1 Medium Access Control (MAC)

One of the problems of accessing the medium (wire, fiber, air, etc.) is who gets to do it at any given time. Imagine a room full of people, where everyone is trying to say something at once—the result being that nobody can hear anything. Similarly, if all devices on the network are trying to speak at the same time, nobody will be able to hear anything (except lots of noise). There needs to be some protocol or a procedure by which devices decide whether they should transmit or not at any given time.

There are two main approaches to medium access:

- Random Access: Each station, on its own, decides when to transmit.
- Controlled Access: There is some other scheme that determines when a station is allowed to transmit—something like a token.

1.1 Random Access

Random Access is also sometimes referred to as *contention* method. In this method, stations compete for the medium. If two stations transmit data at the same time, then a collision results—both stations then must somehow determine when to retransmit data (and note that if they wait exactly the same amount of time, then they'll just cause another collision).

In this scheme, each stations must:

- Determine when to access the medium.
- Determine what to do if the medium is busy.
- Have some way of determining whether transmission succeeded or not.
- Have some way of dealing with collisions (access conflicts with other stations).

1.2 Multiple Access (MA)

Multiple Access just refers to the fact that multiple stations have access to the medium. They can all chose to transmit at any time they wish.

1.3 ALOHA

ALOHA was one of the first MA schemes implemented (at University of Hawaii in early 1970s). It was designed for radio transmissions at 9600bps.

Many stations would access a single base station, which would then retransmit the messages to the destination. Different frequencies are used for uploading (407MHz) and downloading (413MHz).

In 'pure' ALOHA, each station:

- Each station transmits (access the medium) whenever it has data to send.
- Stations do not sense the medium (whether anyone else is sending data at the same time).
- Stations do not look for collisions (two stations transmitting at the same time).
- Reliability is achieved through acknowledgments. After transmitting, the station waits for an acknowledgment for twice the time it takes for the signal to travel the distance. If no acknowledgment arrives, then the station assumes that the message was lost, and after waiting a random amount of time, retransmits. The station will retry transmitting several times before giving up.

1.3.1 Vulnerable Time

Lets imagine that each station sends fixed length frames, and it takes T_{frame} time for the frame to be sent. Picture this situation: Station A sends a frame at time $t - T_{frame}$, at time t (but *slightly* before A's frame ends), station B starts transmitting. A collision results. Similarly, if station C decides to transmit at any time between t and $t + T_{frame}$ then a collision will result.

So, for station B's frame to be transmitted without any problems, nobody has to be transmitting from time $t - T_{frame}$ and time $t + T_{frame}$, which is:

$$\text{Vulnerable Time} = 2 \times T_{frame}$$

1.4 Slotted ALOHA

The Slotted ALOHA tries to deal with the Vulnerable Time of the pure ALOHA. Basically, the reason the vulnerable time is twice the frame size is because there is no rule when a station can start transmitting. If, for example, they all agree to start transmitting at certain intervals (or time *slots*) then station that starts transmitting at time t, doesn't need to care about anyone transmitting at $t - T_{frame}$ because nobody should still be transmitting. It only has to worry about time from t to $t + T_{frame}$, which means that in slotted ALOHA

$$\text{Vulnerable Time} = T_{frame}$$

1.5 Carrier Sense Multiple Access (CSMA)

The key idea behind CSMA is: listen before you start speaking. In other words, stations will check the medium to see if anyone else is transmitting, and if the medium is available, then they'll transmit.

1.5.1 Vulnerable Time

The vulnerable time calculation changes slightly. The length of time it takes to send a frame, T_{frame} doesn't really matter. What matters is how quickly can other stations "hear" (or sense) that any other station is speaking. What matters here is the propagation delay: T_{prop}, so

$$\text{Vulnerable Time} = T_{prop}$$

Note that T_{prop} depends on the speed of the signal in the medium, and the medium's length (or rather, the distance between the farthest stations).

1.5.2 Persistence Strategy

Persistence Strategy defines what the station does when it sense the medium and finds it busy. Here, the station can be non-persistent or persistent.

If the station is non-persistent, then after sensing the medium and finding it busy, the station waits for a random amount of time, and tries again (sense the medium again, etc.)

If the station is persistent, then after sending the medium and finding it busy, the station sense the medium again, and again, and again, until the medium is not busy. Once the medium is not busy, the station transmits with a certain probability. For example, if the probability is 1, then the station will always transmit. If the probability is 0.2, then the station will transmit only 20% of the time.

The reason to use non-1 probability for persistent strategies is if more than one stations are waiting for the medium to free up, it would be a bad idea if they all start transmitting at the same time. At least if not all of them start transmitting (with a probably of 1), then there is some chance for the transmission to start and complete successfully.

We still need to worry about collisions (two stations on opposite ends of the LAN can sense the medium, find it unoccupied, and start transmitting—causing a collision).

1.6 CSMA/CD

Carrier Sense Multiple Access / Collision Detect is a scheme where the stations try to detect collisions, and if they occur, stop transmitting (once they spot the collision), and then try to retransmit at a later time.

The question becomes, how long should a station wait before retransmitting, after it has detected a collision. If this is the first time the station is transmitting, it makes sense for it not to wait at all. If there was a collision, it makes sense for the station to wait a little bit, and if it's the third time it is transmitting, then it should wait longer, and if that doesn't work, then wait even longer. The way CSMA/CD is setup, the tractions will wait an exponential amount of time, incremented every time they try to send and get a collision.

For example, first time the station sends, and detects a collision, it will wait for

$$2^0 \times \text{maximum propagation time}$$

the second time it tries to send and detects a collision it waits for

$$2^1 \times \text{maximum propagation time}$$

In general, the time will be

$$2^N \times \text{maximum propagation time}$$

where N is the number of retry attempts starting count at 0.

1.7 CSMA/CA

Another way to deal with collisions is try to prevent them to begin with (Collision Avoidance). Basically, once the station determine that the medium is not busy, the station waits for IFG (Inter Frame Gap) amount of time, then waits for another random amount of time, and then transmits. If the station doesn't hear a reply (acknowledgment) in a certain amount of time, then it knows something is wrong, and it waits for an exponential amount of 'back-off' time (determined by number of restarts) before restarting (sensing the channel, etc.)

It is important to realize that collisions can still occur; they're just much more unlikely.

1.8 Controlled Access

Controlled Access refers to something outside the individual stations grants them rights on using the medium (transmit). An example of that scheme is the Token Ring.

In a Token Ring, a frame representing the 'token' circulates around the network, from station to station. Whatever station has the token is allowed to transmit. All stations are allowed to listen to the medium.

Note that it doesn't have to be a physical 'Ring' network. You could use a 'Bus' Topology, but still have stations send each other 'tokens'.