Item 191
git_comments:

1. the parent thread remains blocked to simulate the child thread holding some lock in the registry/reporter the child thread continues execution and calls getAllVariables() in the past this would block indefinitely since the method acquires the locks of all parent groups
2. start both threads and have them block in the registry, so they acquire the lock of their respective group
3. wait with a timeout to ensure the finally block is executed _at some point_, un-blocking the parent

git_commits:

1. **summary:** [FLINK-10761][metrics] Do not acquire lock for getAllVariables()
   message: [FLINK-10761][metrics] Do not acquire lock for getAllVariables()

github_issues:

github_issues_comments:

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** MetricGroup#getAllVariables can deadlock
   description: {{AbstractMetricGroup#getAllVariables}} acquires the locks of both the current and all parent groups when assembling the variables map. This can lead to a deadlock if metrics are registered concurrently on a child and parent if the child registration is applied first and the reporter uses said method (which many do). Assume we have a MetricGroup Mc(hild) and Mp(arent). 2 separate threads Tc and Tp each register a metric on their respective group, acquiring the lock. Let's assume that Tc has a slight headstart. Tc will now call {{MetricRegistry#register}} first, acquiring the MR lock. Tp will block on this lock. Tc now iterates over all reporters calling {{MetricReporter#notifyOfAddedMetric}}. Assume that in this method {{MetricGroup#getAllVariables}} is called on Mc by Tc. Tc still holds the lock to Mc, and attempts to acquire the lock to Mp. The lock to Mp is still held by Tp however, which waits for the MR lock to be released by Tc. Thus a deadlock is created. This may deadlock anything, be it minor threads, tasks, or entire components. This has not surfaced so far since usually metrics are no longer added to a group once children have been created (since the component initialization at that point is complete).
2. **summary:** MetricGroup#getAllVariables can deadlock
   description: {{AbstractMetricGroup#getAllVariables}} acquires the locks of both the current and all parent groups when assembling the variables map. This can lead to a deadlock if metrics are registered concurrently on a child and parent if the child registration is applied first and the reporter uses said method (which many do). Assume we have a MetricGroup Mc(hild) and Mp(arent). 2 separate threads Tc and Tp each register a metric on their respective group, acquiring the lock. Let's assume that Tc has a slight headstart. Tc will now call {{MetricRegistry#register}} first, acquiring the MR lock. Tp will block on this lock. Tc now iterates over all reporters calling {{MetricReporter#notifyOfAddedMetric}}. Assume that in this method {{MetricGroup#getAllVariables}} is called on Mc by Tc. Tc still holds the lock to Mc, and attempts to acquire the lock to Mp. The lock to Mp is still held by Tp however, which waits for the MR lock to be released by Tc. Thus a deadlock is created. This may deadlock anything, be it minor threads, tasks, or entire components. This has not surfaced so far since usually metrics are no longer added to a group once children have been created (since the component initialization at that point is complete).
3. **summary:** MetricGroup#getAllVariables can deadlock
   description: {{AbstractMetricGroup#getAllVariables}} acquires the locks of both the current and all parent groups when assembling the variables map. This can lead to a deadlock if metrics are registered concurrently on a child and parent if the child registration is applied first and the reporter uses said method (which many do). Assume we have a MetricGroup Mc(hild) and Mp(arent). 2 separate threads Tc

and Tp each register a metric on their respective group, acquiring the lock. Let's assume that Tc has a slight headstart. Tc will now call {{MetricRegistry#register}} first, acquiring the MR lock. Tp will block on this lock. Tc now iterates over all reporters calling {{MetricReporter#notifyOfAddedMetric}}. Assume that in this method {{MetricGroup#getAllVariables}} is called on Mc by Tc. Tc still holds the lock to Mc, and attempts to acquire the lock to Mp. The lock to Mp is still held by Tp however, which waits for the MR lock to be released by Tc. Thus a deadlock is created. This may deadlock anything, be it minor threads, tasks, or entire components. This has not surfaced so far since usually metrics are no longer added to a group once children have been created (since the component initialization at that point is complete).
label: code-design

4. **summary:** MetricGroup#getAllVariables can deadlock
description: {{AbstractMetricGroup#getAllVariables}} acquires the locks of both the current and all parent groups when assembling the variables map. This can lead to a deadlock if metrics are registered concurrently on a child and parent if the child registration is applied first and the reporter uses said method (which many do). Assume we have a MetricGroup Mc(hild) and Mp(arent). 2 separate threads Tc and Tp each register a metric on their respective group, acquiring the lock. Let's assume that Tc has a slight headstart. Tc will now call {{MetricRegistry#register}} first, acquiring the MR lock. Tp will block on this lock. Tc now iterates over all reporters calling {{MetricReporter#notifyOfAddedMetric}}. Assume that in this method {{MetricGroup#getAllVariables}} is called on Mc by Tc. Tc still holds the lock to Mc, and attempts to acquire the lock to Mp. The lock to Mp is still held by Tp however, which waits for the MR lock to be released by Tc. Thus a deadlock is created. This may deadlock anything, be it minor threads, tasks, or entire components. This has not surfaced so far since usually metrics are no longer added to a group once children have been created (since the component initialization at that point is complete).
5. **summary:** MetricGroup#getAllVariables can deadlock
description: {{AbstractMetricGroup#getAllVariables}} acquires the locks of both the current and all parent groups when assembling the variables map. This can lead to a deadlock if metrics are registered concurrently on a child and parent if the child registration is applied first and the reporter uses said method (which many do). Assume we have a MetricGroup Mc(hild) and Mp(arent). 2 separate threads Tc and Tp each register a metric on their respective group, acquiring the lock. Let's assume that Tc has a slight headstart. Tc will now call {{MetricRegistry#register}} first, acquiring the MR lock. Tp will block on this lock. Tc now iterates over all reporters calling {{MetricReporter#notifyOfAddedMetric}}. Assume that in this method {{MetricGroup#getAllVariables}} is called on Mc by Tc. Tc still holds the lock to Mc, and attempts to acquire the lock to Mp. The lock to Mp is still held by Tp however, which waits for the MR lock to be released by Tc. Thus a deadlock is created. This may deadlock anything, be it minor threads, tasks, or entire components. This has not surfaced so far since usually metrics are no longer added to a group once children have been created (since the component initialization at that point is complete).
6. **summary:** MetricGroup#getAllVariables can deadlock
description: {{AbstractMetricGroup#getAllVariables}} acquires the locks of both the current and all parent groups when assembling the variables map. This can lead to a deadlock if metrics are registered concurrently on a child and parent if the child registration is applied first and the reporter uses said method (which many do). Assume we have a MetricGroup Mc(hild) and Mp(arent). 2 separate threads Tc and Tp each register a metric on their respective group, acquiring the lock. Let's assume that Tc has a slight headstart. Tc will now call {{MetricRegistry#register}} first, acquiring the MR lock. Tp will block on this lock. Tc now iterates over all reporters calling {{MetricReporter#notifyOfAddedMetric}}. Assume that in this method {{MetricGroup#getAllVariables}} is called on Mc by Tc. Tc still holds the lock to Mc, and attempts to acquire the lock to Mp. The lock to Mp is still held by Tp however, which waits for the MR lock to be released by Tc. Thus a deadlock is created. This may deadlock anything, be it minor threads, tasks, or entire components. This has not surfaced so far since usually metrics are no longer added to a group once children have been created (since the component initialization at that point is complete).

jira_issues_comments:

1. Is this strictly a blocker given that it is in Flink for a couple of versions [~chesnay]. I think we should try to fix for 1.7, though.
2. Probably doesn't have to be a blocker considering we've never observed it so far outside of dev branches. It's fairly unlikely to happen; the only case i can imagine on master is a registration in a UDF from a

separate thread while a latency histogram is being registered. This is quite specific, but would deadlock a single task.

3. **body:** This is a bit difficult to solve since multiple methods may violate the lock-order. It would be easier to fix if asynchronous registration of metrics is already in place, as in that case no thread could be blocked while registering a metric, and thus while holding a MetricGroup lock. Given that these issues aren't new and have never been observed I'm unblocking the 1.7 release. We should look into these issues for 1.8 as part of a general concurrency-cleanup in the metric system.
   label: code-design
4. master: aa92438c4aa6218c5d7e67083152733271697089 1.7: 45d53d27e4dec934057404674e38c6eab2e66e62 1.6: cb85e62b5b122b7cbb9eeb72a116dd133467a920