Item 297
git_comments:

1. * * Message bodies are written to XML as one or more Base64 encoded CDATA elements. Some parser implementations won't * read an entire CDATA element at once (e.g. Woodstox) so it's possible that multiple CDATA/CHARACTERS events need * to be combined to reconstruct the Base64 encoded string. You can't decode bits and pieces of each CDATA. Each * CDATA has to be decoded in its entirety. * * @param processor used to deal with the decoded CDATA elements * @throws IOException * @throws XMLStreamException
2. when we hit a whitespace CHARACTERS event we know that the entire CDATA is complete so decode, pass back to * the processor, and reset the cdata for the next event(s)

git_commits:

1. **summary:** This closes #791
   message: This closes #791

github_issues:

github_issues_comments:

github_pulls:

1. **title:** ARTEMIS-747 multiple CDATA events on import fails
   body:
2. **title:** ARTEMIS-747 multiple CDATA events on import fails
   body:
3. **title:** ARTEMIS-747 multiple CDATA events on import fails
   body:
   label: code-design
4. **title:** ARTEMIS-747 multiple CDATA events on import fails
   body:
5. **title:** ARTEMIS-747 multiple CDATA events on import fails
   body:
6. **title:** ARTEMIS-747 multiple CDATA events on import fails
   body:

github_pulls_comments:

github_pulls_reviews:

1. use if (log.isDebug()) log.debug(...) And get a logger for this class. This is change I have made in all the longer not long ago.
2. **body:** JDK's STAX implementation reports CDATA sections also as XMLStreamConstants.CHARACTERS (instead of XMLStreamConstants.CDATA like Woodstox does) by default. So this condition needs to additionally check that the text contains white spaces only. Probably reader.isWhiteSpace() should do.
   label: code-design
3. Although the problem would only occur if JDK's STAX implementation happened to fragment the the CDATA, which it (probably?) doesn't do. So it may be OK how it is.
4. I think isWhiteSpace() is better. Good idea!

jira_issues:

1. **summary:** Multiple CDATA events during import fails
   description: Message bodies are written to XML as Base64 encoded CDATA elements. Some parser implementations won't read the entire CDATA element at once (e.g. Woodstox) so it's possible for multiple CDATA events to be combined into a single Base64 encoded string. You can't decode bits and pieces of each CDATA. Each CDATA has to be decoded in its entirety. The current importer doesn't deal with this properly.

2. **summary:** Multiple CDATA events during import fails
   description: Message bodies are written to XML as Base64 encoded CDATA elements. Some parser implementations won't read the entire CDATA element at once (e.g. Woodstox) so it's possible for multiple CDATA events to be combined into a single Base64 encoded string. You can't decode bits and pieces of each CDATA. Each CDATA has to be decoded in its entirety. The current importer doesn't deal with this properly.
3. **summary:** Multiple CDATA events during import fails
   description: Message bodies are written to XML as Base64 encoded CDATA elements. Some parser implementations won't read the entire CDATA element at once (e.g. Woodstox) so it's possible for multiple CDATA events to be combined into a single Base64 encoded string. You can't decode bits and pieces of each CDATA. Each CDATA has to be decoded in its entirety. The current importer doesn't deal with this properly.

jira_issues_comments:

1. GitHub user jbertram opened a pull request: https://github.com/apache/activemq-artemis/pull/791 ARTEMIS-747 multiple CDATA events on import fails You can merge this pull request into a Git repository by running: $ git pull https://github.com/jbertram/activemq-artemis ARTEMIS-747 Alternatively you can review and apply these changes as the patch at: https://github.com/apache/activemq-artemis/pull/791.patch To close this pull request, make a commit to your master/trunk branch with (at least) the following in the commit message: This closes #791 ---- commit 5838b2c11c0776303f912e4738d6ac4448d22988 Author: jbertram <jbertram@apache.com> Date: 2016-09-16T15:25:08Z ARTEMIS-747 multiple CDATA events on import fails ----
2. Github user clebertsuconic commented on a diff in the pull request: https://github.com/apache/activemq-artemis/pull/791#discussion_r80174580 --- Diff: artemis-cli/src/main/java/org/apache/activemq/artemis/cli/commands/tools/XmlDataImporter.java --- @@ -444,33 +444,59 @@ private void processMessageBody(Message message) throws XMLStreamException, IOEx } } reader.next(); + ActiveMQServerLogger.LOGGER.debug("XMLStreamReader impl: " + reader); --- End diff -- use if (log.isDebug()) log.debug(...) And get a logger for this class. This is change I have made in all the longer not long ago.
3. Github user TomasHofman commented on a diff in the pull request: https://github.com/apache/activemq-artemis/pull/791#discussion_r80201908 --- Diff: artemis-cli/src/main/java/org/apache/activemq/artemis/cli/commands/tools/XmlDataImporter.java --- @@ -444,33 +444,59 @@ private void processMessageBody(Message message) throws XMLStreamException, IOEx } } reader.next(); + ActiveMQServerLogger.LOGGER.debug("XMLStreamReader impl: " + reader); if (isLarge) { tempFileName = UUID.randomUUID().toString() + ".tmp"; ActiveMQServerLogger.LOGGER.debug("Creating temp file " + tempFileName + " for large message."); try (OutputStream out = new FileOutputStream(tempFileName)) { - while (reader.hasNext()) { - if (reader.getEventType() == XMLStreamConstants.END_ELEMENT) { - break; - } - else { - String characters = new String(reader.getTextCharacters(), reader.getTextStart(), reader.getTextLength()); - String trimmedCharacters = characters.trim(); - if (trimmedCharacters.length() > 0) { // this will skip "indentation" characters - byte[] data = decode(trimmedCharacters); - out.write(data); - } - } - reader.next(); - } + getMessageBodyBytes(new MessageBodyBytesProcessor() { + @Override + public void processBodyBytes(byte[] bytes) throws IOException { + out.write(bytes); + } + }); } FileInputStream fileInputStream = new FileInputStream(tempFileName); BufferedInputStream bufferedInput = new BufferedInputStream(fileInputStream); ((ClientMessage) message).setBodyInputStream(bufferedInput); } else { - reader.next(); // step past the "indentation" characters to get to the CDATA with the message body - String characters = new String(reader.getTextCharacters(), reader.getTextStart(), reader.getTextLength()); - message.getBodyBuffer().writeBytes(decode(characters.trim())); + getMessageBodyBytes(new MessageBodyBytesProcessor() { + @Override + public void processBodyBytes(byte[] bytes) throws IOException { + message.getBodyBuffer().writeBytes(bytes); + } + }); + } + } + + /** + * Message bodies are written to XML as Base64 encoded CDATA elements. Some parser implementations won't read the + * entire CDATA element at once (e.g. Woodstox) so it's possible for multiple CDATA events to be combined into a + * single Base64 encoded string. You can't decode bits and pieces of each CDATA. Each CDATA has to be decoded in + * its entirety. + * + * @param processor used to deal with the decoded CDATA elements + * @throws IOException + * @throws XMLStreamException + */ + private

void getMessageBodyBytes(MessageBodyBytesProcessor processor) throws IOException, XMLStreamException { + int currentEventType; + StringBuilder cdata = new StringBuilder(); + while (reader.hasNext()) { + currentEventType = reader.getEventType(); + if (currentEventType == XMLStreamConstants.END_ELEMENT) { + break; + } + // when we hit a CHARACTERS event we know that the entire CDATA is complete so decode and pass back to the processor + else if (currentEventType == XMLStreamConstants.CHARACTERS && cdata.length() > 0) { --- End diff -- JDK's STAX implementation reports CDATA sections also as XMLStreamConstants.CHARACTERS (instead of XMLStreamConstants.CDATA like Woodstox do) by default. So this condition needs to check that the text contains white spaces only. Probably reader.isWhiteSpace() should do.

4. Github user TomasHofman commented on a diff in the pull request: https://github.com/apache/activemq-artemis/pull/791#discussion_r80204402 --- Diff: artemis-cli/src/main/java/org/apache/activemq/artemis/cli/commands/tools/XmlDataImporter.java --- @@ -444,33 +444,59 @@ private void processMessageBody(Message message) throws XMLStreamException, IOEx } } reader.next(); + ActiveMQServerLogger.LOGGER.debug("XMLStreamReader impl: " + reader); if (isLarge) { tempFileName = UUID.randomUUID().toString() + ".tmp"; ActiveMQServerLogger.LOGGER.debug("Creating temp file " + tempFileName + " for large message."); try (OutputStream out = new FileOutputStream(tempFileName)) { - while (reader.hasNext()) { - if (reader.getEventType() == XMLStreamConstants.END_ELEMENT) { - break; - } - else { - String characters = new String(reader.getTextCharacters(), reader.getTextStart(), reader.getTextLength()); - String trimmedCharacters = characters.trim(); - if (trimmedCharacters.length() > 0) { // this will skip "indentation" characters - byte[] data = decode(trimmedCharacters); - out.write(data); - } - } - reader.next(); - } + getMessageBodyBytes(new MessageBodyBytesProcessor() { + @Override + public void processBodyBytes(byte[] bytes) throws IOException { + out.write(bytes); + } + }); } FileInputStream fileInputStream = new FileInputStream(tempFileName); BufferedInputStream bufferedInput = new BufferedInputStream(fileInputStream); ((ClientMessage) message).setBodyInputStream(bufferedInput); } else { - reader.next(); // step past the "indentation" characters to get to the CDATA with the message body - String characters = new String(reader.getTextCharacters(), reader.getTextStart(), reader.getTextLength()); - message.getBodyBuffer().writeBytes(decode(characters.trim())); + getMessageBodyBytes(new MessageBodyBytesProcessor() { + @Override + public void processBodyBytes(byte[] bytes) throws IOException { + message.getBodyBuffer().writeBytes(bytes); + } + }); + } + } + + /** + * Message bodies are written to XML as Base64 encoded CDATA elements. Some parser implementations won't read the + * entire CDATA element at once (e.g. Woodstox) so it's possible for multiple CDATA events to be combined into a + * single Base64 encoded string. You can't decode bits and pieces of each CDATA. Each CDATA has to be decoded in + * its entirety. + * + * @param processor used to deal with the decoded CDATA elements + * @throws IOException + * @throws XMLStreamException + */ + private void getMessageBodyBytes(MessageBodyBytesProcessor processor) throws IOException, XMLStreamException { + int currentEventType; + StringBuilder cdata = new StringBuilder(); + while (reader.hasNext()) { + currentEventType = reader.getEventType(); + if (currentEventType == XMLStreamConstants.END_ELEMENT) { + break; + } + // when we hit a CHARACTERS event we know that the entire CDATA is complete so decode and pass back to the processor + else if (currentEventType == XMLStreamConstants.CHARACTERS && cdata.length() > 0) { --- End diff -- Although the problem would only occur if JDK's STAX implementation happened to fragment the the CDATA, which it (probably?) doesn't do. So it may be OK how it is.

5. Github user jbertram commented on a diff in the pull request: https://github.com/apache/activemq-artemis/pull/791#discussion_r80250306 --- Diff: artemis-cli/src/main/java/org/apache/activemq/artemis/cli/commands/tools/XmlDataImporter.java --- @@ -444,33 +444,59 @@ private void processMessageBody(Message message) throws XMLStreamException, IOEx } } reader.next(); + ActiveMQServerLogger.LOGGER.debug("XMLStreamReader impl: " + reader); if (isLarge) { tempFileName = UUID.randomUUID().toString() + ".tmp"; ActiveMQServerLogger.LOGGER.debug("Creating temp file " + tempFileName + " for large message."); try (OutputStream out = new FileOutputStream(tempFileName)) { - while (reader.hasNext()) { - if (reader.getEventType() == XMLStreamConstants.END_ELEMENT) { - break; - } - else { - String characters = new String(reader.getTextCharacters(), reader.getTextStart(), reader.getTextLength()); - String trimmedCharacters = characters.trim(); - if (trimmedCharacters.length() > 0) { // this will skip "indentation" characters - byte[] data = decode(trimmedCharacters); - out.write(data); - } - } - reader.next(); - } + getMessageBodyBytes(new MessageBodyBytesProcessor() { + @Override + public

void processBodyBytes(byte[] bytes) throws IOException { + out.write(bytes); + } + }); } FileInputStream fileInputStream = new FileInputStream(tempFileName); BufferedInputStream bufferedInput = new BufferedInputStream(fileInputStream); ((ClientMessage) message).setBodyInputStream(bufferedInput); } else { - reader.next(); // step past the "indentation" characters to get to the CDATA with the message body - String characters = new String(reader.getTextCharacters(), reader.getTextStart(), reader.getTextLength()); - message.getBodyBuffer().writeBytes(decode(characters.trim())); + getMessageBodyBytes(new MessageBodyBytesProcessor() { + @Override + public void processBodyBytes(byte[] bytes) throws IOException { + message.getBodyBuffer().writeBytes(bytes); + } + }); + } + } + + /** + * Message bodies are written to XML as Base64 encoded CDATA elements. Some parser implementations won't read the + * entire CDATA element at once (e.g. Woodstox) so it's possible for multiple CDATA events to be combined into a + * single Base64 encoded string. You can't decode bits and pieces of each CDATA. Each CDATA has to be decoded in + * its entirety. + * + * @param processor used to deal with the decoded CDATA elements + * @throws IOException + * @throws XMLStreamException + */ + private void getMessageBodyBytes(MessageBodyBytesProcessor processor) throws IOException, XMLStreamException { + int currentEventType; + StringBuilder cdata = new StringBuilder(); + while (reader.hasNext()) { + currentEventType = reader.getEventType(); + if (currentEventType == XMLStreamConstants.END_ELEMENT) { + break; + } + // when we hit a CHARACTERS event we know that the entire CDATA is complete so decode and pass back to the processor + else if (currentEventType == XMLStreamConstants.CHARACTERS && cdata.length() > 0) { --- End diff -- I think isWhiteSpace() is better. Good idea!

6. Commit ea552a1f88c1cbc1a3e9352a936c428db5af9527 in activemq-artemis's branch refs/heads/master from jbertram [https://git-wip-us.apache.org/repos/asf?p=activemq-artemis.git;h=ea552a1] ARTEMIS-747 multiple CDATA events on import fails
7. Github user asfgit closed the pull request at: https://github.com/apache/activemq-artemis/pull/791