

# Thread Dump - Intelligence Report

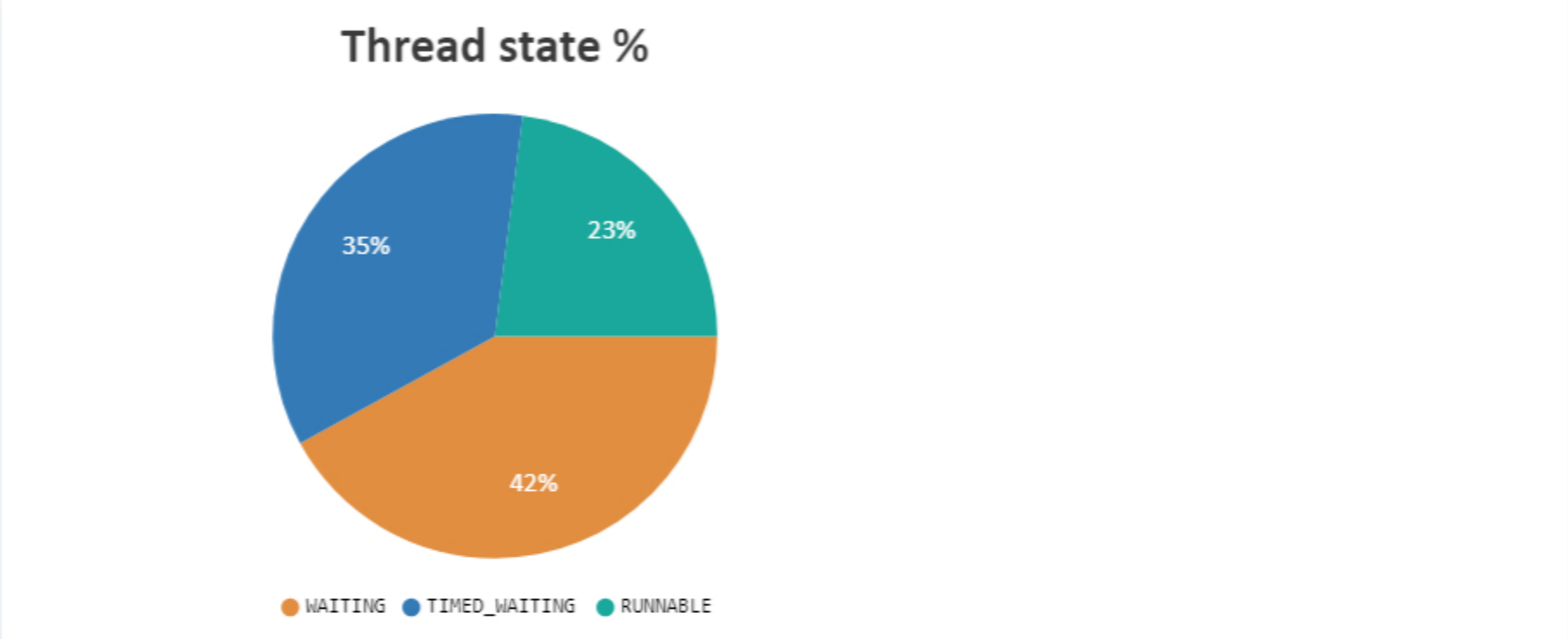
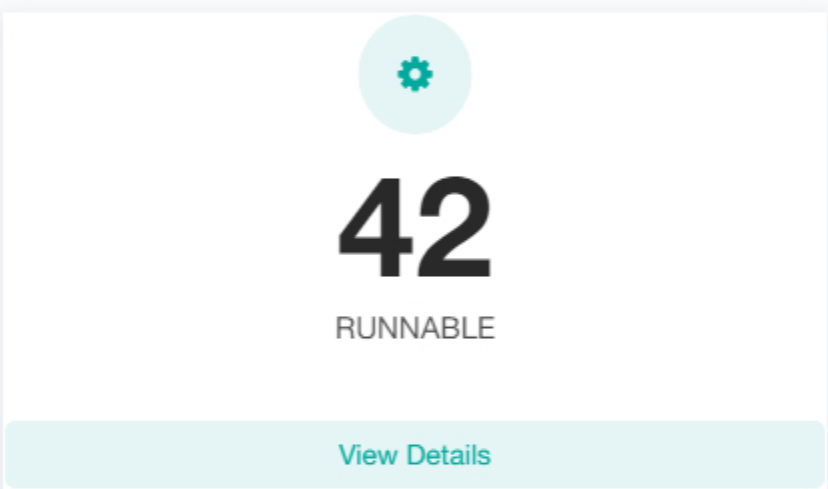
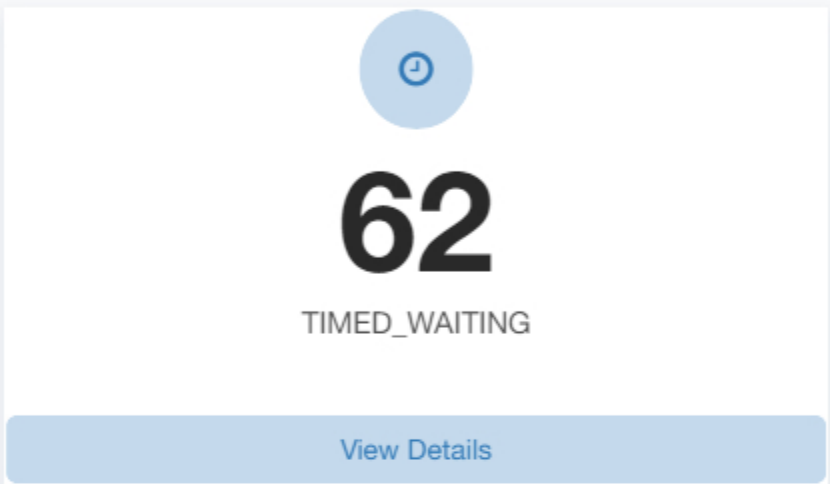
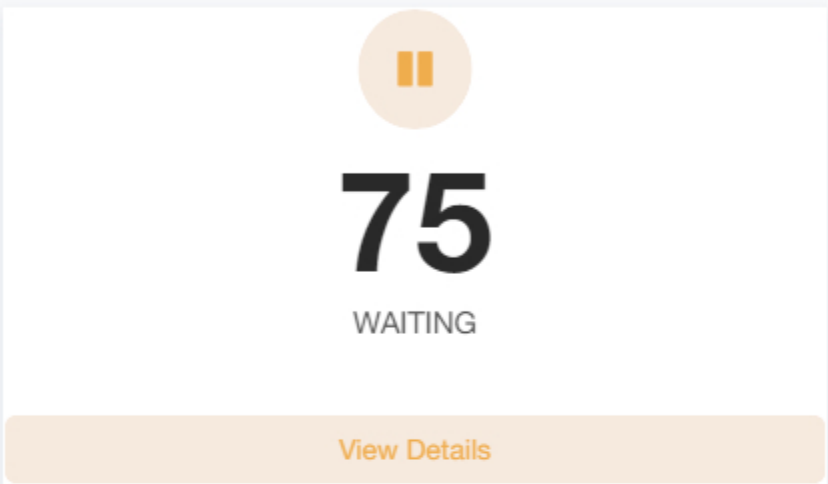
📄 File: *prd-s2w-sailpoint-workday-prc-api-instance-66533439163cad1197f4121f-0-diag (1).txt*

🕒 Timestamp: 2024-06-08 19:23:52

## Thread Count Summary

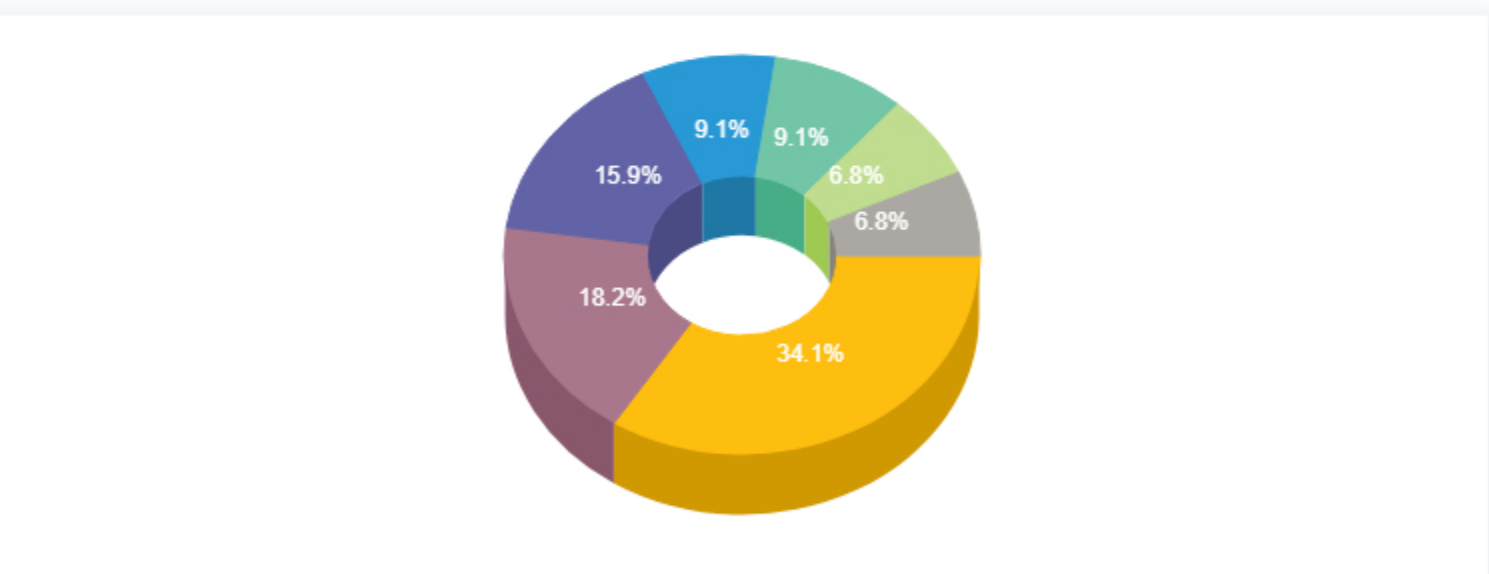
📖 To learn about different thread states through real-life example, check out this [video tutorial](#)

Total Threads count: 179



## Thread Pools

Threads with similar names are grouped in this section

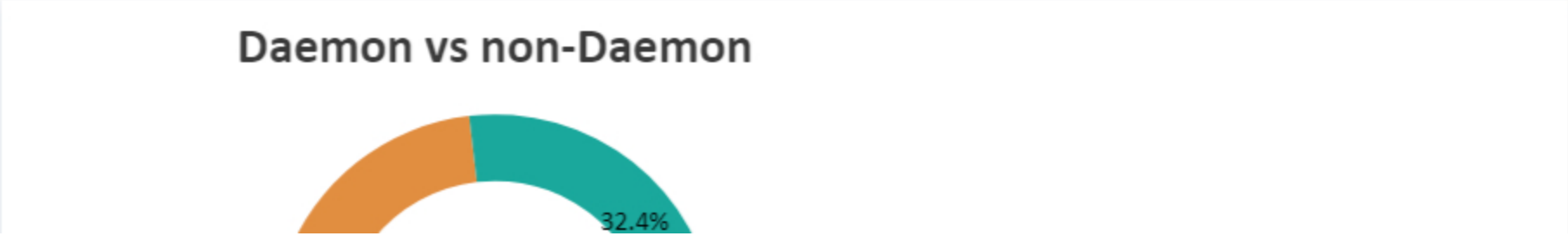
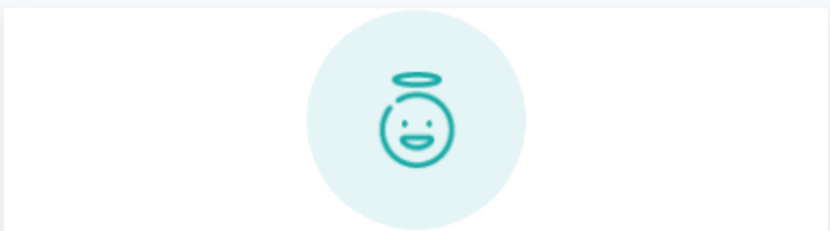


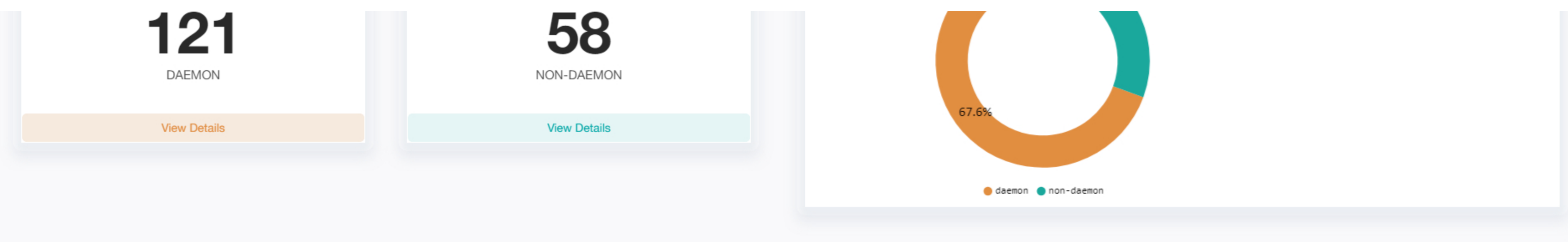
	Thread Pool	Count	States
●	<a href="#">Thread</a>	15 threads	RUNNABLE: 15
●	<a href="#">Weave File Deleter</a>	8 threads	WAITING: 8
●	<a href="#">qtp2056119935</a>	7 threads	TIMED_WAITING: 6, RUNNABLE: 1
●	<a href="#">cloudhub.platform.extensions-4</a>	4 threads	RUNNABLE: 4
●	<a href="#">AsyncHttpClient-3</a>	4 threads	RUNNABLE: 4
●	<a href="#">sshd-SshClient[748fc8e7]-nio2-thread</a>	3 threads	WAITING: 3

[Show all thread groups >>](#)


## Daemon vs non-Daemon

Learn more about [daemon and non-daemon \(i.e. user threads\)](#)

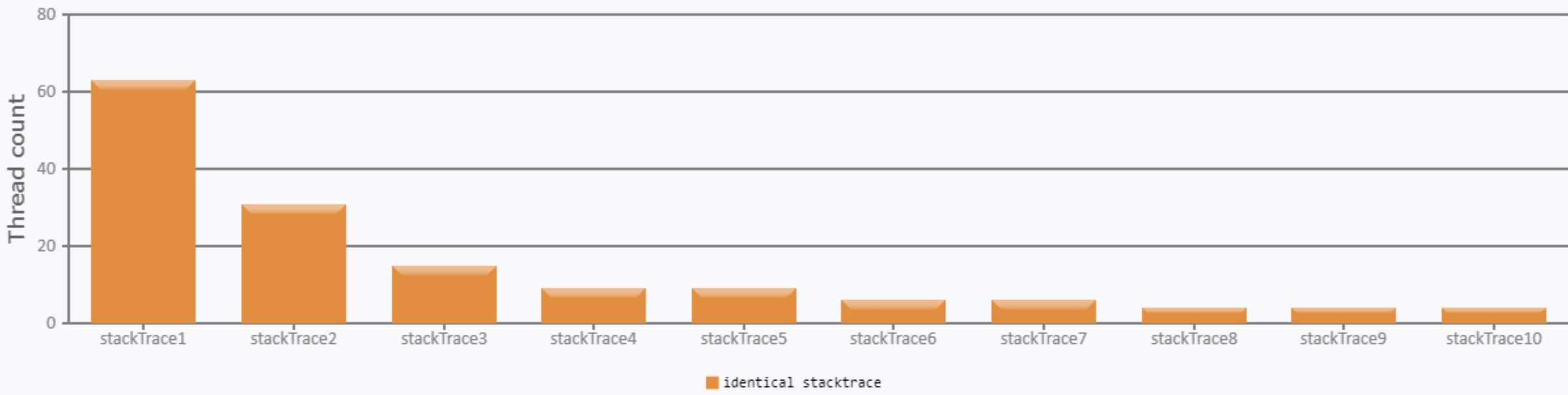




## Threads with identical stack trace

 Become Performance Expert! Training from FastThread Architect!

Threads with identical stack traces are grouped here. If lot of threads start to exhibit identical stack trace it might be a concern, learn [BSI Pattern](#)



Thread Count	Identical Stack trace
63 WAITING threads	<div>java.lang.Thread.State: WAITING (parking) at sun.misc.Unsafe.park(Native Method) - parking to wait for &lt;0x00000000e89efd80&gt; (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) ... See complete <a href="#">stacktrace</a>.</div>
31 TIMED_WAITING threads	<div>java.lang.Thread.State: TIMED_WAITING (parking) at sun.misc.Unsafe.park(Native Method) - parking to wait for &lt;0x00000000e89f9ba8&gt; (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) ... See complete <a href="#">stacktrace</a>.</div>
15 RUNNABLE threads	<div>java.lang.Thread.State: RUNNABLE at sun.nio.ch.EPoll.epollWait(Native Method) at sun.nio.ch.EPollPort\$EventHandlerTask.poll(EPollPort.java:194) at sun.nio.ch.EPollPort\$EventHandlerTask.run(EPollPort.java:268) at java.lang.Thread.run(Thread.java:750)  See complete <a href="#">stacktrace</a>.</div>
9 RUNNABLE threads	<div>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) ... See complete <a href="#">stacktrace</a>.</div>
9 TIMED_WAITING threads	<div>java.lang.Thread.State: TIMED_WAITING (on object monitor) at java.lang.Object.wait(Native Method) at java.lang.Object.wait(Object.java:460) at org.apache.logging.log4j.core.async.TimeoutBlockingWaitStrategy.awaitNanos(TimeoutBlockingWaitStrategy.java:130) at org.apache.logging.log4j.core.async.TimeoutBlockingWaitStrategy.waitFor(TimeoutBlockingWaitStrategy.java:86) ... See complete <a href="#">stacktrace</a>.</div>
6 RUNNABLE threads	<div>java.lang.Thread.State: RUNNABLE  See complete <a href="#">stacktrace</a>.</div>
6 RUNNABLE threads	<div>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) ... See complete <a href="#">stacktrace</a>.</div>
4 RUNNABLE threads	<div>stacktrace See complete <a href="#">stacktrace</a>.</div>
4 WAITING threads	<div>java.lang.Thread.State: WAITING (parking) at sun.misc.Unsafe.park(Native Method) - parking to wait for &lt;0x00000000e6ce5678&gt; (a java.util.concurrent.locks.AbstractQueuedSynchronizer\$ConditionObject)</div>

	<pre>parking to wait for: &lt;0x00000000e36b1ce0&gt; (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) ... See complete <a href="#">stacktrace</a>.</pre>
4 TIMED_WAITING threads	<pre>java.lang.Thread.State: TIMED_WAITING (parking) at sun.misc.Unsafe.park(Native Method) - parking to wait for &lt;0x00000000e36b1ce0&gt; (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) ... See complete <a href="#">stacktrace</a>.</pre>

## Last executed methods

Methods that threads were executing when thread dump was captured is reported. Learn [All roads lead to Rome pattern](#)

Thread Count	Method	Percentage
109 threads	<pre>sun.misc.Unsafe.park(Native Method) To see stack trace <a href="#">click here</a>.</pre>	61% <div><div></div></div>
22 threads	<pre>java.lang.Object.wait(Native Method) To see stack trace <a href="#">click here</a>.</pre>	12% <div><div></div></div>
16 threads	<pre>sun.nio.ch.EPollArrayWrapper.epollWait(Native Method) To see stack trace <a href="#">click here</a>.</pre>	9% <div><div></div></div>
15 threads	<pre>sun.nio.ch.EPoll.epollWait(Native Method) To see stack trace <a href="#">click here</a>.</pre>	8% <div><div></div></div>
5 threads	<pre>java.lang.Thread.sleep(Native Method) To see stack trace <a href="#">click here</a>.</pre>	3% <div><div></div></div>

[Show all methods >>](#)

## CPU consuming threads

If application is consuming high CPU, investigate below threads. Learn [Athlete pattern](#)

⊖

Not reported!

Need help diagnosing high CPU consumption? Learn our 

💡 Effective Tips

## Blocking Threads - Transitive Graph

Threads that block other threads are displayed here. Blocking threads makes application unresponsive, learn [Traffic Jam pattern](#)

☑

No transitive blocks found

## GC Threads

Garbage collection threads count reported. Learn [Scavengers pattern](#)

2

GC threads

View Details

☑

GC thread count is normal

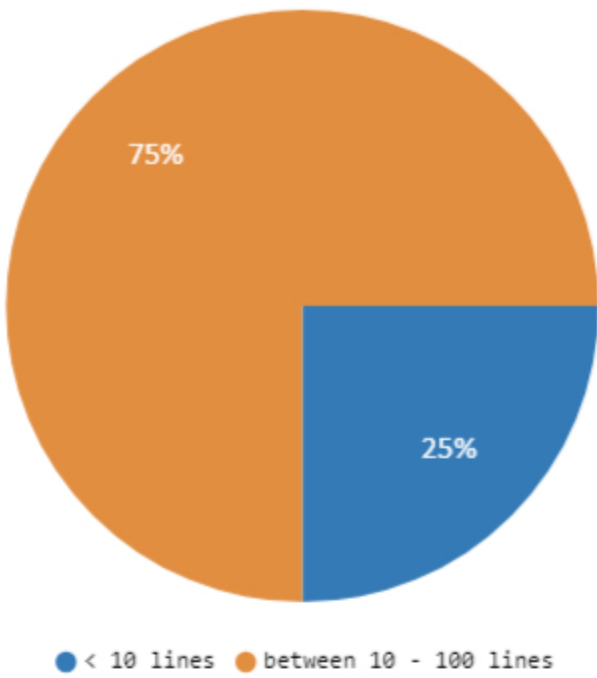
## Threads Stack Length

Lengthy stacks can cause StackOverflowError. [Learn more](#)

Longer stacktraces can cause stack overflow on [JVMs](#)

☒ No Problem in Stack trace length.

Stack Length	Thread count
< 10 lines	44
between 10 - 100 lines	135



## 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.

## Exception

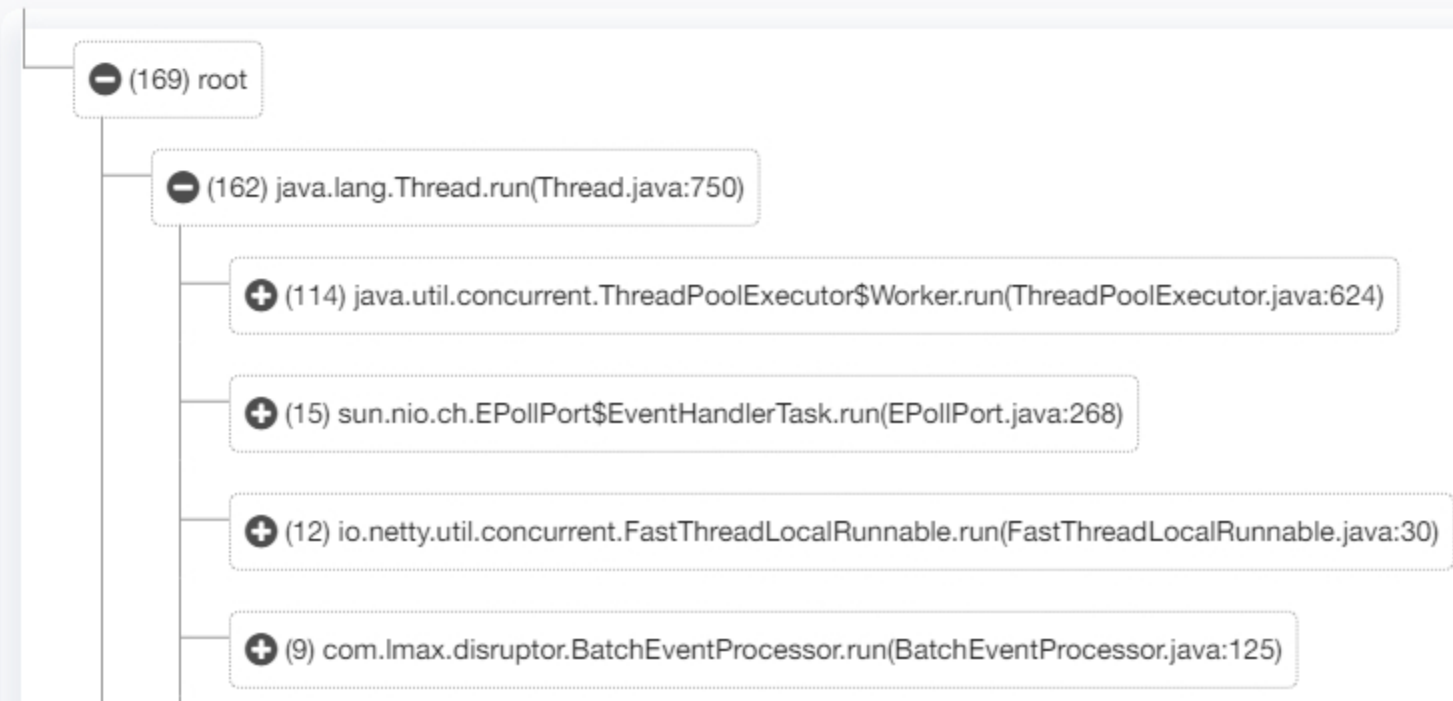
Threads throwing commonly known Exceptions/Errors are reported here. [Learn more](#)

☒ No known exceptions are reported.

## Bottom up Call Stack Tree

Reverse Call stack

All threads stacktrace are combined in to one single tree. Learn [it's benefits](#).







## My Patterns

Manage my Patterns

Create, apply, and visualize custom domain patterns for precise analysis. [Learn More](#)



No Pattern Found



### Divide & Conquer Panel

View More



### Search Panel

View More