COMP3310/6331 2020 – Tute/Lab #5

#	
1	**Exercises:** With the marks out for assignment 1, as well as the guide sheet to what was being looked for, take the chance to discuss it with the tutor. They marked your assignment, so if you have any questions about your mark you can raise it with them. By now your assignment 2 code should be making good progress. If not, ask for help. If it is running, check your outputs against others in the group. There have been a few questions asked in the forum about interpretation of what to report, or how to find the information you may need. RFC1945 is your go-to document, and have a good look at Section 10 (server headers). Next week you'll get to experiment with some MQTT clients and brokers, in preparation for the third assignment.
2	**Review questions** Realtime & IoT 1. Look at the curve on slide 9 of T5c (Realtime). Make sure you understand what the graph is showing you, in terms of delivery-fraction against time. Compare this against the 'ideal' situation in slide 8 Try to explain the various components shown: ○ What causes the 'latency' on the left? Transmission delay between sender/receiver, can't avoid it. ○ What causes the long tail at the far right? Packets that took the long way round or got held up (in queues) the most along the way ○ What would cause jitter to become very large? It's measuring the delays for a particular sample of packets, so they must have experienced a wide diversity of delays - perhaps the sender/receiver path has changed a lot during this sample, and/or the routers along the way got quiet/busy for a period. 2. Look at the diagram on slide 14 of T5c. ○ Why is packet 4 not able to be played out, since it was received? How could we fix that? What problem does that cause? The receive buffer was not long enough (in time) to capture it before the application had to play it out. We could make the receive buffer longer, but that will make the delay between sending and playout longer as well - so the experience won't be as realtime. 3. Why is retransmission (except in multicast) generally a bad idea? Why is it ok in multicast? ○ It takes a lot of time, to detect the missing packet, to send a request back to the source, in the hope it still has a copy of the missing packet, and receive the missing packet (hopefully, on the second go!). In multicast, the server sending you the retransmission may be very close, and so the added delay is minimised. 4. Why are the performance problems reduced in streaming applications?

- The problems still exist, but it's one-way traffic, so you don't sense the delay that is occurring. That means you can have much larger buffers, use more reliable transport protocols, etc.

5. Do you own any IoT devices (an internet-enabled… "thing" that isn't a computer? Do you know where its packets go? Have you got any good horror stories about IoT?Please share!

6. What are the main network and device issues for IoT? Why is it different to other devices connecting to the internet?
   - Slide 8 of T5d (IoT). Scale, Power, Networking, Timeliness and Reliability, which each create design implications, discussed on subsequent slides.

7. As an application model, how is PubSub different to Client/Server? Why is it better suited to large-scale IoT applications?
   - Clients ask Servers for data they have, a direct one-to-one relationship. A PubSub model separates the data source that is publishing information from the consumer needing it through a mediating server/broker.
   - It provides better scalability (N producers + M consumers each talking with a server, rather than N*M direct relationships). It also takes load off the producers, they only need to send the data once (in principle) to a specific server, rather than maintaining relationships with an evolving pool of consumers. It also implicitly pushes the analysis/interpretation of the data to the consumers that care about it, again reducing load on the producers (and brokers)

8. Why are MQTT packets so compact and simple? (compared to HTTP say)
   - All about performance/load. Short messages don't consume bandwidth, and simplicity avoids spending too much CPU time forming packets.

9. Why does MQTT offer multiple levels of QoS (quality of service)? Why is there separate, potentially different, QoS on both sides of the broker (producer->broker, broker->consumer)?
   - Grudgingly, to fit in with the needs of applications. Some really care that the messages arrive reliably, while others are happy to get the occasional update, and can wait if they miss it.
   - The overall QoS is only as good as the weakest link. If a producer has data that is known to be really valuable to someone (but perhaps not everyone), they would use a high QoS. The consumers can then each choose individually whether they really need the data (and use a high QoS), or just the occasional update and use a low QoS (as above). It provides greater flexibility for the consumption of data.
10. In what circumstances is it useful for a server to 'retain' an MQTT-published message?
    - Where the source may only publish intermittently, where the consumers may join at any time, or be unreliably connected. It ensures that when things start up, the current state of things is quickly learnt.