Item 297
git_comments:

git_commits:

1. **summary:** This closes #1592
   message: This closes #1592

github_issues:

github_issues_comments:

github_pulls:

1. **title:** NO-JIRA improve MQTT protocol logging
   body:

github_pulls_comments:

1. Some comments : - You can avoid to show will information if the isWillFlag is false so there aren't empty fields to show - avoid to show the will message payload as you are avoiding to show the publish payload. I guess you are using an old version of the MQTT Netty codec where will payload is a String but in the newer version it's a byte array like publish payload - having the CONNACK packet as well with the result code and if there is a session is useful - having PUBACK, PUBREL, PUBCOMP are useful as well with related packet-id
2. I just added a commit and rebased. Changes include: - Avoiding empty fields related to will in CONNECT messages - Adding CONNACK details - System property for enabling logging for UTF8 publish payload
3. What special handling does the UNSUBACK packet need? It should already be logged with the packet-id by virtue of line 164. It has no payload to deal with from what I can tell in the spec.
4. @jbertram sure right. No other needs for SUBACK
5. I just pushed the latest iteration with the suggestion from @franz1981 about using a new method as well as removing the system property and enforcing a hard limit of 256 characters when printing the payload.

github_pulls_reviews:

1. rather than commented out should this just be removed.
2. ditto as per other comment. "rather than commented out should this just be removed."
3. It should shows the packet-id as well
4. From a security perspective I think that's a fair point.
5. Ditto as above
6. Ditto as above
7. I disagree that this should be removed as it can serve as a really convenient way to look at the payload granted they're UTF-8 encoded strings. The only reason it's commented out is because the MQTT spec has no rules on the format of the payload. It's just an array of bytes. However, if you happen to be working with UTF-8 encoded string then this might prove helpful.
8. Yes I agree that is useful if you are using UTF8 encoded string as payload. Is it possible having this logging line as configurable (disabled by default) so that people using Strings can enable it for showing the payload.
9. If that's the case would configuration to turn it on / off be better than requiring a complete rebuild in order to use them once uncommented ? Sidebar is there a gating mechanism on the logger being used to check if it's even enabled, all those append calls for something that might not actually be turned on seems less than ideal.
10. The method starts with: if (logger.isTraceEnabled()) So none of this will be executed if TRACE (or equivalent) isn't turned on in the logging configuration.
11. And there is no way for making one of them configurable as well ? For example printing the PUBLISH payload ?
12. Packet-ID is already logged above before the switch statement starts.
13. @jbertram sorry you are right !

14. Packet-ID is already logged above before the switch statement starts.
15. Packet-ID is already logged above before the switch statement starts.
16. As above you are right sorry !
17. As above you are right sorry !
18. My only thought here is to use something really basic like a system property. I don't think this is worth changing the configuration schema to support a new option in broker.xml.
19. name and version are related to the protocol (i.e. "MQTT" and protocol level which is 4 for 3.1.1). @jbertram what do you think to log something like, protocol=(name, version) ? without having two distinct fields ?
20. @jbertram In order to avoid the JVM to compile `logMessage` with a big method (killing futher optimisations of the caller) I suggest to move the log message composition in a separate method (eg `private static void traceMessage(...)`) and use a pattern of calls like this: ``` public static void logMessage(MQTTSessionState state, MqttMessage message, boolean inbound) { if (logger.isTraceEnabled()) { traceMessage(state, message, inbound); } } ``` This will let the JVM inline the `logMessage` call when is not needed.
21. @jbertram I would create the logMessage @franz1981 is suggesting, but my concern here is not much about perf (since this method is already only called upon logging).. but out of making the code concise. Just a suggestion though.
22. this is about tracing.. why don't you just always log the UTF? if you don't need that level of information, just take trace out. of you could do something like.. have this whole thing at debug... and if trace, throw the body on log.trace. The system property is a bit odd here.
23. My 2 cents. The PUBLISH payload has a maximum size of 256 MB. I could have a use case where I need tracing packets but not printing 256 MB of bytes on the console. Use case where could be a problem in the messages flow that is not dependent on the payload. In this moment using just TRACE enabled or not it's just about having or not the logging. Maybe we could use DEBUG for normal packets logging and then TRACE for printing the payload as well.
24. To keep this as simple as possible I'll get rid of the system property and always log the payload as UTF-8, but also always limit the output to a certain number of characters so we don't get hundreds of megs of data (potentially).
25. We do that in 5.x for things like logs on STOMP messages where you don't want to dump the full content portion of the frame but a small bit can help, I can't recall what the default is but something like 60 chars or the like is used. You end up with a content string of "ABCD...XYZ"
26. So for people who need to have the entire payload because it's in a reasonable range size, they need to use Wireshark ?
27. nice +1

jira_issues:

jira_issues_comments: