Fixes memory leaks in IoT #1175

New issue

Closed jkennington wants to merge 1 commit into `aws-amplify:master` from `jkennington:bugfix/iot-memory-leaks`

Conversation 6 Commits 1 Checks 0 Files changed 5 +60 −2

jkennington commented on 15 Jan 2019 • edited — Contributor

Similar to earlier PR with another memory leak fixed and target most recent SDK version.

After removing an `AWSIoTDataManager`, the `AWSIoTDataManager`, `AWSIoTMQTTClient`, `AWSURLSessionManager` and other related entities were leaked.

This fixes a retain cycle between the `AWSIoTMQTTClient` where it held a strong reference to the `associatedObject` the `AWSIoTDataManager` which was holding a strong reference to the `mqttClient`. This makes the `associatedObject` reference weak.

If an `AWSIoTDataManager` was disconnected before the `connectionAgeInSeconds` exceeded the `minimumConnectionTime` the `connectionAgeTimer` was never invalidated and held a strong reference to the `AWSIoTMQTTClient` (e.g. `self` passed to the timer on creation of the timer). This fixes the issue by issuing an `invalidate` to the timer on the correct thread in the event the user requests a disconnect before reaching the `minimumConnectionTime`.

The `AWSURLSessionManager` creates and passes `self` to an `NSURLSession` but never invalidated the session to release the strong reference to `self`. `AWSNetworking` could be released but the `AWSURLSessionManager` leaked being held by the `NSURLSession`. This is fixed by having the `AWSNetworking` let the `AWSURLSessionManager` know it needs to invalidate the session when it is being released, as none of them should be needed after the `AWSNetworking` creating them is released.

The `reconnectTimer` was invalidated on whatever thread `disconnect` was called on rather than the `reconnectThread` causing a memory leak; this is fixed by having the `reconnectTimer` invalidated on the correct thread.

Fixes memory leaks in IoT 2a0ed2d

jkennington mentioned this pull request on 15 Jan 2019

Fix memory leaks for IoT #1037 Closed

jkennington reviewed on 15 Jan 2019 View changes

AWSIoT/Internal/AWSIoTMQTTClient.m

```
...  ...  @@ -639,6 +672,12 @@ - (void)openStreams:(id)sender
639  672        [defaultRunLoopTimer invalidate];
640  673
641  674        if (!self.runLoopShouldContinue ) {
     675  +
     676  +         if (self.connectionAgeTimer != nil ) {
```

jkennington on 15 Jan 2019 — Author — Contributor

@cbommas, your earlier suggestion to simplify this was correct as this method will be called already on the correct thread.

cbommas on 29 Jan 2019 — Contributor

frankmuellr added `iot` `pull request` labels on 15 Jan 2019

frankmuellr requested a review from **cbommas** on 15 Jan 2019

frankmuellr assigned **cbommas** on 15 Jan 2019

jkennington reviewed on 15 Jan 2019 View changes

AWSIoT/Internal/AWSIoTMQTTClient.m

```
     579  +         // NOTE: This does not work as intended. The reconnection attempt is always done on the reconnec
     580  +         // but the timer needs to be invalidated on the streams thread to ensure memory is released.
     581  +         // However, as currently implemented a switch to the streams thread here to clean this up is alw
     582  +         // invalidation by other events which are are already being processed on the streams thread.
550  583          [self.connectionAgeTimer invalidate];
```

jkennington on 15 Jan 2019 — Author — Contributor

As noted this doesn't really work but other code avoids a memory leak. Ideally, this would have code to switch threads to clean up correctly but I left this alone for now as by the time a thread switch got control another event had already done the cleanup. I can update PR to have the thread switch if desired.

cbommas on 29 Jan 2019 — Contributor

For this change, rather than doing a thread switch, lets just remove this code, as it is handled in one of the other events.

cbommas commented on 29 Jan 2019 • edited — Contributor

@jkennington

Can we break this PR up into two separate ones? There are changes to the IOT code and the Core code.

The challenge I have with AWSCore code changes is that we had to make the change to not invalidate the session due to crashes (see #913). We will have to dive deeper on this.

For the IOT changes, I will be testing them and prepping for release, pending the test results.

jkennington commented on 30 Jan 2019 — Contributor — Author

Have split into #1202 and #1203. Closing this PR.

jkennington closed this on 30 Jan 2019

Reviewers
cbommas

Assignees
cbommas

Labels
iot

Projects
None yet

Milestone
No milestone

Linked issues
Successfully merging this pull request may close these issues.
None yet

3 participants