IV10381 – Out of memory during data synch if there are many administrators for Admin domains

This issue was originally reported in ITIM 4.6 and was fixed as part of APAR IV09100.

This APAR addresses a memory issue reported during data synchronization when customers have many administrators for one or more admin domains.

As an example, consider the following organization (container) structure:

Organization

|--- Client Admin Domains – Have 1000 Administrators

|--- Client Admin Domain 1 – Have 3 Administrators

|--- Client Admin Domain 2 - Have 3 Administrators

|--- Client Admin Domain 3 - Have 3 Administrators

|--- …….

|--- …….

|--- Client Admin Domain 50000 - Have 3 Administrators

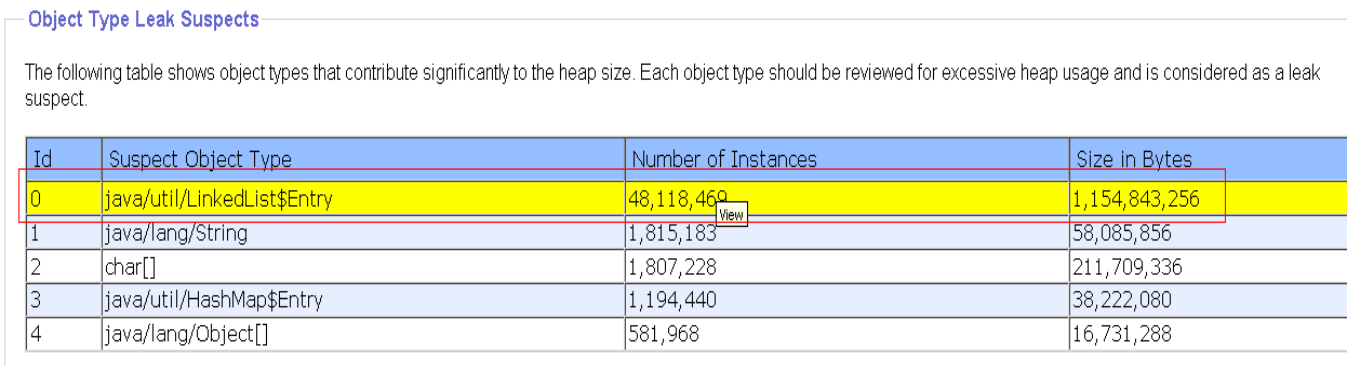
As per the design, administrators of the parent container are also administrators of child containers. So each child admin domain from 1 to 50000 will have 1003 administrators. This administrator detail is stored in memory in a 'HashMap' during data synch process.

The data stored in 'HashMap' has the following format:

(Key - DistinguishedName of container, Value – LinkedList of SystemUser DNs of the administrators)

Since there are around 50000 admin domains (child containers), each having approximately 1000 administrators, the number of LinkedList objects will be approximately 50000 and LinkedList$Entry instances will be 50,000,000 (50,000 containers * 1000 administrator entries for each container). Thus the amount of memory used will be very large and results in OOM.

The following numerical values from the heap dump analysis will give us a general idea about the amount of memory used to store the data:



Object Type Leak Suspects

The following table shows object types that contribute significantly to the heap size. Each object type should be reviewed for excessive heap usage and is considered as a leak suspect.

Id	Suspect Object Type	Number of Instances	Size in Bytes
0	java/util/LinkedList$Entry	48,118,469	1,154,843,256
1	java/lang/String	1,815,183	58,085,856
2	char[]	1,807,228	211,709,336
3	java/util/HashMap$Entry	1,194,440	38,222,080
4	java/lang/Object[]	581,968	16,731,288

From the above table we can see that there are 48,118,469 instances of LinkedList$Entry that occupy 1,154,843,256 bytes of memory.

Each LinkedList$Entry instance will take approximately 24 bytes of memory (1,154,843,256/ 48,118,469). As per the container structure referred above, each LinkedList object will have around 1000 LinkedList$Entry objects. So the memory used up by each LinkedList object will be approximately 24000 bytes (24 bytes of each LinkedList$Entry * 1000 such entries) = 23 KB to store 1000 LinkedList$Entry in one LinkedList object.
Since there are approximately 50k containers, the approximate memory required to store all the data will be 1200000000 bytes (50000 containers * 24000 bytes used for storing administrator details of each container) = 1.11 GB.

Note: The above heap dump analysis result is from a 4.6 set up as the issue was reported by customer for ITIM 4.6.

As part of fix for APAR IV10381, two new properties (adminCacheSize and supervisorCacheSize) have been introduced to limit the size of the HashMap. The property, adminCacheSize, specifies the maximum number of admin domain containers whose administrator data will be stored in the cache (HashMap). The property, supervisorCacheSize, specifies the maximum number of OU/Location containers whose supervisor data will be stored in the cache (HashMap).
In 5.x version, HashSet is used instead of LinkedList.

Hence, the data stored in 'HashMap' have the following format:

(Key - DistinguishedName of container, Value – HashSet of SystemUser DNs of the administrators)

Considering the same example as above, each entry in HashSet takes 32 bytes of memory. So the memory occupied by 1000 SystemUser DNs will be approximately 32000 bytes = 31 KB. 31 KB of memory will be required to store SystemUser information of each container. Since there are 50000 containers, total memory required will be approximately 1600000000 bytes (50000 containers * 32000 bytes used for storing administrators details of each container) = 1.49 GB.

The adminCacheSize value can be set to 8000, so that the memory used will be 256000000 bytes (8000 containers * 32000 bytes used to store administrators of each container) = 244 MB approximately.

So to generalize, the customer basically needs to have idea about number of administrators for any container. (Note administrators of a parent container are also administrators of child containers).

The following things should be kept in mind:

- Number of administrators of the parent container = Number of elements stored in the HashMap for parent container. Each element is an object of HashMap$Entry instance.
- Memory used by HashMap$Entry instances to store administrator information for a single container = Number of administrators of a single container * 32 bytes
- If there are x number of child containers for this particular parent container, then the number of HashMap$Entry instances will be x * number of administrators of the parent container.
- Memory used = x containers * number of administrators of the parent container * 32 bytes

Depending on the above memory usage value, the number of container details to be stored in memory should be decided, so that the memory used will not be high. In an ideal scenario one can limit the memory occupied by admin domain cache to 256 MB.

Use the following formula to calculate the appropriate value for the adminCacheSize.

Value of adminCacheSize property = (max memory limit * 1024 * 1024) bytes / (average number of administrators for each container * 32 bytes)

Example:
If the above container structure scenario is considered, then the value of adminCacheSize property will be calculated as follows:
Value of adminCacheSize = (256*1024*1024) / (1000*32) = 8388.608 = 8000 approximately (rounded figure). This will limit the amount of memory used by the admin domain admin cache to 256 MB.

Generally customers will not have such large number of administrators, so the default value of -1 will be appropriate for them. This default setting will store all the administrator details in memory.

If any customer set up has a large number of administrators in the parent container and if they also have many child containers for the parent container, then they must confirm whether they are facing memory issues during data synchronization. The next step will be to analyze the heap dumps and if the leak suspect is the HashMap storing HashSet objects, then they can configure the values of adminCacheSize and supervisorCacheSize by performing the calculation as explained above.

Note: A similar calculation will also be applicable for the property, supervisorCacheSize. Supervisor is applicable only for OU and locations. Admin Domains and BPOs do not have supervisors. This should be considered while deciding the number of containers present in the Org structure.