SUTD 2022 Term 7 50.046 Homework 1

Hand-out: 27 Jan

Hand-in: 10 Feb (Thursday) 23:59

Name: James Raphael Tiovalen Student ID: 1004555

Question 1. [7pt] Regarding the general design principle of "Design for offline behavior", please briefly discuss about how that principle can affect the design decision at each of the seven layers (as identified in the AWS well-architected framework)? For each layer, you are expected to highlight one or two key points, each in one sentence.

Answer: The aforementioned design principle can affect each of the seven layers as such:

- Design and Manufacturing Layer: At the manufacturing stage, the quality of hardware related to network connectivity needs to be rigorously checked, and multiple methods of connecting to the cloud should be provided to each device (to serve as backup) and should be easily accessible from the software side.
- Edge Layer: The edge devices need to be able to properly detect if it is currently offline and account for the data collected during the offline period properly, while still maintaining a proper tradeoff between processing power, storage capacity, and network speed (by implementing some kind of aggregator function such as sum or average, depending on the business needs). The software should also continuously attempt to reconnect to the cloud whenever it is offline.
- Provisioning Layer: The provisioning layer needs to ensure that there are backup methods of attempting to go back online once the edge devices detect that they are offline. It also needs to provide an easy workflow for developers to access the devices for maintenance during offline periods and decommission them if they are not servicable anymore.
- Communication Layer: The communication layer should monitor the status of the devices periodically and check if they are offline, and it should be able to timeout and relay the status (both online and offline) to the cloud properly. It should also be able to handle aggregated data packets if necessary, which might be of a different format compared to normal packets.
- Ingestion Layer: Similar to the communication layer, the ingestion layer should be able to handle status data, actual payload data, as well as aggregated payload data in the case whereby some devices go offline.
- Analytics Layer: The analytics layer should be able to process different types of data payload packets properly.
- Application Layer: The application needs to be able to "fail gracefully" (instead of crashing or encountering errors), which can be done by indicating the specific devices that are currently offline for extended periods of time in the user-facing UI and whether

they might need further manual inspection. It should also be able to store the different data payload types properly.

Question 2. [6pt] When accessing the website http://www.hivemq.com/demos/websocket-client/ to test the MQTT client, we capture the following packet via Wireshark.

```
> Frame 1048: 92 bytes on wire (736 bits), 92 bytes captured (736 bits) on interface en0, id 0
> Ethernet II, Src: Apple_4e:26:c8 (f8:ff:c2:4e:26:c8), Dst: Alcatel-_c4:46:ae (e8:e7:32:c4:46:ae)
> Internet Protocol Version 4, Src: 10.19.242.45, Dst: 35.157.158.119
> Transmission Control Protocol, Src Port: 54447, Dst Port: 8000, Seq: 583, Ack: 164, Len: 26
> WebSocket
v Data (20 bytes)
    Data: 101200064d51497364700302003c000453555444
    [Length: 20]
```

Please answer the following questions:

1) [3pt] The 20 bytes data field being shown above corresponds to a MQTT message. By referring to https://openlabpro.com/guide/mqtt-packet-format/, could you decode which type of control packet this belongs to? Briefly explain at least three different pieces of information you can decode from this MQTT message.

Answer: The first 10 indicates that it is a CONNECT control packet command from a client to a server. Next, 12 (which is 18 in decimal) indicates the Remaining Length (in number of bytes) within the current packet in hexadecimal format, including the data in the variable header and the payload, but not including the Remaining Length byte itself. 00 06 indicates the length (in number of bytes) of the Protocol Name. 4D 51 49 73 64 70 indicates the Protocol Name used: MQIsdp (which is the name used in MQTT v3.1). 03 indicates the Protocol Level/version number, and 02 indicates the Connect Flags byte, which sets the Clean Session bit to 1 in this case. 00 3C indicates that the Keep-Alive duration is 60 seconds. 00 04 indicates the length of the actual data payload, which is the client ID in this case. 53 55 54 44 indicates the client ID represented in ASCII: "SUTD".

2) [1pt] What is the IP address of the MQTT server, as can be found from the packet?

Answer: 35.157.158.119.

3) [2pt] We learnt that MQTT uses TCP as the transport layer protocol. However, there is a WebSocket header between the TCP header and the Data. Briefly explain why WebSocket is used here.

Answer: The MQTT client demo/showcase service provided by HiveMQ utilizes WebSocket to allow a direct and full duplex communication access to the MQTT broker/server from the web application/website/webpage contained and running within a browser.

Question 3. [7pt] Consider the following MQTT setup: a group of 100 IoT devices periodically publish messages to a topic **iot/broadcastgroup**, to which they all subscribe as well. Each device publishes a small message every second. Each message results in a 100-byte packet.

The network is lossy. For each device, a message (sent in either direction) between the device and the broker can get lost with probability of p = 2% (despite of the use of TCP). Assume we use QoS level 0 for message delivery from both the publishing client to the broker, and from the broker to the subscribing client.

1) [5pt] Estimate what is the expected bandwidth usage (i.e., the sum of both incoming and outgoing traffic) for the broker? Express your answer in terms of both packet-per-second and bit-per-second (bps). You can ignore all additional packets introduced by TCP layer (e.g., the TCP handshake, ACK, etc.)

Answer: On average, the expected bandwidth usage is:

$$\begin{aligned}\text{Bandwidth} &= 0.98 \times 100 + 0.98 \times (98 \times 100) = 9702 \text{ packets/s} \\ &= 9702 \times 100 \times 8 = 7761600 \text{ bits/s}\end{aligned}$$

2) [2pt] On average, one device can receive how many messages every second?

Answer: On average, one device would receive $0.98 \times 0.98 \times 100 = 96.04$ messages per second.

3) [optional --- no point will be awarded, but you are encouraged to think about it] Now consider the use of QoS level 1 and QoS level 2 for all the messages (including the messages between publishers and MQTT server, and between MQTT server and subscribers). This will result in additional Publish ACK / Received / Release / Complete packets as well as possible retransmissions of lost packets. Remember that these additional packets can also get lost with the same probability of p = 2%. We also assume all the retransmissions are triggered correctly, e.g., with proper timeout. Please estimate the overall bandwidth requirement on the broker (in terms of packet-per-second), for both the case of using QoS level 1 and QoS level 2. Same as above, you can ignore the additional packets introduced by TCP layer (e.g., the TCP handshake, ACK, etc.) and just focus on the MQTT packets.

Answer: In this analysis, we will assume that the broker would also require the needed Publish ACK / Received / Release / Complete packets from the subscribers as well to ensure complete delivery.

For QoS Level 1:

$$\begin{aligned}\text{Bandwidth} &= 101(1 + 0.98)(100 + 2 + 0.04 + 0.0008 + \cdots) \\ &= 101 \times 1.98 \times \frac{100}{1 - 0.02} \approx 20406.12 \text{ packets/s}\end{aligned}$$

For QoS Level 2:

$$\begin{aligned}\text{Bandwidth} &= 2 \times 101(1 + 0.98)(100 + 2 + 0.04 + 0.0008 + \cdots) \\ &= 202 \times 1.98 \times \frac{100}{1 - 0.02} \approx 40812.24 \text{ packets/s}\end{aligned}$$

- END -