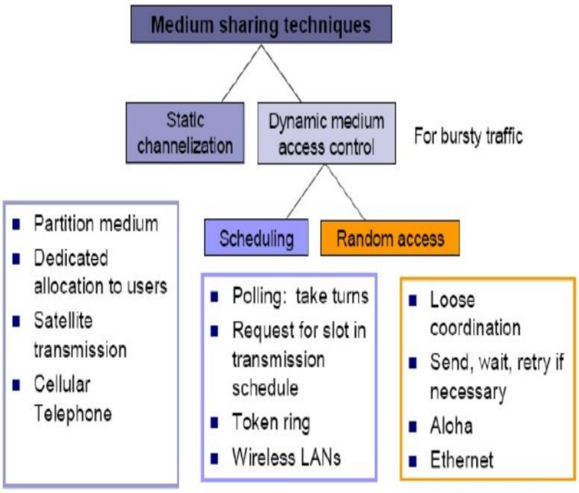
A comparison of scheduling approaches to media access control

Rachit Bhanage
Department of Internetworking
Dalhousie University
Halifax, Nova Scotia

Abstract: Media Access Control (MAC), After Physical layer the participation and role of Media access control sublayer in Data Link layer in undeniable. Developing new MAC protocols is difficult but a lot work can be done existing protocols and their mechanisms in the manner they operate and their scheduling approaches. The scheduling approaches employed are interesting and versatile depending on the requirement of their implementation. We start by briefing the most well-known and then detailing one of them.

Introduction:

1. Media Access Control techniques can be bifurcated according to following. Figure 1.1 gives an overall view of various methods.



Concentrating on the scheduling approach, its methods has its own drawbacks and benefits under certain requirements and situations. They are briefed as follows:

2. Token Ring: Mostly employed for local area network (LAN). It makes use of three frame bytes call token and a logically conceptualized ring that constitutes if workstations or servers. Standardized with **IEEE 802.5** protocol name.

3. Wireless LAN: Based on **IEEE 802.11** as Wireless Local Area Networks**.[1]** They may employee wireless distribution systems. Multiple WLAN, when connected together, can span a huge network. WLANS are used everywhere both domestically and commercially.

4. Reservation system: Stations transmit information taking turns organized into cycles of variable lengths. Reservation interval is marked with the beginning of a cycle which consists of mini-slots. They are then utilized by the stations to indicate transmission. Length of the cycle is directly proportional to the number of stations contained in system or domain. In a general reservation system, to increase utilization, Idle slots are made available by using Time-division Multiplexing.
Reservation systems can be modified to implement various scheduling techniques.

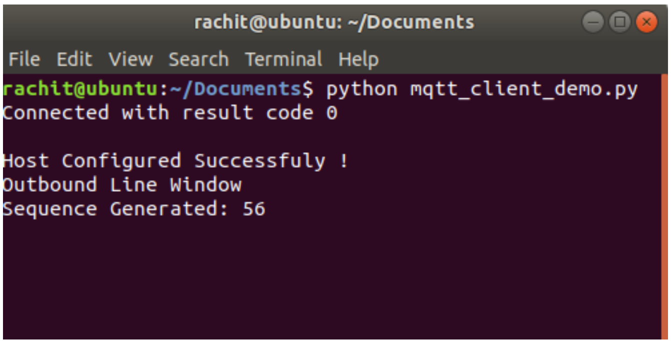
5. Polling: Stations individually take turns to access the system. Only one station is allowed to transmit at a given instance of time. After the station is done transmitting, another station is then allowed to transmit. Polling systems can be bifurcated into two types: Centralized & Distributed.

MQTT Protocol to Implement a Polling System:

6. Message Queuing Telemetry Transport is Machine-to-Machine (M2M) protocol.**[2]** It is finds its implementations in IOT (Internet of Things) for sending numerous sensor information. The information Transmitted varies in type. MQTT is a very lightweight and easy to implement making it easy to implement irrespective of the platform used. Versions are available written in multiple programming languages making it flexible to run on systems with different operating systems.

It employees a broker-subscriber mechanism which can be programmed and logically modified into an appropriate Polling system. Apart from that, the protocol offers SSL security on the transmission. Username and password can be configured. Additional security if needed can be programmed by setting required encryption.

In order, to initiate the protocol, a broker/server should be set up (resembling centralized polling system). A subscriber/client station then connected to the central server. Access medium is regardless – physical or wireless. Information transmission can be carried out in a wireless manner over different network domains(globally). When server/broker is successful setup, the protocol returns a void zero (process) message as an acknowledgment. The initialization of server is demonstrated in
Figure 1.2:

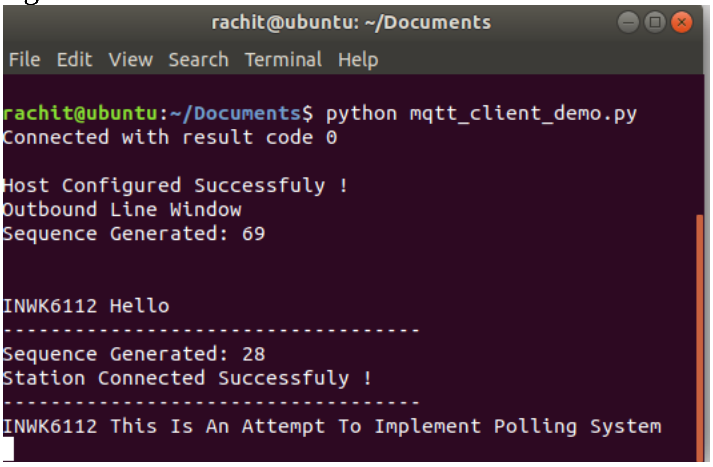

rachit@ubuntu: ~/Documents
File Edit View Search Terminal Help

```
rachit@ubuntu:~/Documents$ python mqtt_client_demo.py
Connected with result code 0

Host Configured Successfuly !
Outbound Line Window
Sequence Generated: 56
```

Figure:1.2[Broker/Server Configuration]

Once result code is obtained the later acknowledgment and transfer of data can be programmed. Similarly, when a station is successfully connected/subscribed, it acknowledges. It is demonstrated in the figure: 1.3


rachit@ubuntu: ~/Documents
File Edit View Search Terminal Help

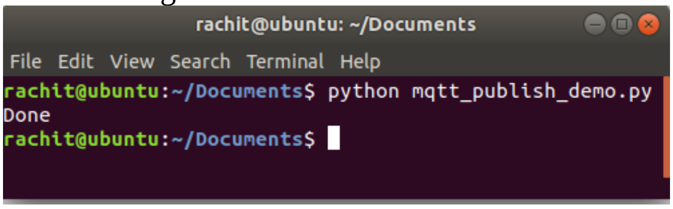
```
rachit@ubuntu:~/Documents$ python mqtt_client_demo.py
Connected with result code 0

Host Configured Successfuly !
Outbound Line Window
Sequence Generated: 69


INWK6112 Hello
-----------------------------------
Sequence Generated: 28
Station Connected Successfuly !
-----------------------------------
INWK6112 This Is An Attempt To Implement Polling System
```

Figure:1.3 [Client - Station Connection]

Upon successful connection of a client to server/broker station, it should be programmed an acknowledgment or some sort of return message mechanism as follows in figure 1.4.


rachit@ubuntu: ~/Documents
File Edit View Search Terminal Help

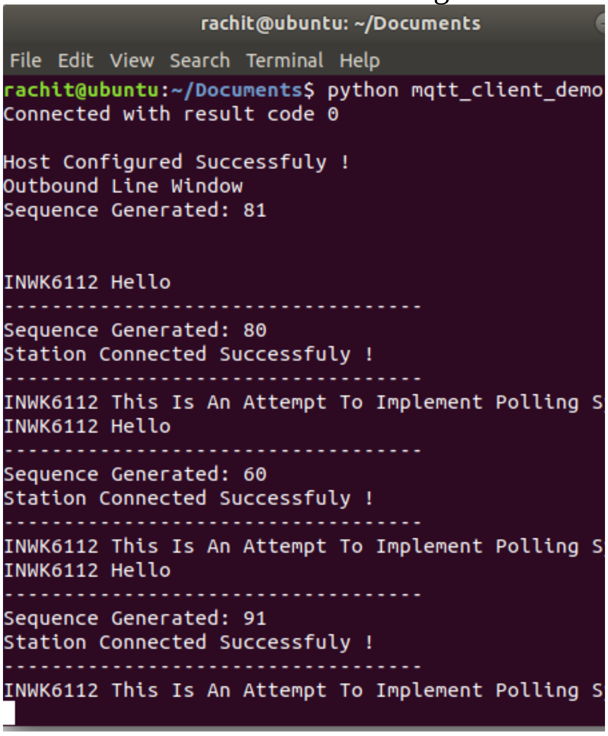
```
rachit@ubuntu:~/Documents$ python mqtt_publish_demo.py
Done
rachit@ubuntu:~/Documents$ █
```

Figure: 1.4 [Client Station Setup]

Multiple Station Connection in MQTT Polling:

7. Stations connecting to Central Broker/server station should be referred with a unique identity sequence in order to distinguish them uniquely apart from that, a unique connection string is recommended to be maintained. Here from figure 1.2 & figure 1.3, it’s **INWK6112**.

Participation of multiple client stations is simulated. It is demonstrated in figure 1.5


rachit@ubuntu: ~/Documents
File Edit View Search Terminal Help

```
rachit@ubuntu:~/Documents$ python mqtt_client_demo
Connected with result code 0

Host Configured Successfuly !
Outbound Line Window
Sequence Generated: 81


INWK6112 Hello
-----------------------------------
Sequence Generated: 80
Station Connected Successfuly !
-----------------------------------
INWK6112 This Is An Attempt To Implement Polling S
INWK6112 Hello
-----------------------------------
Sequence Generated: 60
Station Connected Successfuly !
-----------------------------------
INWK6112 This Is An Attempt To Implement Polling S
INWK6112 Hello
-----------------------------------
Sequence Generated: 91
Station Connected Successfuly !
-----------------------------------
INWK6112 This Is An Attempt To Implement Polling S
```

Figure:1.5 [Multiple Broker-Client Connections]

Represented in figure 1.5 each station connected to broker/server will initiate its own unique sequence id. Each process then generated is associated with respect to that particular id and shared among other subscribed inter-related stations. There is no particular sequence defined or manner in which this should happen but, a timestamp should be maintained making sure that only one station participates in transmission at a given interval of time.

Scalability of the MQTT Polling System:

The major benefit of using MQTT protocol in a polling system is that – it can be flexibly implemented for internet of things systems as well on regular workstations with standard processing power. It can be a good alternative for a conventional port forwarding method for exchanging information/data over internet. Apart from that, MQTT Polling is more secure as security can be developed as per needs. A polling synchronization can be set between micro-controllers connected to broker microprocessor to develop a hybrid system for real-time information exchange.

MQTT Compatibility and Release:

Name	Developed By	Language	Type	Release date
Adafruit IO	Adafruit	Ruby on Rails, Node.js	Client	To be released
M2Mqtt	eclipse	C#	Client	2017-05-20
Machine Head	Clojure Werkz	Clojure	Client	2013-11-03
moquette	Selva, Andrea	Java	Broker	2015-07-08
Paho MQTT	eclipse	Python	Client	2014-05-02

References:

[1] Eby, David W., et al. "Use, perceptions, and benefits of automotive technologies among aging drivers." *Injury epidemiology* 3.1 (2016): 28.

[2] Chang, Hui-Ling, et al. "Gateway-Assisted Retransmission for Lightweight and Reliable IoT Communications." *Sensors* 16.10 (2016): 1560.