Thread Dump - Intelligence Report

File: *high-cpu-tdump.out*

We have found few suspect(s). It may or may not be a problem:

To learn about the suspects, see the ⚠ icons in this report.

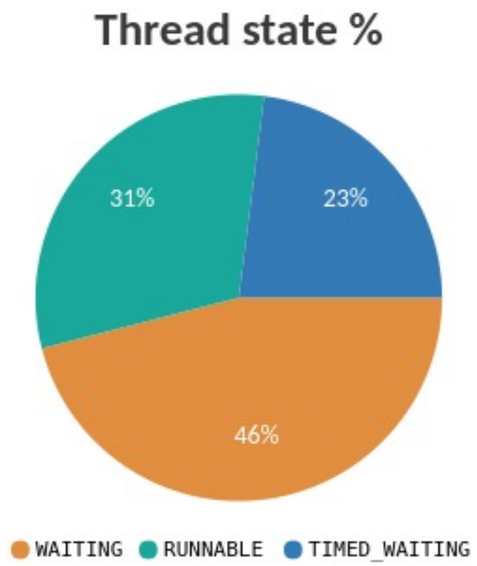
Thread Count Summary

(To learn about different thread states through real-life example, check out this video tutorial)

160
WAITING
View Details

109
RUNNABLE
View Details

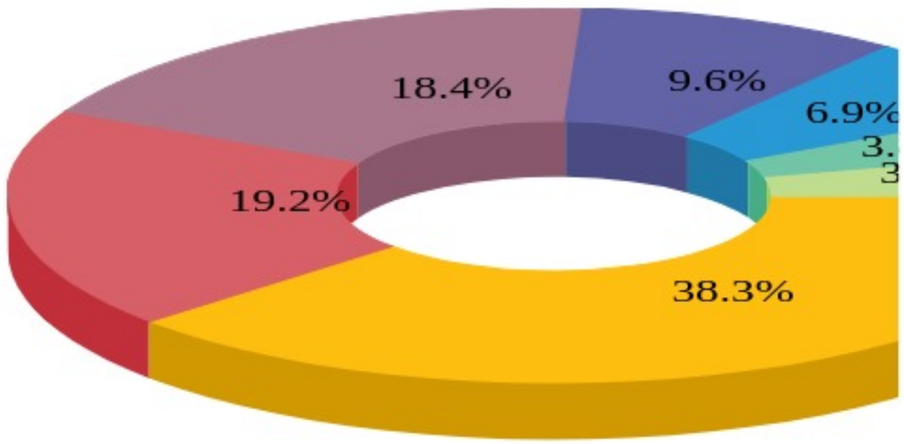
81
TIMED_WAITING
View Details



Total Threads count: **350**

Thread Group

(Threads with similar names are grouped in this section)



Legends	Thread Group	Count	States
●	EE-ManagedThreadFactory-er	**100** threads	:16 WAITING:74 RUNNABLE:10
●	EE-ManagedThreadFactory-er	**50** threads	TIMED_WAITING :50
●	default I/O	**48** threads	RUNNABLE:48
●	Weld Thread Pool -	**25** threads	WAITING:25
●	GC task thread	**18** threads	RUNNABLE:18

Show all thread groups >>

Daemon vs non-Daemon

(daemon and non-daemon(i.e. user) threads count is shown in this section)

210

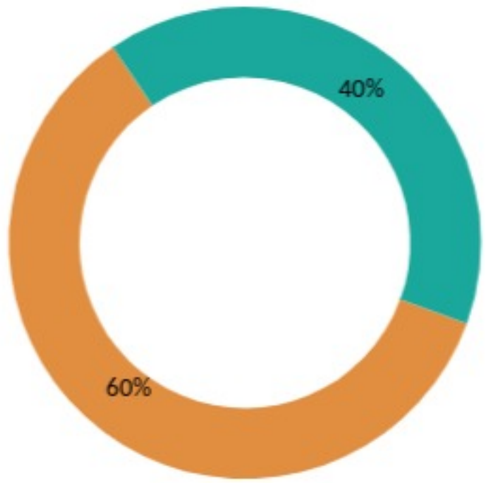
DAEMON

View Details

140

NON-DAEMON

View Details

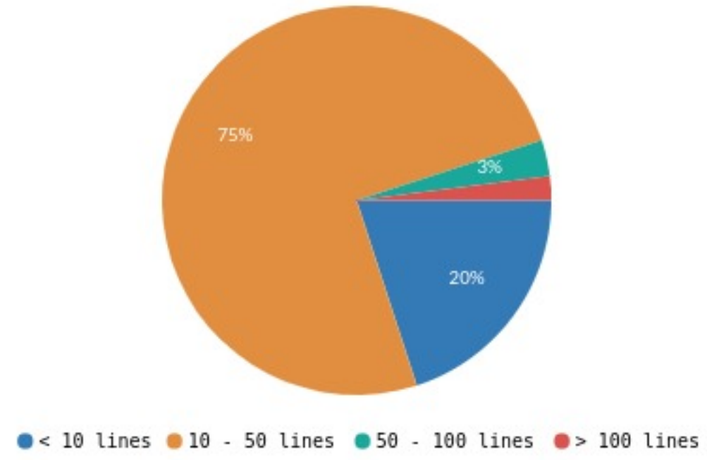


Threads Stack Length

(Lengthy stacks can cause StackOverflowError. To learn more visit here)

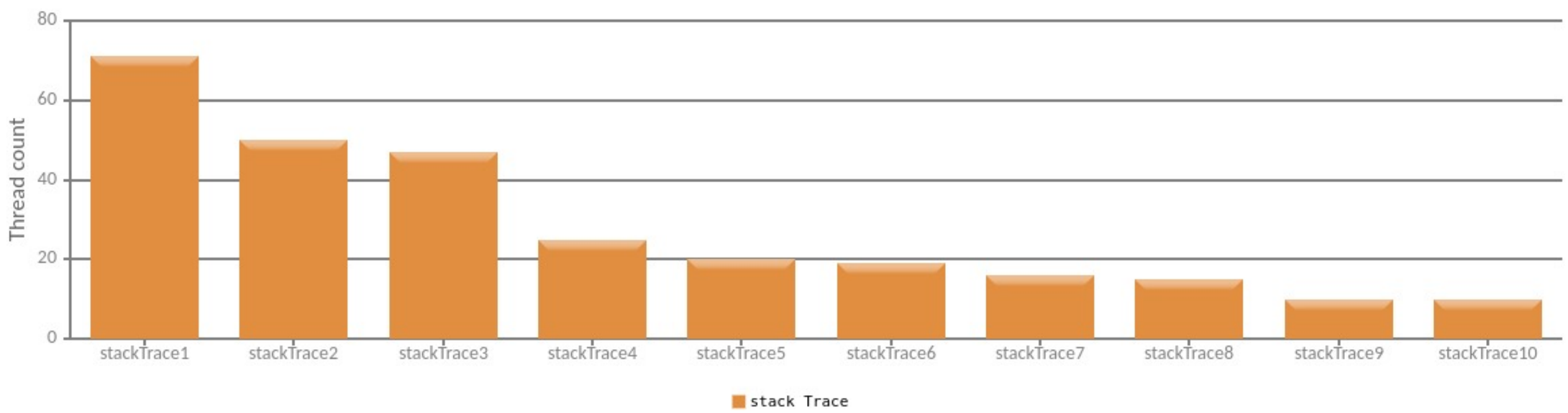
6 threads stack length is greater 100 lines. Large size stacks may cause StackOverflowError. Examine their stack trace, to make sure they are OK.

Stack Length	Thread count
< 10 lines	71
10 - 50 lines	263
50 - 100 lines	10
> 100 lines	6



Threads with identical stack trace

(Threads with identical stack traces are grouped here. If too many threads have identical stack trace then it might be a concern, to learn more visit RSI Pattern)



Thread Count	Identical Stack trace
71 WAITING threads	java.lang.Thread.State: WAITING (parking) at sun.misc.Unsafe.park(Native Method) - parking to wait for <0x00000006c0f8a930> (a java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject) at java.util.concurrent.locks.LockSupport.park(LockSupport.java:175) at java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject.await(AbstractQueuedSynchronizer.java:2039) ... To see full stack trace click here.
50 TIMED_WAITING threads	java.lang.Thread.State: TIMED_WAITING (parking) at sun.misc.Unsafe.park(Native Method) - parking to wait for <0x00000006c5413e38> (a java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject) at java.util.concurrent.locks.LockSupport.parkNanos(LockSupport.java:215) at java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject.awaitNanos(AbstractQueuedSynchronizer.java:2078) ... To see full stack trace click here.
47 RUNNABLE threads	java.lang.Thread.State: RUNNABLE at sun.nio.ch.EPollArrayWrapper.epollWait(Native Method) at sun.nio.ch.EPollArrayWrapper.poll(EPollArrayWrapper.java:269) at sun.nio.ch.EPollSelectorImpl.doSelect(EPollSelectorImpl.java:93) at sun.nio.ch.SelectorImpl.lockAndDoSelect(SelectorImpl.java:86) ... To see full stack trace click here.
25 WAITING threads	java.lang.Thread.State: WAITING (parking) at sun.misc.Unsafe.park(Native Method) - parking to wait for <0x00000006c18edf20> (a java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject) at java.util.concurrent.locks.LockSupport.park(LockSupport.java:175) at java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject.await(AbstractQueuedSynchronizer.java:2039) ... To see full stack trace click here.
20 RUNNABLE threads	stacktrace To see full stack trace click here.
19 TIMED_WAITING threads	java.lang.Thread.State: TIMED_WAITING (parking) at sun.misc.Unsafe.park(Native Method) - parking to wait for <0x00000006c10a00d0> (a java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject) at java.util.concurrent.locks.LockSupport.parkNanos(LockSupport.java:215) at java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject.awaitNanos(AbstractQueuedSynchronizer.java:2078) ...

	To see full stack trace click here.
16 RUNNABLE threads	java.lang.Thread.State: RUNNABLE Locked ownable synchronizers: -None To see full stack trace click here.
15 WAITING threads	java.lang.Thread.State: WAITING (parking) at sun.misc.Unsafe.park(Native Method) - parking to wait for <0x00000006c0e4bb20> (a java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject) at java.util.concurrent.locks.LockSupport.park(LockSupport.java:175) at java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject.await(AbstractQueuedSynchronizer.java:2039) ... To see full stack trace click here.
10 WAITING threads	java.lang.Thread.State: WAITING (parking) at sun.misc.Unsafe.park(Native Method) - parking to wait for <0x00000006c582fab0> (a java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject) at java.util.concurrent.locks.LockSupport.park(LockSupport.java:175) at java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject.await(AbstractQueuedSynchronizer.java:2039) ... To see full stack trace click here.
10 RUNNABLE threads	java.lang.Thread.State: RUNNABLE at sun.nio.fs.LinuxWatchService.poll(Native Method) at sun.nio.fs.LinuxWatchService.access$600(LinuxWatchService.java:47) at sun.nio.fs.LinuxWatchService$Poller.run(LinuxWatchService.java:314) at java.lang.Thread.run(Thread.java:748) ... To see full stack trace click here.

Most used methods

(Methods in which most threads are working are displayed here. If too many threads end up on the same method, it may be a concern, to learn more visit All roads lead to Rome pattern)

Thread Count	Method	Percentage
226 threads	sun.misc.Unsafe.park(Native Method). To see stack trace click here.	65%

52 threads	sun.nio.ch.EPollArrayWrapper.epollWait(Native Method). To see stack trace click here.	15%
12 threads	java.lang.Object.wait(Native Method). To see stack trace click here.	3%
10 threads	sun.nio.fs.LinuxWatchService.poll(Native Method). To see stack trace click here.	3%
4 threads	java.net.SocketInputStream.socketRead0(Native Method). To see stack trace click here.	1%

Show all methods >>

CPU consuming threads

(If application is consuming high CPU, investigate below threads. To learn more visit Athlete pattern)

Thread	CPU consuming thread's stacktrace
EE-ManagedThreadFactory-engine nativeId: 24394	java.lang.Thread.State: RUNNABLE at java.lang.Object.hashCode(Native Method) at java.util.HashMap.hash(HashMap.java:339) at java.util.HashMap.put(HashMap.java:612) at java.util.HashSet.add(HashSet.java:220) ... To see full stack trace click here.
EE-ManagedThreadFactory-engine nativeId: 24387	java.lang.Thread.State: RUNNABLE at java.util.LinkedHashMap.newNode(LinkedHashMap.java:256) at java.util.HashMap.putVal(HashMap.java:631) at java.util.HashMap.put(HashMap.java:612) at org.codehaus.jackson.map.deser.std.UntypedObjectDeserializer.mapObject(UntypedObjectDeserializer.java:218) ... To see full stack trace click here.
EE-ManagedThreadFactory-engine nativeId: 24383	java.lang.Thread.State: RUNNABLE at java.lang.Object.hashCode(Native Method) at java.util.HashMap.hash(HashMap.java:339) at java.util.HashMap.put(HashMap.java:612) at java.util.HashSet.add(HashSet.java:220)

	... To see full stack trace click here.
EE-ManagedThreadFactory-engine nativeId: 24379	java.lang.Thread.State: RUNNABLE at org.postgresql.core.v3.SimpleParameterList.(SimpleParameterList.java:41) at org.postgresql.core.v3.SimpleParameterList.copy(SimpleParameterList.java:343) at org.postgresql.jdbc2.AbstractJdbc2Statement.addBatch(AbstractJdbc2Statement.java:2975) at org.jboss.jca.adapters.jdbc.CachedPreparedStatement.addBatch(CachedPreparedStatement.java:311) ... To see full stack trace click here.
EE-ManagedThreadFactory-engine nativeId: 24359	java.lang.Thread.State: RUNNABLE at java.lang.Object.hashCode(Native Method) at java.util.HashMap.hash(HashMap.java:339) at java.util.HashMap.put(HashMap.java:612) at java.util.HashSet.add(HashSet.java:220) ... To see full stack trace click here.

Show all CPU consuming threads >>

Blocking Threads - Transitive Graph

(Threads that block other threads are displayed here. Blocking threads makes application unresponsive, to learn more visit Traffic jam pattern)

☑ No transitive blocks found

GC Threads

(Displays garbage collection threads count. To learn more visit Scavengers pattern)

18
GC threads

View Details

GC thread count is normal

Complex DeadLocks

(Learn more about Complex Deadlock)

No Complex Deadlocks found

Dead Lock

(Learn more about Deadlock)

No Deadlock found

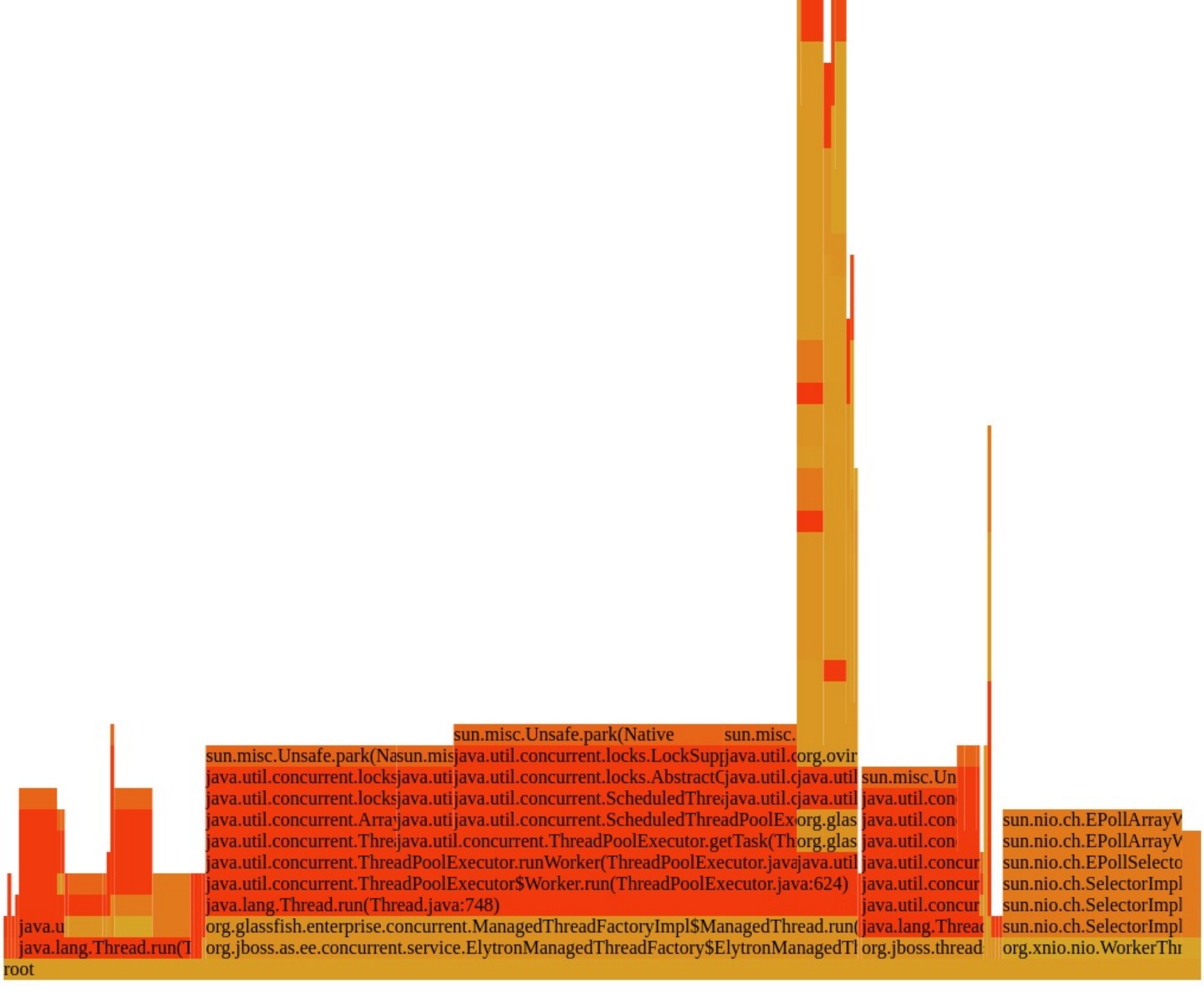
Finalizer Thread

(If finalizer thread is BLOCKED or WAITING for a prolonged period, it can result in OutOfMemoryError, to learn more visit Leprechaun Trap pattern)

No problem with Finalizer Thread.

Flame Graph

(All threads stack trace is combined in to one single flame graph. See it's benefits.)


sun.misc.Unsafe.park(Native
sun.misc.
sun.misc.Unsafe.park(Na
sun.mis
java.util.concurrent.locks.LockSupp
java.util.c
org.ovir
java.util.concurrent.locks
java.uti
java.util.concurrent.locks.AbstractC
java.util.c
java.util
sun.misc.Un
java.util.concurrent.locks
java.uti
java.util.concurrent.ScheduledThre
java.util.c
java.util
java.util.con
java.util.concurrent.Arra
java.uti
java.util.concurrent.ScheduledThreadPoolEx
org.glas
java.util.con
sun.nio.ch.EPollArrayV
java.util.concurrent.Thre
java.util.concurrent.ThreadPoolExecutor.getTask(Th
org.glas
java.util.con
sun.nio.ch.EPollArrayV
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java
java.util
java.util.concur
sun.nio.ch.EPollSelecto
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
java.util.concur
sun.nio.ch.SelectorImpl
java.lang.Thread.run(Thread.java:748)
java.util.concur
sun.nio.ch.SelectorImpl
java.u
org.glassfish.enterprise.concurrent.ManagedThreadFactoryImpl$ManagedThread.run(
java.lang.Thread
sun.nio.ch.SelectorImpl
java.lang.Thread.run(T
org.jboss.as.ee.concurrent.service.ElytronManagedThreadFactory$ElytronManagedTh
org.jboss.thread
org.xnio.nio.WorkerThr
root

Bottom up Call Stack Tree

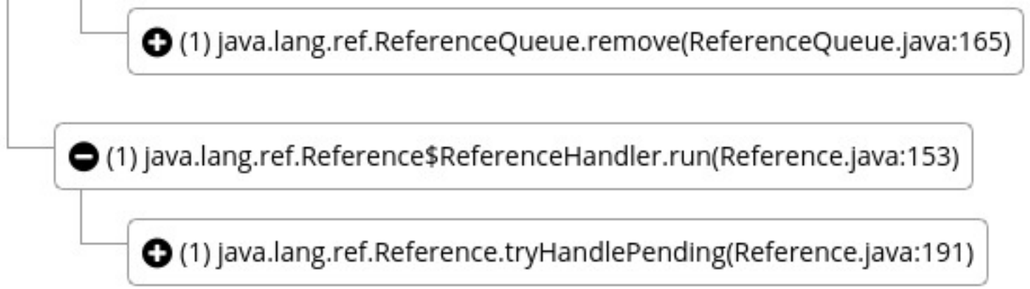
(All threads stack trace is combined in to one single tree. See it's benefits.)

Show Top Down call stack

- (314) root
  - (171) org.jboss.as.ee.concurrent.service.ElytronManagedThreadFactory$ElytronManagedThread.run(ElytronManagedThreadFactory.java:78)
    - (171) org.glassfish.enterprise.concurrent.ManagedThreadFactoryImpl$ManagedThread.run(ManagedThreadFactoryImpl.java:250)
  - (47) org.xnio.nio.WorkerThread.run(WorkerThread.java:532)
    - (47) sun.nio.ch.SelectorImpl.select(SelectorImpl.java:101)
  - (45) java.lang.Thread.run(Thread.java:748)
    - (12) java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
    - (10) org.xnio.nio.WatchServiceFileSystemWatcher.run(WatchServiceFileSystemWatcher.java:86)
    - (10) sun.nio.fs.LinuxWatchService$Poller.run(LinuxWatchService.java:314)
    - (9) org.jboss.threads.EnhancedQueueExecutor$ThreadBody.run(EnhancedQueueExecutor.java:1410)
    - (1) org.jboss.threads.EnhancedQueueExecutor$ThreadBody.run(EnhancedQueueExecutor.java:1422)
    - (1) org.ovirt.vdsm.jsonrpc.client.internal.ResponseTracker.run(ResponseTracker.java:98)
    - (1) org.wildfly.clustering.service.concurrent.ClassLoaderThreadFactory$$Lambda$657/282809408.run(Unknown Source)

- (1) org.jboss.jca.core.connectionmanager.pool.mcp.PoolFiller.run(PoolFiller.java:107)
- (32) org.jboss.threads.JBossThread.run(JBossThread.java:485)
  - (32) java.lang.Thread.run(Thread.java:748)
- (5) org.xnio.nio.WorkerThread.run(WorkerThread.java:551)
  - (5) sun.nio.ch.SelectorImpl.select(SelectorImpl.java:97)
- (1) org.apache.commons.httpclient.MultiThreadedHttpConnectionManager$ReferenceQueueThread.run(Unknown Source)
  - (1) java.lang.ref.ReferenceQueue.remove(ReferenceQueue.java:165)
- (1) org.ovirt.vdsm.jsonrpc.client.internal.ResponseWorker.run(ResponseWorker.java:108)
  - (1) org.codehaus.jackson.map.ObjectMapper.readTree(ObjectMapper.java:1558)
- (1) org.ovirt.vdsm.jsonrpc.client.reactors.Reactor.run(Reactor.java:65)
  - (1) org.ovirt.vdsm.jsonrpc.client.reactors.Reactor.processChannels(Reactor.java:76)
- (1) org.quartz.impl.jdbcjobstore.JobStoreSupport$MisfireHandler.run(JobStoreSupport.java:3987)
  - (1) java.lang.Thread.sleep(Native Method)
- (1) org.quartz.core.QuartzSchedulerThread.run(QuartzSchedulerThread.java:431)
  - (1) java.lang.Object.wait(Native Method)

- (1) org.quartz.core.QuartzSchedulerThread.run(QuartzSchedulerThread.java:301)
  - (1) java.lang.Object.wait(Native Method)
- (1) com.arjuna.ats.internal.arjuna.coordinator.ReaperWorkerThread.run(ReaperWorkerThread.java:65)
  - (1) com.arjuna.ats.arjuna.coordinator.TransactionReaper.waitForCancellations(TransactionReaper.java:328)
- (1) com.arjuna.ats.internal.arjuna.coordinator.ReaperThread.run(ReaperThread.java:90)
  - (1) java.lang.Object.wait(Native Method)
- (1) com.arjuna.ats.internal.arjuna.recovery.PeriodicRecovery.run(PeriodicRecovery.java:398)
  - (1) com.arjuna.ats.internal.arjuna.recovery.PeriodicRecovery.doPeriodicWait(PeriodicRecovery.java:678)
- (1) com.arjuna.ats.internal.arjuna.recovery.ExpiredEntryMonitor.run(ExpiredEntryMonitor.java:190)
  - (1) java.lang.Object.wait(Native Method)
- (1) java.util.TimerThread.run(Timer.java:505)
  - (1) java.util.TimerThread.mainLoop(Timer.java:526)
- (1) org.jboss.modules.ref.References$ReaperThread.run(References.java:64)
  - (1) java.lang.ref.ReferenceQueue.remove(ReferenceQueue.java:165)
- (1) java.lang.ref.Finalizer$FinalizerThread.run(Finalizer.java:216)


(1) java.lang.ref.ReferenceQueue.remove(ReferenceQueue.java:165)
(1) java.lang.ref.Reference$ReferenceHandler.run(Reference.java:153)
(1) java.lang.ref.Reference.tryHandlePending(Reference.java:191)