Shutdown tomcat server node in case of memory depletion - Avoiding Tomcat Out of memory

Author : admin

Out Of Memory Errors, or OOMEs, are one of the most common problems faced by Apache Tomcat users. **Tomcat cluster behind Apache unreachable (causing customer downtimes)**. OOME errors occur on ***production servers that are experiencing an unusually high spike of traffic***.

Out of memory errors are usually a problem of application and not of Tomcat server. OMEs have become such a persistent topic of discussion in the *Apache Tomcat community* cause its so difficult to trace to their root cause. Usually *'incorrect' web app code causing Tomcat to run out of memory* is usually technically correct.

Most common reasons for Out of Memory errors in application code are:

- **the heap size being too small**
- **running out of file descriptors**
- **more open threads than the host OS allows**
- **code with high amounts of recursion**
- **code that loads a very large file into memory**
- **code that retaining references to objects or classloaders**
- **a large number of web apps and a small PermGen**

The following java option ***-XX:OnOutOfMemoryError=*** could be added to any of tomcat java application servers in **setenv.sh** in **JAVA_OPTS=** variable in case of regular **Out of Memory errors occur** making an application unstable.

-XX:OnOutOfMemoryError=

Where *path_to tomcat_shutdown_script.sh* > is shutdown script(which performs kill if normal shutdown fails) for the tomcat instance.

With this setup *if any tomcat instance run out of memory it will be shutdown (shutdown script invoked)* - as result the *Apache proxy infront of Tomcats* **should not pass any further requests to this instance** and application will visualize / work properly for end customers.

Usually a **tomcat_shutdown_script.sh** to invoke in case of OOM would initiate a *Tomcat server restart* something like:

```
  for i in `ps -ef |grep tomcat |grep /my_path_to_my_instance | awk '{print $2}'`
do
kill -9 "$i"
#path and script to start tomcat
done
```

To prevent blank pages returned to customer because of shutdown_script.sh starting stopping Tomcat you can set in Reverse Apache Proxy something like:

```
   BalancerMember ajp://10.16.166.48:11010/ route=delivery1 timeout=30 retry=1
   BalancerMember ajp://10.16.166.70:11010/ route=delivery2 timeout=30 retry=1
```

Where in above example I assume, there are only two tomcat nodes, for more just add respective ones.

Note that if the deployed application along all servers is having some code making it crash all tomcat nodes can get shutdown all time and you can get in a client havoc :)