0.47.jar:8.0.47]
        at org.apache.catalina.startup.HostConfig
$DeployDirectory.run(HostConfig.java:1871) [tomcat-embed-
core-8.0.47.jar:8.0.47]
        at java.util.concurrent.Executors
$RunnableAdapter.call(Executors.java:511) [na:1.8.0_181]
        at java.util.concurrent.FutureTask.run(FutureTask.java:266)
 [na:1.8.0_181]
        at
 java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
 [na:1.8.0_181]
        at java.util.concurrent.ThreadPoolExecutor
$Worker.run(ThreadPoolExecutor.java:624) [na:1.8.0_181]
        at java.lang.Thread.run(Thread.java:748) [na:1.8.0_181]
```

To resolve:

- Ensure that the `catalina.properties` and `context.xml` files are present in the Tomcat `conf` dir. DSE will not start after upgrade if these files are missing.

  The default location of the Tomcat `conf` directory depends on the type of installation:

  # Package installations: `/etc/dse/tomcat/conf`
  # Tarball installations: `*installation_location*/resources/tomcat/conf`

Handling inconsistencies in query results

The DataStax Enterprise Help Center also provides troubleshooting information.

To troubleshoot inconsistencies in query results, consider session stickiness, subrange node repair, and follow best practices for soft commit points on different replica nodes.

DSE Search implements an efficient, highly available distributed search algorithm on top of the database, which tries to select the minimum number of replica nodes required to cover all token ranges, and also avoid hot spots. Consequently, due to the eventually consistent nature of the database, some replica nodes might not have received or might not have indexed the latest updates yet. This situation might cause DSE Search to return inconsistent results (different numFound counts) between queries due to different replica node selections. This behavior is intrinsic to how highly available distributed systems work, as described in the ACM article, "Eventually Consistent" by Werner Vogels. Most of the time, eventual consistency is not an issue, yet DSE Search implements *session stickiness* to guarantee that consecutive queries will hit the same set of nodes on a healthy, stable cluster, to provide monotonic results. Session stickiness works by adding a *session seed* to request parameters as follows:

```
shard.shuffling.strategy=SEED
shard.shuffling.seed=session_id
```

In the event of unstable clusters with missed updates due to failures or network partitions, consistent results can be achieved by repairing nodes using the DSE OpsCenter Repair Service.

Finally, another minor source of inconsistencies is caused by different soft commit points on different replica nodes: A given item might be indexed and committed on a given node, but not yet on its replica. This situation is primarily a function of the load on each node. Implement the following best practices:

- Evenly balance read/write load between nodes
- Properly tune soft commit time and async indexing concurrency
- Configure back pressure in the dse.yaml file

For information about multi-threaded asynchronous indexing that uses a back pressure mechanism, see Configuring and tuning indexing performance.

To maximize insert throughput, DSE Search buffers insert requests from the database so that application insert requests can be acknowledged as quickly as possible. However, if too many requests accumulate in the buffer (a configurable setting), DSE Search pauses or blocks incoming requests until DSE Search catches up with the buffered requests. In extreme cases, that pause causes a timeout to the application.

Tracing Solr HTTP requests

The DataStax Enterprise Help Center also provides troubleshooting information.

For debugging and troubleshooting queries, you can trace Solr HTTP requests in one of the following ways:

- Enable probabilistic tracing.
- Pass an explicit cassandra.trace=true request parameter in the HTTP query.

After running the example of using a join query, you can trace the join query by adding the cassandra.trace parameter to the HTTP request:

```
http://localhost:8983/solr/internet.songs/select/?
q={!join+from=song+to=id+fromIndex=internet.lyrics
+force=true}words:love&indent=true&wt=json&cassandra.trace=true
```

The Solr response includes a cassandra.trace.session value, the unique session id of the tracing session for the request:

```
{
  "cassandra.trace.session":"3e503490-bdb9-11e3-860f-73ded3cb6170",
  "responseHeader":{
    "status":0,
    "QTime":1,
    "params":{
      "indent":"true",
      "q":"{!join from=song to=id fromIndex=internet.lyrics
 force=true}words:love",
      "wt":"json",
      "cassandra.trace":"true"}},
  "response":{"numFound":2,"start":0,"docs":[
      {
        "id":"8a172618-b121-4136-bb10-f665cfc469eb",
        "title":"Internet Love Song",
        "artist":"John Cedrick"},
      {
        "id":"a3e64f8f-bd44-4f28-b8d9-6938726e34d4",
        "title":"Dangerous",
        "artist":"Big Data"}]
}}
```

To see the information from the trace, query the system_traces.events, using the session id to filter the output.

```
cqlsh> select * from system_traces.events where session_id = 3e503490-
bdb9-11e3-860f-73ded3cb6170;

 session_id  | activity
               | source_elapsed
-------------
+---------------------------------------------------------------------------
+----------------
 3e503490... |          Parsing SELECT * from "internet"."songs" WHERE
 "id" = 8a172618...|           2607
 3e503490... |
 Preparing statement |           3943
 3e503490... |                                Executing single-partition
 query on songs |           4246
 3e503490... |                                            Acquiring
 sstable references |           4261
 3e503490... |                                             Merging
 memtable tombstones |           4305
 3e503490... |                                             Key cache hit
 for sstable 1 |           4388
 3e503490... |                         Seeking to partition indexed
 section in data file |           4399
 3e503490... | Skipped 0/1 non-slice-intersecting sstables, included 0 due
 to tombstones |           4873
 3e503490... |                               Merging data from memtables
 and 1 sstables |           4954
 3e503490... |                                       Read 1 live and 0
 tombstoned cells |           5162
 3e503490... |        Parsing SELECT * from "internet"."songs" WHERE "id"
 = a3e64f8f...  |           6160
 3e503490... |
 Preparing statement |           7424
 . . .
```

For example purposes, the event_id, node IP address, and thread id have been deleted from this output to fit on the page.

In the case of distributed queries over several nodes, the same tracing session id is used on all nodes, which makes it possible to correlate database operations on all the nodes taking part in the distributed query.

Using the ShardRouter MBean

The DataStax Enterprise Help Center also provides troubleshooting information.

Use the com.datastax.bdp:type=ShardRouter MBean to retrieve information and update the list of endpoints.

The ShardRouter MBean, not present in open source Solr, provides information about how DSE search routes queries. JMX MBeans can be accessed by connecting to the (default)

JMX port 7199 on any DataStax Enterprise node using a JMX application like JConsole. The following the attributes and operations are available in this MBean:

- getShardSelectionStrategy(String core) retrieves the name of the shard selection strategy used for the given search core.
- getEndpoints(String core) retrieves the list of endpoints that can be queried for the given search core.
- getEndpointLoad(String core) retrieves the list of endpoints with related query load for the given search core. The load is computed as a 1-minute, 5-minutes and 15-minutes exponentially weighted moving average, based on the number of queries received by the given node.
- refreshEndpoints() manually refreshes the list of endpoints to be used for querying Solr cores.

Using Solr MBeans

The DataStax Enterprise Help Center also provides troubleshooting information.

DataStax Enterprise provides enhanced visibility into native memory allocation through the solr/NativeAllocatorStats MBean, exposing the following information:

- enabled: if native memory is enabled or not.
- debug: if debug mode is enabled or not.
- numAlloc: number of native objects allocations.
- numFree: number of freed native objects.
- activeAllocatedMemoryInBytes: allocated native memory currently in use.
- totalAllocatedMemoryInBytes: total allocated native memory over time.
- totalFreedMemoryInBytes: total freed native memory over time.

The solr/NativeTrackerStats MBean provides information about the tracked native objects and related threads that allocated them:

- registeredThreads: number of threads currently registered and actively tracking (allocating) native objects.
- trackedObjects: number of currently tracked (allocated and not freed) native objects.
- handedOffObjects: number of currently handed off (allocated and stored for later reuse) native objects.

Troubleshooting security

The DataStax Enterprise Help Center also provides troubleshooting information.

DSE fails to start

The DataStax Enterprise Help Center also provides troubleshooting information.

DSE verifies authentication services during start up. When a service is not available, such as Kerberos or LDAP, that has been set as a default_scheme or other_schemes in the authentication_options, the following error appears in the log:

```
ERROR [main] 2017-03-08 17:03:33,482  DseModule.java:97 - {}. Exiting...

com.google.inject.CreationException: Unable to create injector, see the
 following errors:

1) An exception was caught and reported. Message: The dse service keytab
 at this location resources/dse/conf/dse.keytab either doesn't exist or
 cannot be read by the dse service
at com.datastax.bdp.DseModule.configure(Unknown Source)

1 error
```

To allow DSE to properly start either ensure that the service is available or remove it from the configuration.

SSL certificate doesn't match

The Common Name (CN) that is used to generate the SSL certificate must match the DNS resolvable host name. Mismatches between the CN and node hostname cause an exception and the connection is refused. With dsetool and other tools that issue commands to the cluster, error messages indicate that the environment is not configured correctly. For example:

```
dsetool -h 10.236.136.55 reload_core keyspace_name.table_name
 deleteAll=true reindex=true distributed=false
...
javax.net.ssl.SSLException: Certificate for <node35.foo.com> doesn't match
 any of the subject alternative names: clustercert.foo.com
...
```

This message shows that `10.236.136.55` is being resolved to `node35.foo.com`, then node `node35.foo.com` is being asked for it's certificate which is a generic certificate issued with a CN of `clustercert.foo.com`. For security reasons, SSL verifies that `node35.foo.com` and `clustercert.foo.com` match. If they don't match, a certificate mismatch error occurs. Do not use a generic certificate across multiple nodes, because each node has a different name that won't match. Do not copy a certificate that is issued to `node35.foo.com` over to another node.

Nodes must be configured with correct names that match the certificate CN. You can use a wildcard in named certificates, like `*` in `CN=*.foo.com`, or any other matching mechanism allowed by SSL standards. All others configurations where names mismatch will result in an error.

When testing connections with other tools, enable them with secure settings. Avoid testing with insecure settings that do not require name matching. For example, `curl --insecure`. These insecure settings do not identify certificate mismatches and are not supported in DSE tools.

SSL exceptions occur on start up or no connections

Follow these steps to troubleshoot SSL connections when exceptions occur on start up, or no connections to the database can be established.

- To enable debugging, add the following option to cassandra-env.sh:

  ```
  -Djavax.net.debug=ssl
  ```

  Detail startup and connection messages are printed to STDOUT, including SSL handshake errors. See Debugging SSL/TLS Connections for message details.

- Verify SSL encryption messages in `/var/log/cassandra/system.log`.
  - # SSL starts properly:
    - # For client-to-server connections:

      ```
      INFO  [main] 2017-06-15 21:50:41,928  Server.java:145 -
       Enabling encrypted CQL connections between client and server
      ```

    - # For node-to-node messaging:

      ```
      INFO  [main] 2017-06-15 21:50:23,037
       MessagingService.java:702 - Starting Encrypted Messaging
       Service on SSL port 7001
      ```

  - # SSL fails to start:
    - # Truststore or keystore file not found:

      ```
      Caused by:
       org.apache.cassandra.exceptions.ConfigurationException:
       Failed to initialize SSL

      ...

      Caused by: java.io.FileNotFoundException: resources/dse/
      conf/.truststore (No such file or directory)
      ```

Note: Example shows when the default setting was not changed for node-to-node.

# Truststore or keystore password is invalid:

```
Caused by: java.io.IOException: Error creating the
 initializing the SSL Context

 at
 org.apache.cassandra.security.SSLFactory.createSSLContext(SSLFactory.java:201)

 at
 com.datastax.bdp.node.transport.SSLOptions.getDefault(SSLOptions.java:82)

....

Caused by: java.io.IOException: Keystore was tampered with, or
 password was incorrect
```

Note: Example show when password of keystore for node-to-node is incorrect.

Connection errors with cqlsh and other DSE tools

If `authentication_options` are enabled (`true`) in the dse.yaml, credentials are required to launch cqlsh and other tools:

```
cqlsh -u username -p password
```

If a client connection is attempted without permissions, the following error occurs:

```
Connection error: ('Unable to connect to any servers', {'127.0.0.1':
 error(111, "Tried connecting to [('127.0.0.1', 9042)]. Last error:
 Connection refused")})
```

See Providing credentials with cqlsh and SSL certificate doesn't match *(page 39)*.

Troubleshooting DataStax Studio

The DataStax Enterprise Help Center also provides troubleshooting information.

Studio will not start with wrong Java version

Studio checks the JAVA_HOME environment variable at startup. To ensure that Studio starts with the correct version of Java, make sure the JAVA_HOME environment variable points to the installation directory with the supported version of Java. See Supported platforms. Note that even when the `java -version` command shows a supported Java version, it is possible that JAVA_HOME environment variable is not set to the correct Java installation directory.

NOT ENABLED error: AlwaysOn SQL cell cannot be executed

The AlwaysOn SQL cell cannot be executed when the connection is to a DSE cluster that is not correctly configured for the AlwaysOn SQL service.

In order to run AlwaysOn SQL, you must have:

- A running datacenter with DSE Analytics nodes enabled.
- Enabled AlwaysOn SQL on every Analytics node in the datacenter.
- Modify the eplication factor for all Analytics nodes, if necessary.
- Set the native_transport_address in cassandra.yaml to an IP address that is accessible by the AlwaysOn SQL clients. This address depends on your network topology and deployment scenario.
- Configured AlwaysOn SQL for security, if authentication is enabled.

STOPPED error: AlwaysOn SQL cell cannot be executed

The AlwaysOn SQL cell cannot be executed when the AlwaysOn SQL service is stopped on the DSE node. See Checking the status of AlwaysOn SQL.

HTTP protocol

DataStax Studio does not run with `HTTPS` everywhere. Preface the URL used to run DataStax Studio with `HTTP` and not `HTTPS`.

Studio won't accept connections from external addresses

By default, Studio binds itself to the IP address `127.0.0.1` (or `localhost`). If running in the cloud, this IP address prevents Studio from accepting connections from external addresses. If you have already installed Studio in the cloud using `localhost`, change the httpBindAddress value in the `configuration.yaml` Studio configuration file to the IP address of the host it is running on.

> **Important:** Changing the httpBindAddress setting from the default (localhost) can pose a security risk as users on external machines can gain access to notebooks and the DSE clusters those notebooks are connected to. Studio is designed to be used as a desktop application. Distributed deployment introduces potential security risks.

Unable to connect Studio to a secured DSE cluster

If you are unable to connect Studio when client-to-node encryption enabled on a DSE cluster, the Studio log has this entry:

```
Cannot support TLS_RSA_WITH_AES_256_CBC_SHA with currently installed
 providers
```

Java Cryptography Extension (JCE) Unlimited Strength Policy files is required to ensure support for all encryption algorithms. See Using SSL connections in DataStax Studio.

Content assist not working

On MacOS 10.12, a network configuration issue when running Studio causes content assist functionality to become slow or unresponsive. For troubleshooting steps to resolve this networking issue, see this Stack Overflow post.

Notebook needs troubleshooting

When you export a notebook, you can specify actions for the notebook. As appropriate, select to include cell code and results or cell code only. For either of these actions, you can also select to include diagnostics. Exporting a notebook with diagnostics might include sensitive Information. Credentials are not included in the export.

Studio UI behaves strangely after upgrade

If you leave an old Studio session open in a browser, using an older version of Studio, and then upgrade to a new version, you can experience strange behavior. This is caused by caching in the browser. You can solve this problem by force-loading the `index.html` page. For example, if using the Chrome browser, you can navigate to the top-level `index.html` page and click on the refresh button while holding down the control key.

Starting Studio on Windows 10

While running Studio on the Microsoft Windows 10 operating system, out-of-date libraries can prevent system-specific JNI libraries from being loaded correctly. The result is the following stack trace:

```
[Studio-akka.actor.default-dispatcher-4] ERROR akka.actor.OneForOneStrategy
 ID:  TS: - head of empty list
java.util.NoSuchElementException: head of empty list
    at scala.collection.immutable.Nil$.head(List.scala:420) ~[scala-
library-2.11.8.jar:?]
    at scala.collection.immutable.Nil$.head(List.scala:417) ~[scala-
library-2.11.8.jar:?]
    at akka.actor.ActorCell.receiveMessage(ActorCell.scala:526) ~[akka-
actor_2.11-2.4.17.jar:?]
    at akka.actor.ActorCell.invoke(ActorCell.scala:495) ~[akka-
actor_2.11-2.4.17.jar:?]
    at akka.dispatch.Mailbox.processMailbox(Mailbox.scala:257) ~[akka-
actor_2.11-2.4.17.jar:?]
```

```
    at akka.dispatch.Mailbox.run(Mailbox.scala:224) ~[akka-
actor_2.11-2.4.17.jar:?]
    at
 java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149
```

You can fix this problem by updating the Microsoft Visual C++ 2010 Redistributable Package libraries in question:

- 32-bit version
- 64-bit version

Starting Studio on Windows 7

While running Studio on the Microsoft Windows 7 operating system, out-of-date libraries can prevent system-specific JNI libraries from being loaded correctly. The result is the following stack trace:

```
leveldbjni-64-
1-6036870325998847314.8: Can't find dependent libraries]
        at org.fusesource.hawtjni.runtime.Library.doLoad(Library.java:182)
        at org.fusesource.hawtjni.runtime.Library.load(Library.java:140)
        at
 org.fusesource.leveldbjni.JniDBFactory.<clinit>(JniDBFactory.java:48)
        at
 com.rbmhtechnology.eventuate.log.leveldb.LeveldbEventLog.<init>(Level
 dbEventLog.scala:97)
        at com.rbmhtechnology.eventuate.log.leveldb.LeveldbEventLog$
$anonfun$7.apply(LeveldbEventLog.scala:358)
        at com.rbmhtechnology.eventuate.log.leveldb.LeveldbEventLog$
$anonfun$7.apply(LeveldbEventLog.scala:358)
        at
 akka.actor.TypedCreatorFunctionConsumer.produce(IndirectActorProducer.scala:87)
```

You can fix this problem by updating the Microsoft Visual C++ 2010 Redistributable Package libraries in question:

You can fix this problem by updating the Microsoft Visual C++ 2010 Redistributable Package libraries in question:

- 32-bit version
- 64-bit version

Other troubleshooting

The DataStax Enterprise Help Center also provides troubleshooting information.

View of ring differs between some nodes

The DataStax Enterprise Help Center also provides troubleshooting information.

Indicates that the ring is in a bad state.

This situation can happen when not using virtual nodes (vnodes) and token conflicts exist. You might see this when bootstrapping two nodes simultaneously with automatic token selection. Unfortunately, the only way to resolve this is to do a full cluster restart. A rolling restart is insufficient since gossip from nodes with the bad state will repopulate it on newly booted nodes.

SSTables imported using sstableloader are missing rows

The DataStax Enterprise Help Center also provides troubleshooting information.

The source SSTables used by `sstableloader` (bulk loader) may not have included all the table data.

The `sstableloader` streams data from SSTables on disk to a cluster.

Before importing existing SSTables, run nodetool flush on each source node to assure that any data in memtables is written to the SSTables on disk.

For more information and instructions, see sstableloader.

Prepared statements discarded

The DataStax Enterprise Help Center also provides troubleshooting information.

If you repeatedly see messages about prepared statements being discarded because cache limit have been reached:

1. Investigate the root cause of these messages and check whether prepared statements are used correctly by using bind markers for variable parts.

2. Only change the default value of prepared_statements_cache_size_mb when there are more prepared statements than fit in the cache. In most cases it is not necessary to change this value.

Purging gossip state on a node

The DataStax Enterprise Help Center also provides troubleshooting information.

Gossip information is persisted locally by each node to use immediately on node restart without having to wait for gossip communications.

In the unlikely event you need to correct a problem in the gossip state:

1. Use the nodetool assassinate to shut down the problem node.

   This takes approximately 35 seconds to complete, so wait for confirmation that the node is deleted.

2. If this method doesn't solve the problem, stop your client application from sending writes to the cluster.

3. Take the entire cluster offline:

   a. Drain each node.

      ```
      $ nodetool options drain
      ```

   b. Stop each node.

4. Clear the data from the `peers` directory, remove all directories in the peers-*UUID* directory, where *UUID* is the particular directory that corresponds to the appropriate node:

   ```
   $ sudo rm -r /var/lib/cassandra/data/system/peers-UUID/*
   ```

   Caution:

   Use caution when performing this step. The action clears internal system data from the database and may cause application outage without careful execution and validation of the results. To validate the results, run the following query individually on each node to confirm that all of the nodes are able to see all other nodes.

   ```
   select * from system.peers;
   ```

5. Clear the gossip state when the node starts:

- For tarball or Installer No-Services installations, use a command line option or edit the `cassandra-env.sh`. To use the command line:

  ```
  $ installation_location/bin/cassandra -
  Dcassandra.load_ring_state=false
  ```

- For package or Installer Services installations or if you are not using the command line option above *(page 46)*, add the following line to the cassandra-env.sh file:

  ```
  JVM_OPTS="$JVM_OPTS -Dcassandra.load_ring_state=false"
  ```

6. Bring the cluster online one node at a time, starting with the seed nodes.

See Starting and stopping DataStax Enterprise.

What's next:

Remove the line you added in the `cassandra-env.sh` file.

Recovering expired data caused by TTL year 2038 problem

Prior to DataStax Enterprise version 5.1.7 in the 5.1.X series and 5.0.12 in the 5.0.X series, there was no protection against INSERTS if the TTL expiration timestamp was after the maximum date that the storage engine could represent (`2038-01-19T03:14:06+00:00`). Before 5.1.7 or 5.0.12, if an expiration timestamps with a later date was inserted, the date calculation overflowed causing the data to expire immediately. Records expired by overflow are not queryable and are permanently removed after a compaction. This issue occurs only for INSERTs that have a long TTL value that is close to the maximum 630720000 seconds (20 years). The earliest possible date overflow for expiration timestamps is `2018-01-19T03:14:06+00:00`. As time progresses, the maximum supported TTL gradually reduces as the date `2038-01-19T03:14:06+00:00` approaches.

To recover data with overflowed timestamps from SSTables that were backed up or that did not go through compaction, use the `--reinsert-overflowed-ttl` option, because tombstones might have been generated with the original timestamp.

> **Tip:** To find out if an SSTable has an entry with overflowed expiration, inspect it with the sstablemetadata tool and look for a negative `min local deletion time` field. Back up SSTables in this condition immediately, as they are subject to data loss during compaction.

Use only one of the following options.

Offline scrub

For offline scrub *(page 48)*, use the `sstablescrub` command with the `--reinsert-overflowed-ttl` parameter to recover data from a backed up SSTable.

DSE version	Link
DSE 6.7	sstablescrub --reinsert-overflowed-ttl
DSE 6.0	sstablescrub --reinsert-overflowed-ttl
DSE 5.1.7 and later	sstablescrub --reinsert-overflowed-ttl
DSE 5.0.12 and later	sstablescrub --reinsert-overflowed-ttl
DSE 4.8.16	sstablescrub --reinsert-overflowed-ttl

Online scrub

For online scrub *(page 49)*, use the `nodetool scrub` command with the `--reinsert-overflowed-ttl` parameter to recover data from a table that has not gone through compaction.

DSE version	Link
DSE 6.7	nodetool --reinsert-overflowed-ttl
DSE 6.0	nodetool --reinsert-overflowed-ttl
DSE 5.1.7 and later	nodetool --reinsert-overflowed-ttl
DSE 5.0.12 and later	nodetool --reinsert-overflowed-ttl
DSE 4.8.16	nodetool --reinsert-overflowed-ttl

- Use the offline scrub option to recover data from a backed up SSTable:

    1. Clone the data directory tree to another location. Keep only the folders and the contents of the system tables.
    2. Clone the configuration directory to another location. Set the data_file_directories property to the cloned data directory in the cloned cassandra.yaml.
    3. Copy the affected SSTables to the cloned data location of the affected table.
    4. Set the environment variable `CASSANDRA_CONF=`*`cloned_configuration_directory`*.
    5. Run the following command to update the table.

    ```
    $ sstablescrub --reinsert-overflowed-
    ttl keyspace_name table_name
    ```

6. After the scrub is completed, copy the resulting SSTables to the original data directory.
7. Use nodetool refresh on a live node to load the recovered SSTables.

- Use the online scrub option to recover data from a table that has not gone through compaction:

  1. Disable compaction on the node.

     ```
     $ nodetool disableautocompaction
     ```

     Warning: This step is crucial. The data might be removed permanently during compaction.

  2. Copy the SSTables containing entries with overflowed expiration time to the data directory.
  3. Load the SSTables.

     ```
     $ nodetool refresh
     ```

  4. Run the scrub command with reinsert overflow option on the tables.

     ```
     $ nodetool scrub --reinsert-overflowed-
     ttl keyspace_name table_name
     ```

  5. For indexed tables, use the dsetool reload_core command on a search node to load and reindex the reinserted values:

     ```
     $ dsetool reload_core reindex=true keyspace_name.table_name
     ```

  6. Re-enable compactions after verifying that scrub recovered the missing entries.