Item 198
git_comments:

git_commits:

1. **summary:** [PINOT-3882] Bumping up zookeeper version to handle large number of watches (#695)
   message: [PINOT-3882] Bumping up zookeeper version to handle large number of watches (#695) Controller connection to zookeeper can flap due to bug in zookeeper(ZOOKEEPER-706) when large number of watches are set. Bumping up to zookeeper client version that addresses this issues

github_issues:

github_issues_comments:

github_pulls:

1. **title:** [PINOT-3882] Bumping up zookeeper version to handle large number of watches
   body: Controller connection to zookeeper can flap due to bug in zookeeper(ZOOKEEPER-706) when large number of watches are set. Bumping up to zookeeper client version that addresses this issues
2. **title:** [PINOT-3882] Bumping up zookeeper version to handle large number of watches
   body: Controller connection to zookeeper can flap due to bug in zookeeper(ZOOKEEPER-706) when large number of watches are set. Bumping up to zookeeper client version that addresses this issues
3. **title:** [PINOT-3882] Bumping up zookeeper version to handle large number of watches
   body: Controller connection to zookeeper can flap due to bug in zookeeper(ZOOKEEPER-706) when large number of watches are set. Bumping up to zookeeper client version that addresses this issues
4. **title:** [PINOT-3882] Bumping up zookeeper version to handle large number of watches
   body: Controller connection to zookeeper can flap due to bug in zookeeper(ZOOKEEPER-706) when large number of watches are set. Bumping up to zookeeper client version that addresses this issues
5. **title:** [PINOT-3882] Bumping up zookeeper version to handle large number of watches
   body: Controller connection to zookeeper can flap due to bug in zookeeper(ZOOKEEPER-706) when large number of watches are set. Bumping up to zookeeper client version that addresses this issues

github_pulls_comments:

1. ## [Current coverage](https://codecov.io/gh/linkedin/pinot/pull/695?src=pr) is 64.35% (diff: 100%) > No coverage report found for **master** at 8678561. > > Powered by [Codecov](https://codecov.io?src=pr). Last update [8678561...0164449] (https://codecov.io/gh/linkedin/pinot/compare/867856114b24e2c290dfc5b4a020452c24e63a8a...016444912ecc9151d8997fb8672980006c9955a6?src=pr)
2. Yes. Zookeeper SREs also suggested upgrading to this version. -Adwait On Tue, Oct 18, 2016 at 7:33 PM, Kishore Gopalakrishna < notifications@github.com> wrote: > _@kishoreg_ commented on this pull request. > > Can you please confirm the zookeeper server version and make sure this > client version is compatible with server version. > > — > You are receiving this because you authored the thread. > Reply to this email directly, view it on GitHub > https://github.com/linkedin/pinot/pull/695#pullrequestreview-4795298, > or mute the thread > https://github.com/notifications/unsubscribe-auth/ABAmRoN50Kap8cYeC6lfemTSGO47OjJeks5q1YFzgaJpZM4KaXOb > .
3. Confirmed with zk team that this is safe version to use. Checking this in to test mp change.

github_pulls_reviews:

jira_issues:

1. **summary:** large numbers of watches can cause session re-establishment to fail
   description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
2. **summary:** large numbers of watches can cause session re-establishment to fail
   description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
3. **summary:** large numbers of watches can cause session re-establishment to fail
   description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
   label: code-design
4. **summary:** large numbers of watches can cause session re-establishment to fail
   description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
5. **summary:** large numbers of watches can cause session re-establishment to fail
   description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent

during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.

6. **summary:** large numbers of watches can cause session re-establishment to fail
   description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
7. **summary:** large numbers of watches can cause session re-establishment to fail
   description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
8. **summary:** large numbers of watches can cause session re-establishment to fail
   description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
9. **summary:** large numbers of watches can cause session re-establishment to fail
   description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
10. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
11. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
12. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
13. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
14. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
15. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
16. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to

fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.

17. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
18. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
19. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
20. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
21. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
22. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
    label: code-design
23. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
24. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
25. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
26. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
27. **summary:** large numbers of watches can cause session re-establishment to fail

description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.

28. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
29. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
30. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
31. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
32. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
33. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
34. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.
35. **summary:** large numbers of watches can cause session re-establishment to fail
    description: If a client sets a large number of watches the "set watches" operation during session re-establishment can fail. for example: WARN [NIOServerCxn.Factory:22801:NIOServerCnxn@417] - Exception causing close of session 0xe727001201a4ee7c due to java.io.IOException: Len error 4348380 in this case the client was a web monitoring app and had set both data and child watches on > 32k znodes. there are two issues I see here we need to fix: 1) handle this case properly (split up the set watches into multiple calls I guess...) 2) the session should have expired after the "timeout". however we seem to consider any message from the client as re-setting the expiration on the server side. Probably we should only consider messages from the client that are sent during an established session, otherwise we can see this situation where the session is not established however the session is not expired either. Perhaps we should create another JIRA for this particular issue.

jira_issues_comments:

1. This is a pretty easy one for someone to fix, and as more users use more watches it would be good to get this addressed.
2. not a blocker. Moving it out of 3.4 release.
3. **body:** I am seeing this issue coming up quite a bit right now. Some clients are getting continually disconnected/reconnected with that error message from what I assume is the large number of watches. For example: 2011-08-29 13:26:43,750 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn$Factory@251] - Accepted socket connection from /10.141.241.188:32866 2011-08-29 13:26:43,775 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@770] - Client attempting to renew session 0x1319819fcd1000b at /10.141.241.188:32866 2011-08-29 13:26:43,775 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@1580] - Established session 0x1319819fcd1000b with negotiated timeout 10000 for client /10.141.241.188:32866 2011-08-29 13:26:43,775 - WARN [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@639] - Exception causing close of session 0x1319819fcd1000b due to java.io.IOException: Len error 1150247 2011-08-29 13:26:43,775 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@1435] - Closed socket connection for client /10.141.241.188:32866 which had sessionid 0x1319819fcd1000b 2011-08-29 13:26:47,276 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn$Factory@251] - Accepted socket connection from /10.141.241.188:32871 2011-08-29

13:26:47,298 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@770] - Client attempting to renew session 0x1319819fcd1000b at /10.141.241.188:32871 2011-08-29 13:26:47,298 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@1580] - Established session 0x1319819fcd1000b with negotiated timeout 10000 for client /10.141.241.188:32871 2011-08-29 13:26:47,298 - WARN [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@639] - Exception causing close of session 0x1319819fcd1000b due to java.io.IOException: Len error 1150247 2011-08-29 13:26:47,300 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@1435] - Closed socket connection for client /10.141.241.188:32871 which had sessionid 0x1319819fcd1000b 2011-08-29 13:26:51,124 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn$Factory@251] - Accepted socket connection from /10.141.241.188:32879 2011-08-29 13:26:51,142 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@770] - Client attempting to renew session 0x1319819fcd1000b at /10.141.241.188:32879 2011-08-29 13:26:51,143 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@1580] - Established session 0x1319819fcd1000b with negotiated timeout 10000 for client /10.141.241.188:32879 2011-08-29 13:26:51,143 - WARN [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@639] - Exception causing close of session 0x1319819fcd1000b due to java.io.IOException: Len error 1150247 2011-08-29 13:26:51,143 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@1435] - Closed socket connection for client /10.141.241.188:32879 which had sessionid 0x1319819fcd1000b 2011-08-29 13:26:53,985 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn$Factory@251] - Accepted socket connection from /10.141.241.188:32885 2011-08-29 13:26:54,006 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@770] - Client attempting to renew session 0x1319819fcd1000b at /10.141.241.188:32885 2011-08-29 13:26:54,007 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@1580] - Established session 0x1319819fcd1000b with negotiated timeout 10000 for client /10.141.241.188:32885 2011-08-29 13:26:54,007 - WARN [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@639] - Exception causing close of session 0x1319819fcd1000b due to java.io.IOException: Len error 1150247 2011-08-29 13:26:54,007 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@1435] - Closed socket connection for client /10.141.241.188:32885 which had sessionid 0x1319819fcd1000b 2011-08-29 13:26:57,495 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn$Factory@251] - Accepted socket connection from /10.141.241.188:32892 2011-08-29 13:26:57,513 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@770] - Client attempting to renew session 0x1319819fcd1000b at /10.141.241.188:32892 2011-08-29 13:26:57,513 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@1580] - Established session 0x1319819fcd1000b with negotiated timeout 10000 for client /10.141.241.188:32892 2011-08-29 13:26:57,514 - WARN [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@639] - Exception causing close of session 0x1319819fcd1000b due to java.io.IOException: Len error 1150247 2011-08-29 13:26:57,514 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@1435] - Closed socket connection for client /10.141.241.188:32892 which had sessionid 0x1319819fcd1000b 2011-08-29 13:26:59,937 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn$Factory@251] - Accepted socket connection from /10.141.241.188:32897 2011-08-29 13:26:59,958 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@770] - Client attempting to renew session 0x1319819fcd1000b at /10.141.241.188:32897 2011-08-29 13:26:59,958 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@1580] - Established session 0x1319819fcd1000b with negotiated timeout 10000 for client /10.141.241.188:32897 2011-08-29 13:26:59,958 - WARN [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@639] - Exception causing close of session 0x1319819fcd1000b due to java.io.IOException: Len error 1150247 2011-08-29 13:26:59,958 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@1435] - Closed socket connection for client /10.141.241.188:32897 which had sessionid 0x1319819fcd1000b 2011-08-29 13:27:03,711 - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn$Factory@251] - Accepted socket connection from /10.141.241.188:32904 It would be great to get this fixed in an upcoming release since it is impacting us quite a bit.
label: code-design

4. You might be seeing ZOOKEEPER-1162 -- see if the workaround there fixes your problem.
5. Any idea if the jute.maxbuffer setting needs to be applied to both server and client? or just client?
6. Thanks Patrick, setting jute.maxbuffer size to a larger value in the server and client seems to have fixed the problem.
7. Good to know, thanks Eric. Even more reason to address ZOOKEEPER-1162
8. I think not only large number of watches might cause the error, but also too many ops in a single call of ZooKeeper#multi. I'm seeing the following error RM log: {code} 2015-01-20 09:45:45,464 WARN org.apache.zookeeper.ClientCnxn: Session 0x24aeb334e8e000d for server c112/10.149.27.112:2181, unexpected error, closing socket connection and attempting reconn ect java.io.IOException: Broken pipe at sun.nio.ch.FileDispatcherImpl.write0(Native Method) at sun.nio.ch.SocketDispatcher.write(SocketDispatcher.java:47) at sun.nio.ch.IOUtil.writeFromNativeBuffer(IOUtil.java:93) at sun.nio.ch.IOUtil.write(IOUtil.java:65) at sun.nio.ch.SocketChannelImpl.write(SocketChannelImpl.java:470) at org.apache.zookeeper.ClientCnxnSocketNIO.doIO(ClientCnxnSocketNIO.java:117) at org.apache.zookeeper.ClientCnxnSocketNIO.doTransport(ClientCnxnSocketNIO.java:355) at org.apache.zookeeper.ClientCnxn$SendThread.run(ClientCnxn.java:1068) 2015-01-20 09:45:45,564 INFO org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore: Exception while executing a ZK operation. org.apache.zookeeper.KeeperException$ConnectionLossException: KeeperErrorCode = ConnectionLoss at org.apache.zookeeper.KeeperException.create(KeeperException.java:99) at org.apache.zookeeper.ZooKeeper.multiInternal(ZooKeeper.java:931) at org.apache.zookeeper.ZooKeeper.multi(ZooKeeper.java:911) at org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore$4.run(ZKRMStateStore.java:895) at org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore$4.run(ZKRMStateStore.java:892) at org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore$ZKAction.runWithCheck(ZKRMStateStore.java:1031) at org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore$ZKAction.runWithRetries(ZKRMStateStore.java:1050) at org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore.doMultiWithRetries(ZKRMStateStore.java:892) at org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore.doMultiWithRetries(ZKRMStateStore.java:906) at org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore.createWithRetries(ZKRMStateStore.java:915) at org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore.storeRMDTMasterKeyState(ZKRMStateStore.java:785) at org.apache.hadoop.yarn.server.resourcemanager.recovery.RMStateStore.storeRMDTMasterKey(RMStateStore.java:678) at org.apache.hadoop.yarn.server.resourcemanager.security.RMDelegationTokenSecretManager.storeNewMasterKey(RMDelegationTokenSecretManager.java:86) at org.apache.hadoop.security.token.delegation.AbstractDelegationTokenSecretManager.updateCurrentKey(AbstractDelegationTokenSecretManager.java:233) at org.apache.hadoop.security.token.delegation.AbstractDelegationTokenSecretManager.startThreads(AbstractDelegationTokenSecretManager.java:116) at org.apache.hadoop.yarn.server.resourcemanager.RMSecretManagerService.serviceStart(RMSecretManagerService.java:83) at org.apache.hadoop.service.AbstractService.start(AbstractService.java:193) at org.apache.hadoop.service.CompositeService.serviceStart(CompositeService.java:120) at org.apache.hadoop.yarn.server.resourcemanager.ResourceManager$RMActiveServices.serviceStart(ResourceManager.java:514) at org.apache.hadoop.service.AbstractService.start(AbstractService.java:193) at org.apache.hadoop.yarn.server.resourcemanager.ResourceManager.startActiveServices(ResourceManager.java:847) at org.apache.hadoop.yarn.server.resourcemanager.ResourceManager$2.run(ResourceManager.java:888) at org.apache.hadoop.yarn.server.resourcemanager.ResourceManager$2.run(ResourceManager.java:884) at java.security.AccessController.doPrivileged(Native Method) at javax.security.auth.Subject.doAs(Subject.java:422) at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1667) at org.apache.hadoop.yarn.server.resourcemanager.ResourceManager.transitionToActive(ResourceManager.java:884) at org.apache.hadoop.yarn.server.resourcemanager.AdminService.transitionToActive(AdminService.java:275) at org.apache.hadoop.yarn.server.resourcemanager.EmbeddedElectorService.becomeActive(EmbeddedElectorService.java:116) at org.apache.hadoop.ha.ActiveStandbyElector.becomeActive(ActiveStandbyElector.java:804) at org.apache.hadoop.ha.ActiveStandbyElector.processResult(ActiveStandbyElector.java:415) at org.apache.zookeeper.ClientCnxn$EventThread.processEvent(ClientCnxn.java:596) 2015-01-20 09:59:49,665 WARN org.apache.zookeeper.ClientCnxn: Session 0x34aeb34ba360012 for server c114/10.149.27.114:2181, unexpected error, closing socket connection and attempting reconnect java.io.IOException: Broken pipe at sun.nio.ch.FileDispatcherImpl.write0(Native Method) at sun.nio.ch.SocketDispatcher.write(SocketDispatcher.java:47) at sun.nio.ch.IOUtil.writeFromNativeBuffer(IOUtil.java:93) at sun.nio.ch.IOUtil.write(IOUtil.java:65) at sun.nio.ch.SocketChannelImpl.write(SocketChannelImpl.java:470) at org.apache.zookeeper.ClientCnxnSocketNIO.doIO(ClientCnxnSocketNIO.java:117) at org.apache.zookeeper.ClientCnxnSocketNIO.doTransport(ClientCnxnSocketNIO.java:355) at org.apache.zookeeper.ClientCnxn$SendThread.run(ClientCnxn.java:1068) 2015-01-20 09:59:49,765 INFO

org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore: Exception while executing a ZK operation. org.apache.zookeeper.KeeperException$ConnectionLossException: KeeperErrorCode = ConnectionLoss at org.apache.zookeeper.KeeperException.create(KeeperException.java:99) at org.apache.zookeeper.ZooKeeper.multiInternal(ZooKeeper.java:931) at org.apache.zookeeper.ZooKeeper.multi(ZooKeeper.java:911) at org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore$4.run(ZKRMStateStore.java:895) at org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore$4.run(ZKRMStateStore.java:892) at org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore$ZKAction.runWithCheck(ZKRMStateStore.java:1031) at org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore$ZKAction.runWithRetries(ZKRMStateStore.java:1050) at org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore.doMultiWithRetries(ZKRMStateStore.java:892) at org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore.removeApplicationStateInternal(ZKRMStateStore.java:678) at org.apache.hadoop.yarn.server.resourcemanager.recovery.RMStateStore$RemoveAppTransition.transition(RMStateStore.java:184) at org.apache.hadoop.yarn.server.resourcemanager.recovery.RMStateStore$RemoveAppTransition.transition(RMStateStore.java:170) at org.apache.hadoop.yarn.state.StateMachineFactory$SingleInternalArc.doTransition(StateMachineFactory.java:362) at org.apache.hadoop.yarn.state.StateMachineFactory.doTransition(StateMachineFactory.java:302) at org.apache.hadoop.yarn.state.StateMachineFactory.access$300(StateMachineFactory.java:46) at org.apache.hadoop.yarn.state.StateMachineFactory$InternalStateMachine.doTransition(StateMachineFactory.java:448) at org.apache.hadoop.yarn.server.resourcemanager.recovery.RMStateStore.handleStoreEvent(RMStateStore.java:769) at org.apache.hadoop.yarn.server.resourcemanager.recovery.RMStateStore$ForwardingEventHandler.handle(RMStateStore.java:842) at org.apache.hadoop.yarn.server.resourcemanager.recovery.RMStateStore$ForwardingEventHandler.handle(RMStateStore.java:837) at org.apache.hadoop.yarn.event.AsyncDispatcher.dispatch(AsyncDispatcher.java:191) at org.apache.hadoop.yarn.event.AsyncDispatcher$1.run(AsyncDispatcher.java:124) at java.lang.Thread.run(Thread.java:745) {code} zk log {code} 2015-01-20 09:45:46,377 [myid:2] - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxnFactory@197] - Accepted socket connection from /10.149.27.114:33517 2015-01-20 09:45:46,381 [myid:2] - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:ZooKeeperServer@861] - Client attempting to renew session 0x24aeb334e8e000d at /10.149.27.114:33517 2015-01-20 09:45:46,381 [myid:2] - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:ZooKeeperServer@617] - Established session 0x24aeb334e8e000d with negotiated timeout 10000 for client /10.149.27.114:33517 2015-01-20 09:45:46,381 [myid:2] - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:ZooKeeperServer@892] - got auth packet /10.149.27.114:33517 2015-01-20 09:45:46,381 [myid:2] - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:ZooKeeperServer@926] - auth success /10.149.27.114:33517 2015-01-20 09:45:46,481 [myid:2] - WARN [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@362] - Exception causing close of session 0x24aeb334e8e000d due to java.io.IOException: Len error 9271869 2015-01-20 09:45:46,481 [myid:2] - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@1007] - Closed socket connection for client /10.149.27.114:33517 which had sessionid 0x24aeb334e8e000d {code}

9. I would like to submit the attached patch for review. This change handles splitting the SetWatches call after session re-establishment into multiple smaller requests to avoid hitting the maxBuffer limit of the server.
10. lgtm, +1. Thanks for the patch [~cthunes]! Do you think you could add a test? I think we could go without one, but given the assumption has always been there's only one SetWatches() packet after reconnect, it might be good to have a small test for this.
11. -1 overall. Here are the results of testing the latest attachment http://issues.apache.org/jira/secure/attachment/12739232/ZOOKEEPER-706.patch against trunk revision 1684956. +1 @author. The patch does not contain any @author tags. -1 tests included. The patch doesn't appear to include any new or modified tests. Please justify why no new tests are needed for this patch. Also please list what manual steps were performed to verify this patch. -1 patch. The patch command could not apply the patch. Console output: https://builds.apache.org/job/PreCommit-ZOOKEEPER-Build/2765//console This message is automatically generated.
12. Ah, also: the patch you attached only applies to the 3.4 branch. Do you mind renaming that patch to ZOOKEEPER-706-branch-34.patch and also submitting one that applies to trunk (which would be ZOOKEEPER-706.patch)? Thanks!
13. yeah, this is because the patch is against the 3.4 branch.
14. Sure, let me see if I can put something together. It looks like org.apache.zookeeper.test.DisconnectedWatcherTest would an appropriate place for one. Does that seem correct?
15. Will do.
16. Yeah, I think that makes sense. Something like setting 10K watches on /some/100/chars/path maybe?
17. I've add a test and attached the updated patches for both the 3.4 branch and trunk.
18. -1 overall. Here are the results of testing the latest attachment http://issues.apache.org/jira/secure/attachment/12739317/ZOOKEEPER-706.patch against trunk revision 1684956. +1 @author. The patch does not contain any @author tags. +1 tests included. The patch appears to include 3 new or modified tests. +1 javadoc. The javadoc tool did not generate any warning messages. +1 javac. The applied patch does not increase the total number of javac compiler warnings. -1 findbugs. The patch appears to introduce 1 new Findbugs (version 2.0.3) warnings. +1 release audit. The applied patch does not increase the total number of release audit warnings. -1 core tests. The patch failed core unit tests. +1 contrib tests. The patch passed contrib unit tests. Test results: https://builds.apache.org/job/PreCommit-ZOOKEEPER-Build/2766//testReport/ Findbugs warnings: https://builds.apache.org/job/PreCommit-ZOOKEEPER-Build/2766//artifact/trunk/build/test/findbugs/newPatchFindbugsWarnings.html Console output: https://builds.apache.org/job/PreCommit-ZOOKEEPER-Build/2766//console This message is automatically generated.
19. We can ignore the failed test: {code} 1 tests failed. REGRESSION: org.apache.zookeeper.test.ReconfigTest.testPortChange {code} cc: [~shralex]
20. We can ignore the failed test: {code} 1 tests failed. REGRESSION: org.apache.zookeeper.test.ReconfigTest.testPortChange {code} cc: [~shralex]
21. So chatting with [~svoutil] about this, we think there is an issue in: {code} + SetWatches sw = new SetWatches(lastZxid, + dataWatchesBatch, + existWatchesBatch, + childWatchesBatch); {code} The lastZxid should be the *same* for all SetWatches packets. Given it can change while you are looping (because a watch could have fired, and triggered a read), that would cause the client to claim that it's further ahead than it really is. So we should save lastZxid before we start sending the SetWatches packets.
22. **body:** Small nit, in: {code} + String path = zk1.create(pathBase + "/ch-" + i, null, Ids.OPEN_ACL_UNSAFE, + CreateMode.PERSISTENT); + paths.add(path); {code} given that you are saving the path, you could just use CreateMode.PERSISTENT_SEQUENTIAL (to get a unique path) and no need to + i. Other than that and the lastZxid issue, I think it looks good. Thanks!
    label: code-design
23. Ah, good catch. I'll fix that up.
24. Okay, these should address the zxid issue and the PERSISTENT_SEQUENTIAL issue.
25. -1 overall. Here are the results of testing the latest attachment http://issues.apache.org/jira/secure/attachment/12739332/ZOOKEEPER-706-branch-34.patch against trunk revision 1685167. +1 @author. The patch does not contain any @author tags. +1 tests included. The patch appears to include 3 new or modified tests. -1 patch. The patch command could not apply the patch. Console output: https://builds.apache.org/job/PreCommit-ZOOKEEPER-Build/2767//console This message is automatically generated.
26. Merged: trunk https://github.com/apache/zookeeper/commit/aea212fb4cea300dd1eb213541f60352246a74d6 3.5 https://github.com/apache/zookeeper/commit/66532c2b43daf96e769a8ed6cf217808d2c621ec 3.4 https://github.com/apache/zookeeper/commit/05f17a04e5ef261abae19b08ef138f62febfb188 thanks [~cthunes]!
27. Happy to contribute and thanks for the feedback and quick turn around!
28. FAILURE: Integrated in ZooKeeper-trunk #2725 (See [https://builds.apache.org/job/ZooKeeper-trunk/2725/]) ZOOKEEPER-706: Large numbers of watches can cause session re-establishment to fail (Chris Thunes via rgs) (rgs: http://svn.apache.org/viewcvs.cgi/?root=Apache-SVN&view=rev&rev=1685200) * /zookeeper/trunk/CHANGES.txt * /zookeeper/trunk/src/java/main/org/apache/zookeeper/ClientCnxn.java * /zookeeper/trunk/src/java/test/org/apache/zookeeper/test/DisconnectedWatcherTest.java
29. @Raul: Is it safe to backport this to 3.4.6?
30. [~janmejay]: should be, but I am hoping to have an RC for 3.4.7 out this week (waiting on two patches as well...)
31. Hi [~michim], [~fpj] please help to add [~cthunes] to the contributors list and assign the issue to him. Thanks!
32. [~rakeshr] done.
33. Looks like patch has fix only for JAVA client. This has not been fixed for C client. I would like to submit a fix for C client. Please confirm.
34. I agree that the C client is still vulnerable to this - please do put up a patch if you have the time.