Advanced Network Technologies

4G

Recent Advances in Network Protocols

Dr. Wei Bao | Lecturer
School of Computer Science

THE UNIVERSITY OF SYDNEY

THE UNIVERSITY OF SYDNEY

4G/5G cellular networks

- *the* solution for wide-area mobile Internet
- widespread deployment/use:
  - more mobile-broadband-connected devices than fixed-broadband-connected devices (2019)!
  - 4G availability: 97% of time in Korea, 90% in US
- transmission rates up to 100's Mbps
- technical standards: 3rd Generation Partnership Project (3GPP)
  - wwww.3gpp.org
  - 4G: Long-Term Evolution (LTE) standard

4G/5G cellular networks

similarities to wired Internet

- edge/core distinction, but both below to same carrier
- global cellular network: a network of networks
- widespread use of protocols we've studied: HTTP, DNS, TCP, UDP, IP, etc.
- separation of data/control planes, SDN, tunneling
- interconnected to wired Internet

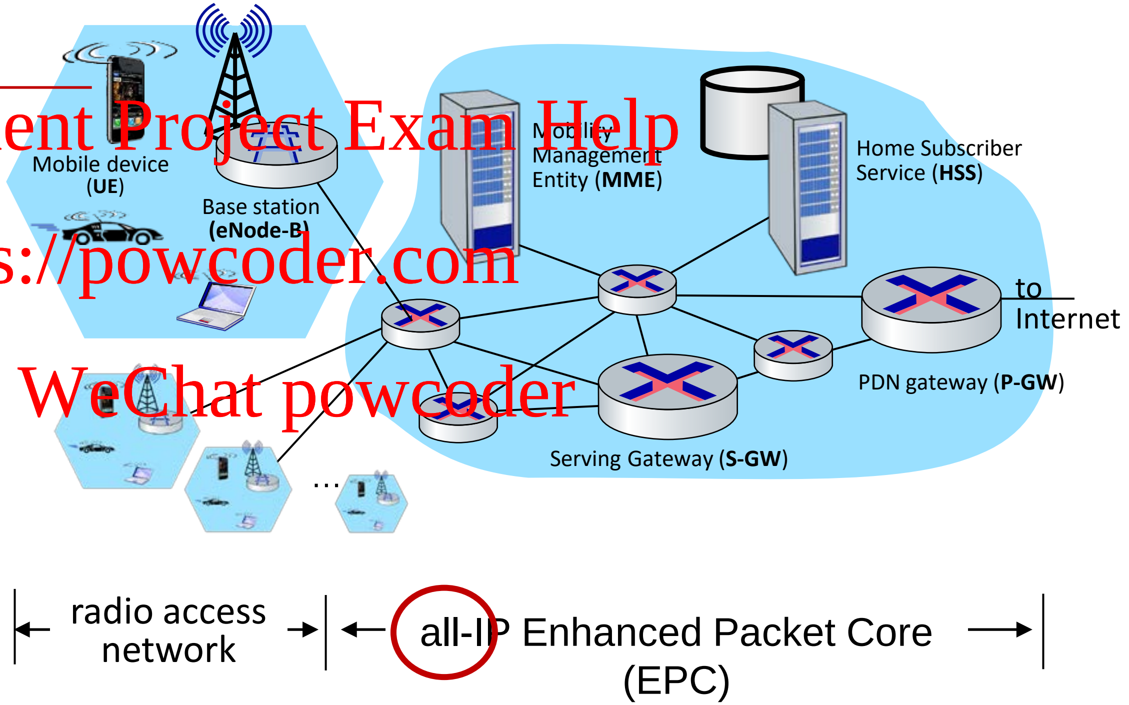
differences from wired Internet

- different wireless link layer
- mobility
- user "identity" (via SIM card)
  - subscriber identification module
- business model: users subscribe to a cellular provider
  - "home network" versus roaming on visited nets
  - global access, with authentication infrastructure, and inter-carrier settlement

Elements of 4G LTE architecture

Mobile device:

- smartphone, tablet, laptop, IoT, ... with 4G LTE radio
- 64-bit International Mobile Subscriber Identity (IMSI), stored on SIM (Subscriber Identity Module) card
- LTE jargon: User Equipment (UE)



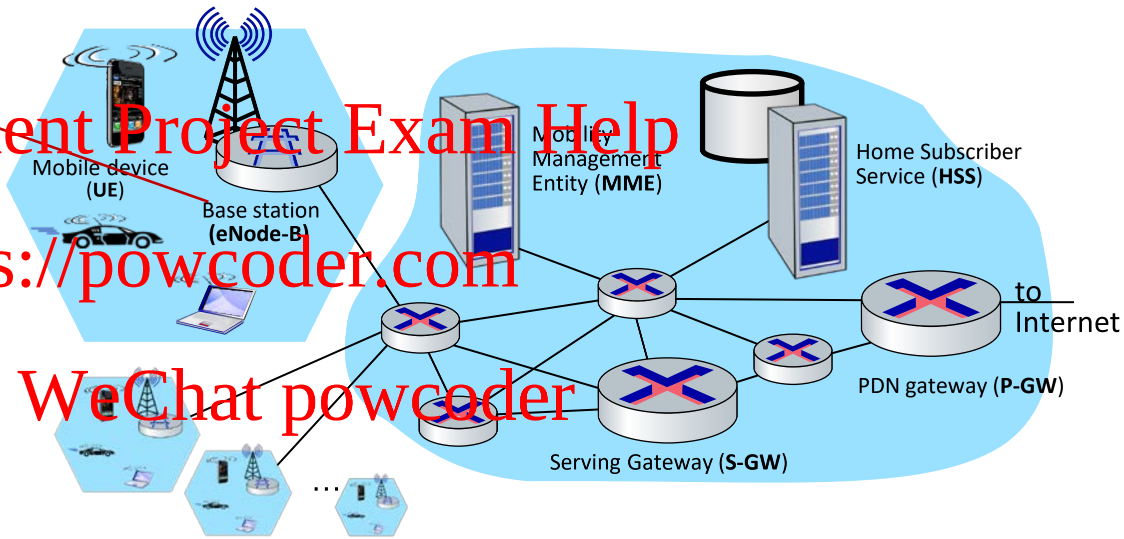
PDN: Packet Data Network

THE UNIVERSITY OF SYDNEY

Elements of 4G LTE architecture

Base station:

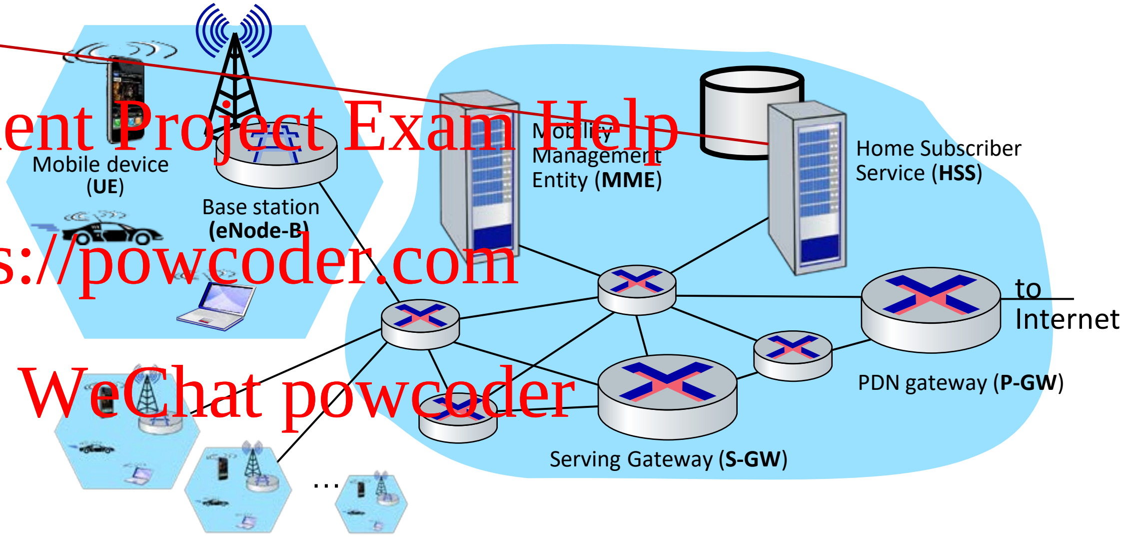
- at “edge” of carrier’s network
- manages wireless radio resources, mobile devices in its coverage area (“cell”)
- coordinates device authentication with other elements
- similar to WiFi AP but:
  - active role in user mobility
  - coordinates with nearly base stations to optimize radio use
- LTE jargon: eNode-B



Elements of 4G LTE architecture

Home Subscriber Service

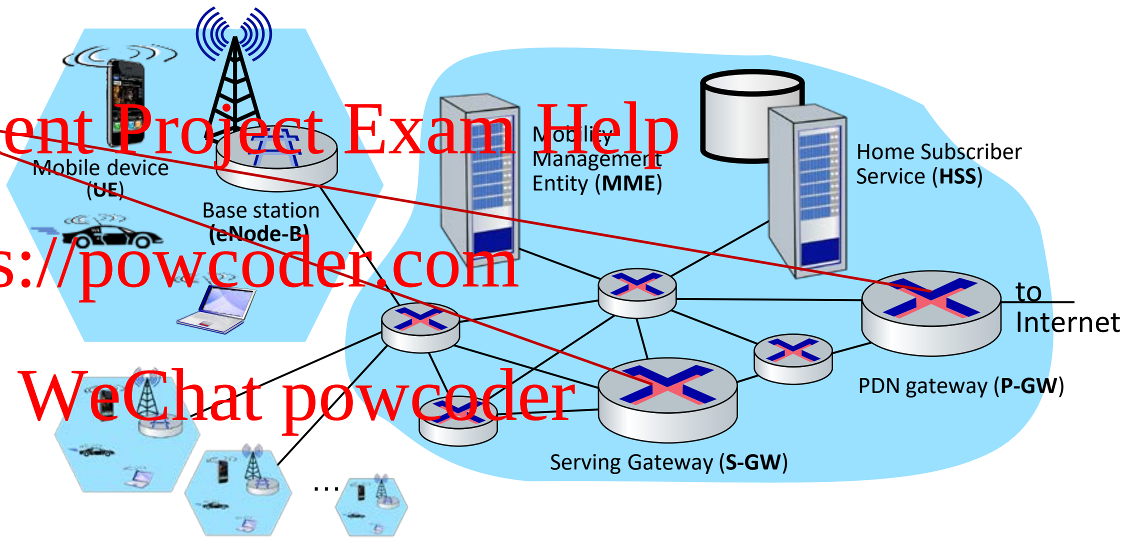
- stores info about mobile devices for which the HSS's network is their "home network"
- works with MME in device authentication



Elements of 4G LTE architecture

Serving Gateway (S-GW), PDN Gateway (P-GW)

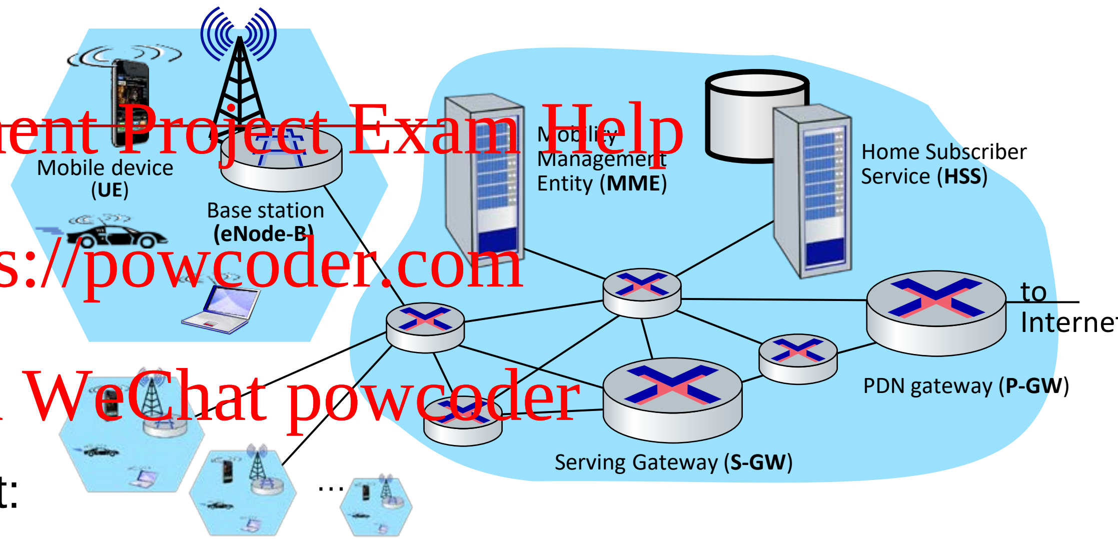
- lie on data path from mobile to/from Internet
- P-GW
  - gateway to mobile cellular network
  - Looks like nay other internet gateway router
  - provides NAT services
- other routers:
  - extensive use of tunneling



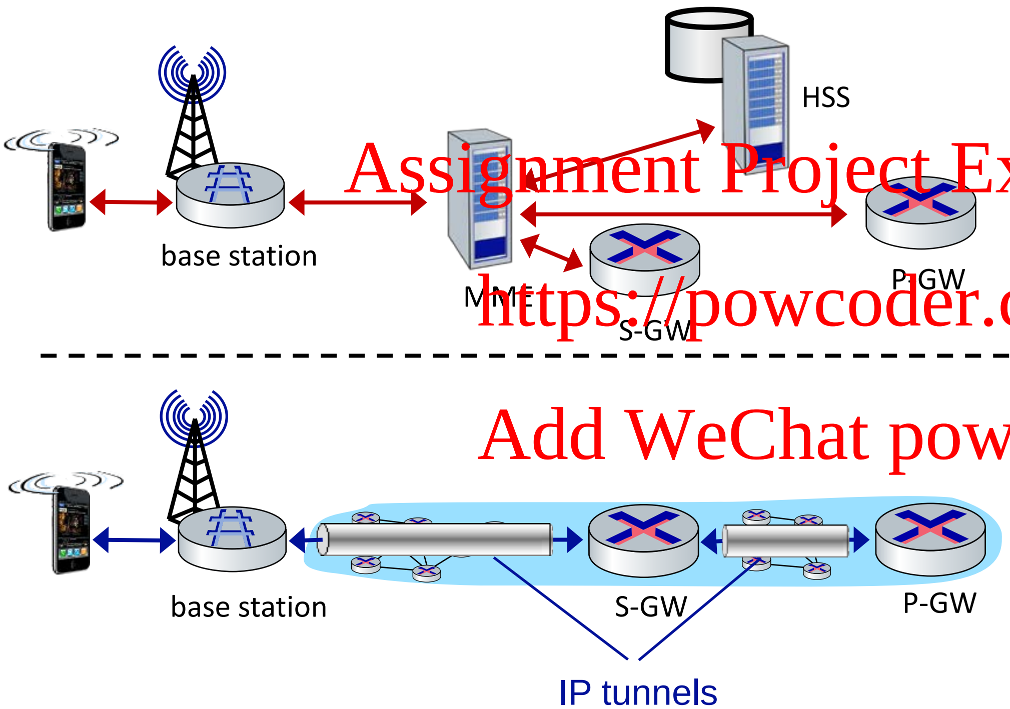
Elements of 4G LTE architecture

Mobility Management Entity

- device authentication (device-to-network, network-to-device) coordinated with mobile home network HSS
- mobile device management:
  - device handover between cells
  - tracking/paging device location
- path (tunneling) setup from mobile device to P-GW



LTE: data plane control plane separation



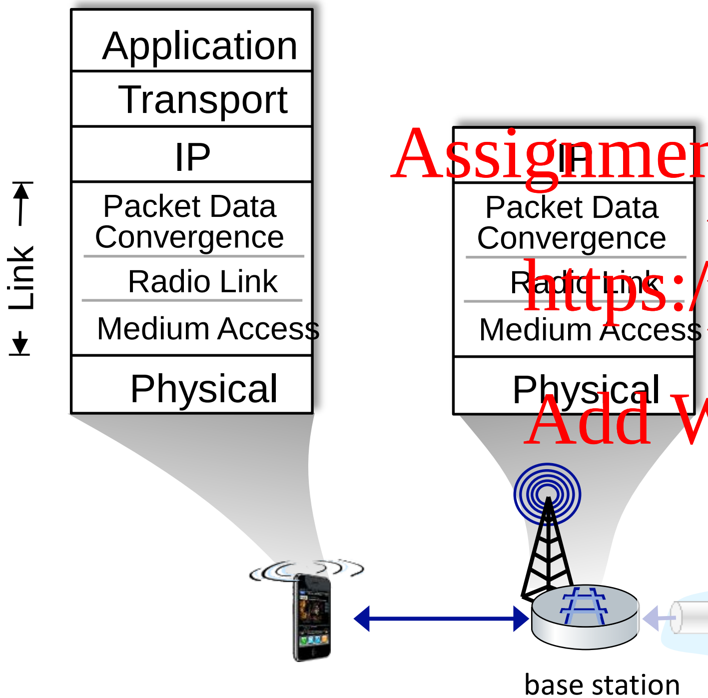
control plane

- new protocols for mobility management , security, authentication

data plane

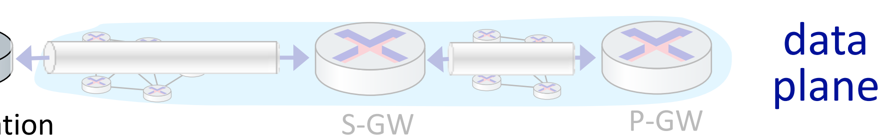
- new protocols at link, physical layers
- extensive use of tunneling to facilitate mobility

LTE data plane protocol stack: first hop

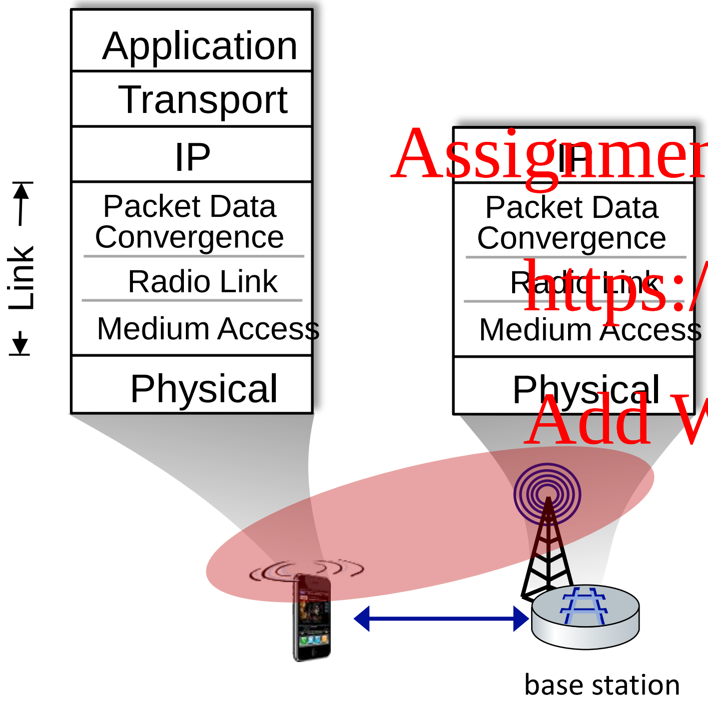


LTE link layer protocols:

- Packet Data Convergence: header compression, encryption
- Radio Link Control (RLC) Protocol: fragmentation/reassembly, reliable data transfer
- Medium Access: requesting, use of radio transmission slots



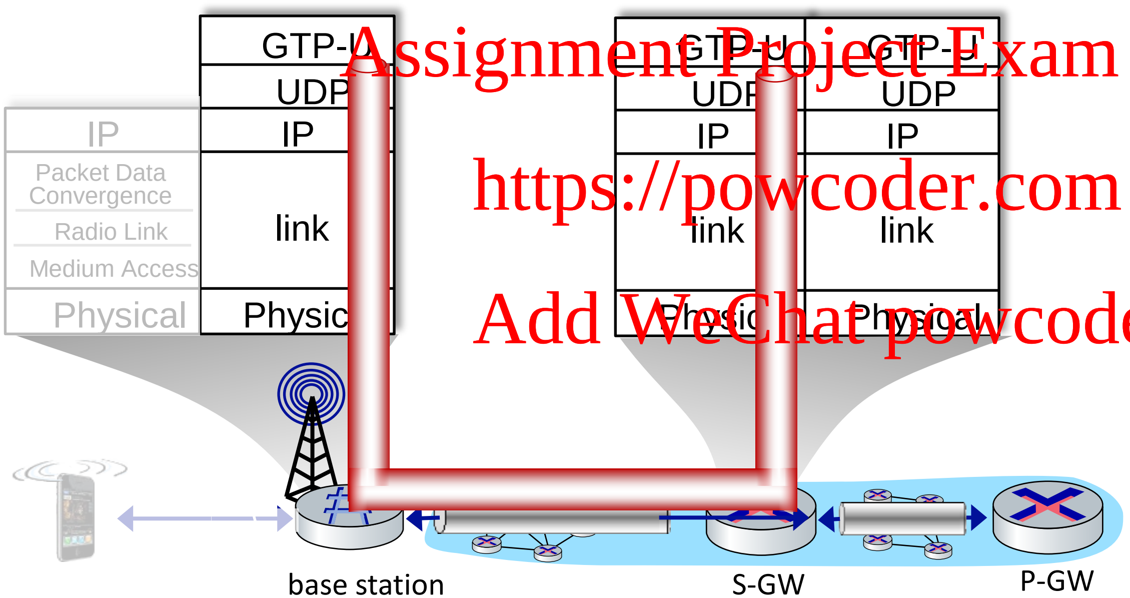
LTE data plane protocol stack: first hop



LTE radio access network:

- downstream channel: FDM, TDM within frequency channel (OFDM - orthogonal frequency division multiplexing)
  - "orthogonal": minimal interference between channels
  - upstream: FDM, TDM similar to OFDM
- each active mobile device allocated two or more 0.5 ms time slots over 12 frequencies
  - scheduling algorithm not standardized – up to operator
  - 100's Mbps per device possible

LTE data plane protocol stack: packet core

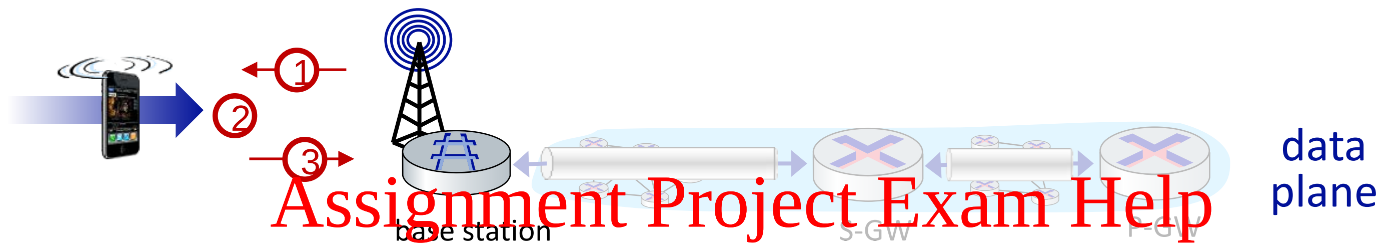


GTP-U: user data
GTP-C: control

tunneling:

- mobile datagram encapsulated using GPRS Tunneling Protocol (GTP), sent inside UDP datagram to S-GW
- S-GW re-tunnels datagrams to P-GW
- supporting mobility: only tunneling endpoints change when mobile user moves

LTE data plane: associating with a BS



1. BS broadcasts primary synch signal every 5 ms on all frequencies
   - BSs from multiple carriers may be broadcasting synch signals

2. mobile finds a primary synch signal, then locates 2nd synch signal on this freq.
   - mobile then finds info broadcast by BS: channel bandwidth, configurations; BS's cellular carrier info
   - mobile may get info from multiple base stations, multiple cellular networks

3. mobile selects which BS to associate with (*e.g.*, preference for home carrier)

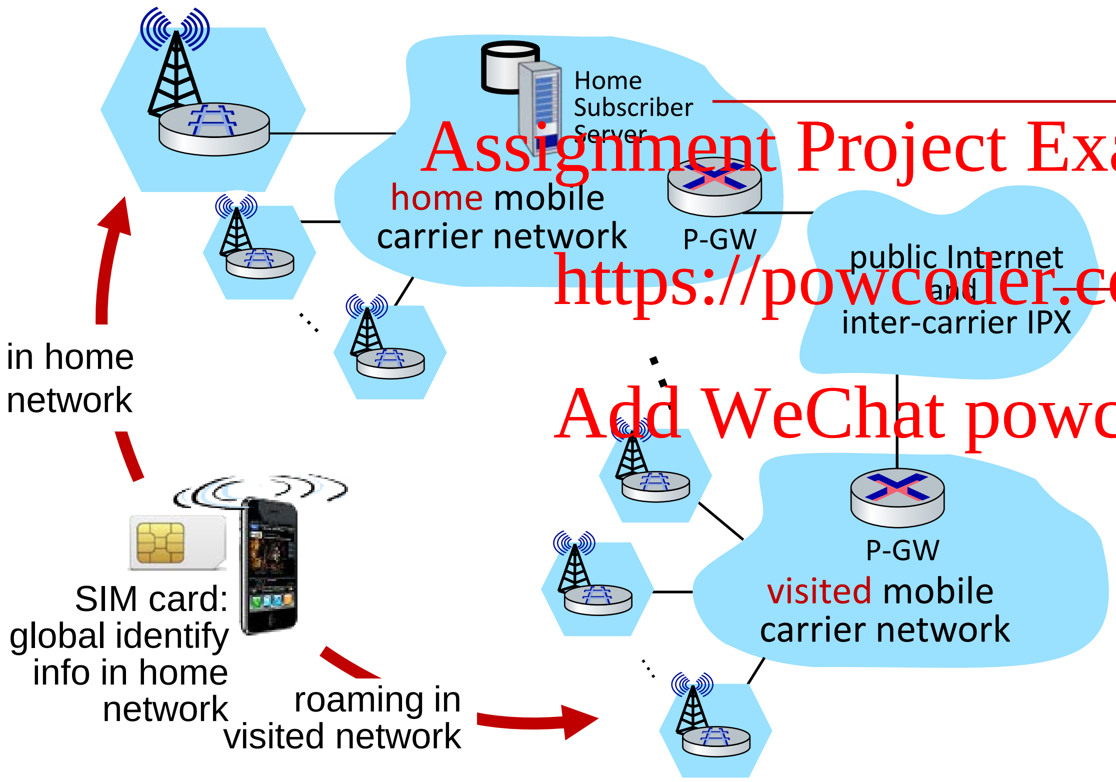
4. more steps still needed to authenticate, establish state, set up data plane

LTE mobiles: sleep modes

as in WiFi, Bluetooth: LTE mobile may put radio to “sleep” to conserve battery:

- light sleep: after 100’s msec of inactivity
  - wake up periodically (100’s msec) to check for downstream transmissions
- deep sleep: after 5-10 secs of inactivity
  - mobile may change cells while deep sleeping – need to re-establish association

Global cellular network: a network of IP networks



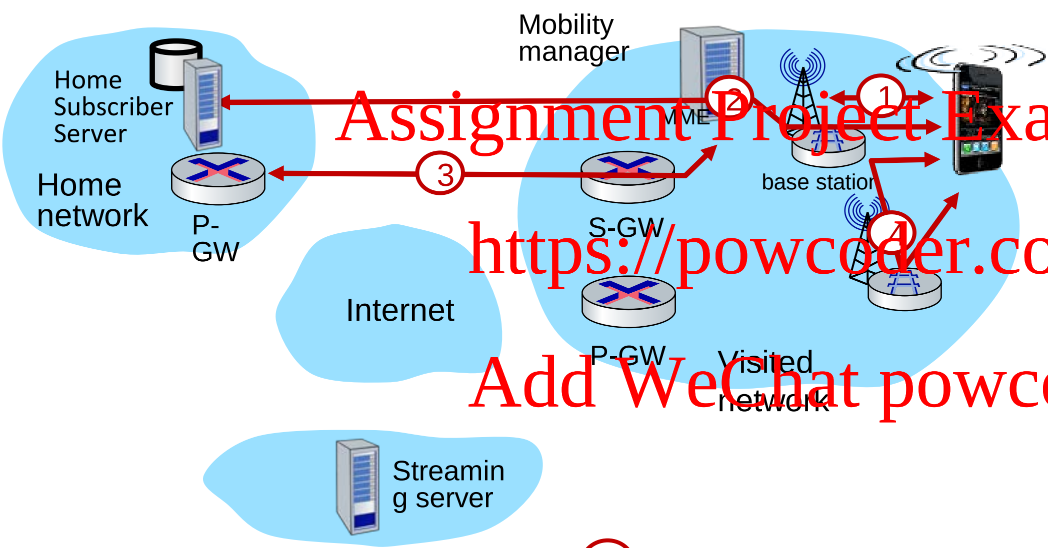
home network HSS:

- identify & services info, while in home network and roaming

all IP:

- carriers interconnect with each other, and public internet at exchange points
- legacy 2G, 3G: not all IP, handled otherwise

Mobility in 4G networks: major mobility tasks



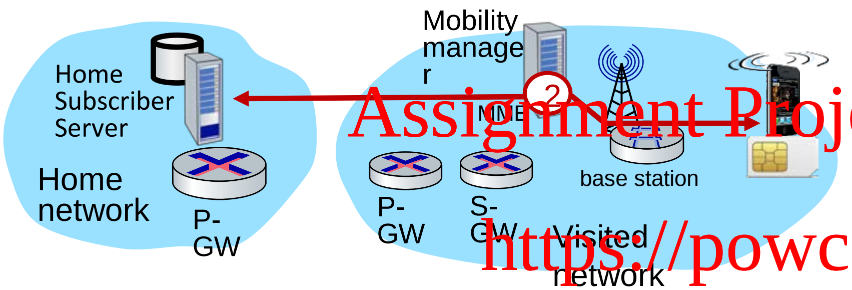
1 base station association:
- covered earlier
- mobile provides IMSI – identifying itself, home network

2 control-plane configuration:
- MME, home HSS establish control-plane state - mobile is in visited network

3 data-plane configuration:
- MME configures forwarding tunnels for mobile
- visited, home network establish tunnels from home P-GW to mobile

4 mobile handover:
- mobile device changes its point of attachment to visited network

Configuring LTE control-plane elements

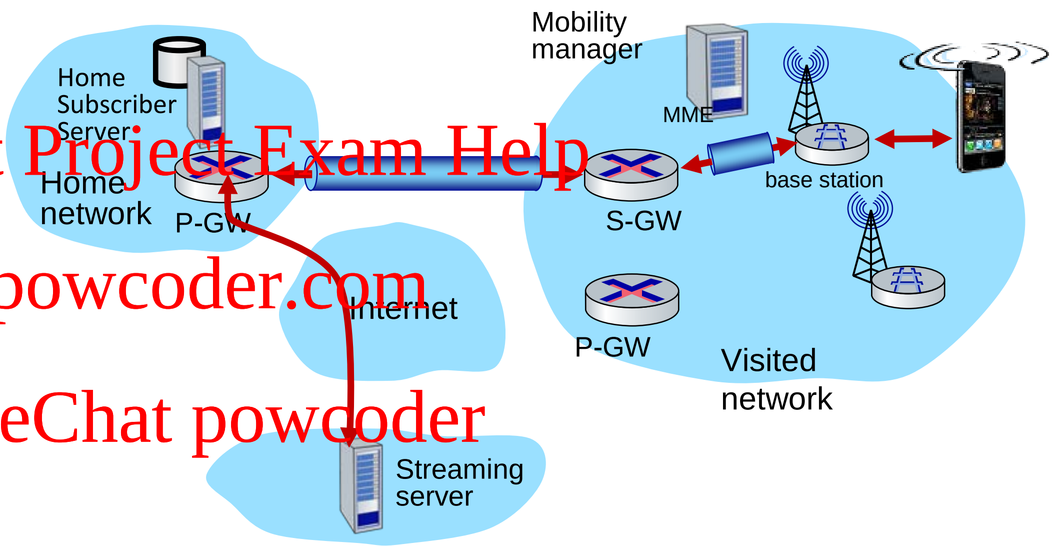


- Mobile communicates with local MME via BS control-plane channel
- MME uses mobile’s IMSI info to contact mobile’s home HSS
  - retrieve authentication, encryption, network service information
  - home HHS knows mobile now resident in visited network
- BS, mobile select parameters for BS-mobile data-plane radio channel

Configuring data-plane tunnels for mobile

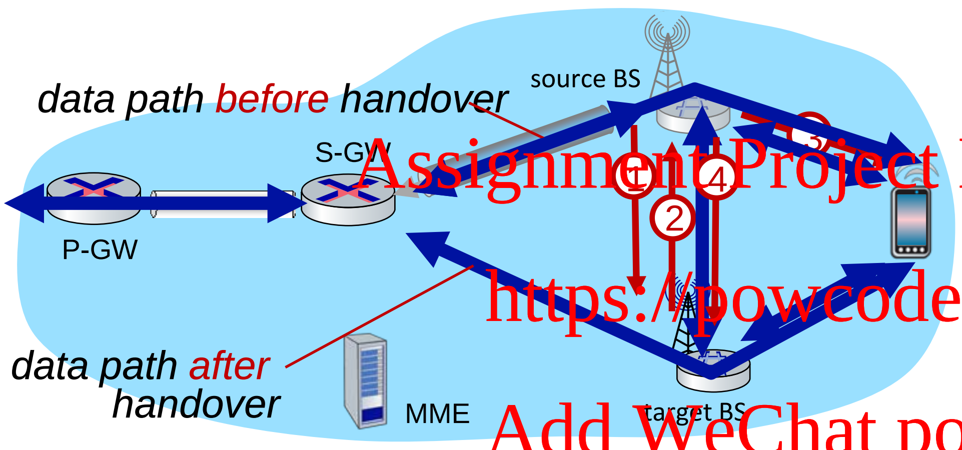
› **S-GW to BS tunnel**: when mobile changes base stations, simply change endpoint IP address of tunnel

› **S-GW to home P-GW tunnel**: implementation of indirect routing



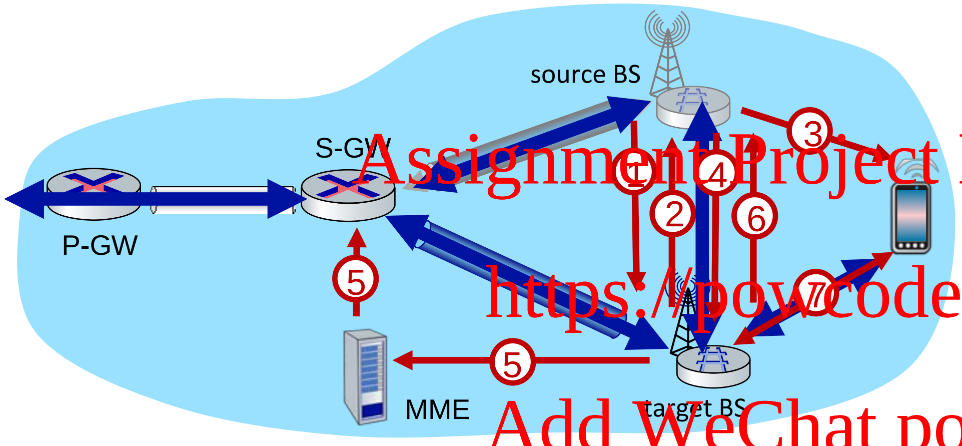
- tunneling via GTP (GPRS tunneling protocol): mobile's datagram to streaming server encapsulated using GTP inside UDP, inside datagram

Handover between BSs in same cellular network



1. current (source) BS selects target BS, sends *Handover Request message* to target BS
2. target BS pre-allocates radio time slots, responds with HR ACK with info for mobile
3. source BS informs mobile of new BS
   - mobile can now send via new BS - handover *looks* complete to mobile
4. source BS stops sending datagrams to mobile, instead forwards to new BS (who forwards to mobile over radio channel)

Handover between BSs in same cellular network



5 target BS informs MME that it is new BS for mobile

- MME instructs S-GW to change tunnel endpoint to be (new) target BS

6 target BS ACKs back to source BS: handover complete, source BS can release resources

7 mobile's datagrams now flow through new tunnel from target BS to S-GW

On to 5G!

- goal: 10x increase in peak bitrate, 10x decrease in latency, 100x increase in traffic capacity over 4G
- 5G NR (new radio):
  - two frequency bands: FR1 (450 MHz–6 GHz) and FR2 (24 GHz–52 GHz): millimeter wave frequencies
  - not backwards-compatible with 4G
  - MIMO: multiple directional antennae
- millimeter wave frequencies: much higher data rates, but over shorter distances
  - pico-cells: cells diameters: 10-100 m
  - massive, dense deployment of new base stations required

QUIC

Quick UDP Internet Connections

draft-ietf-quic-transport

QUIC history

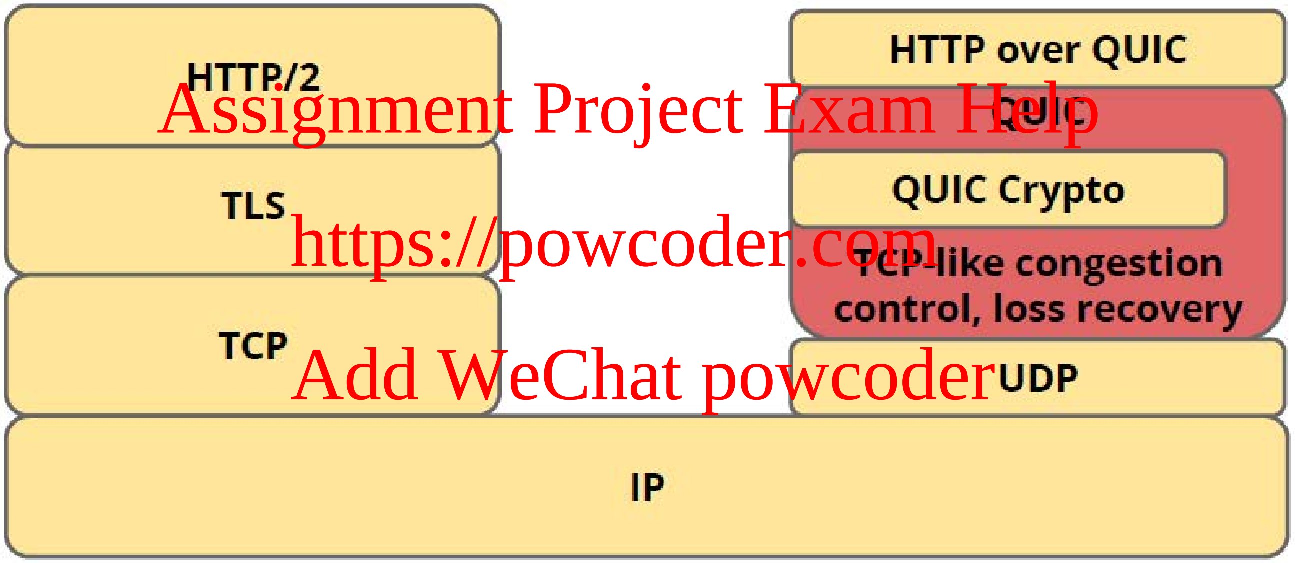
Experimental protocol, deployed at Google starting in 2014

- Between Google services and Chrome
- Improved page load latency, video rebuffer rate
- Successful experiment today
- ~35% of Google's egress traffic (~7% of Internet traffic)
- Akamai deployment in 2016

QUIC Work Group formed in Oct 2016

- Modularize and standardize QUIC in parts
- HTTP as initial application

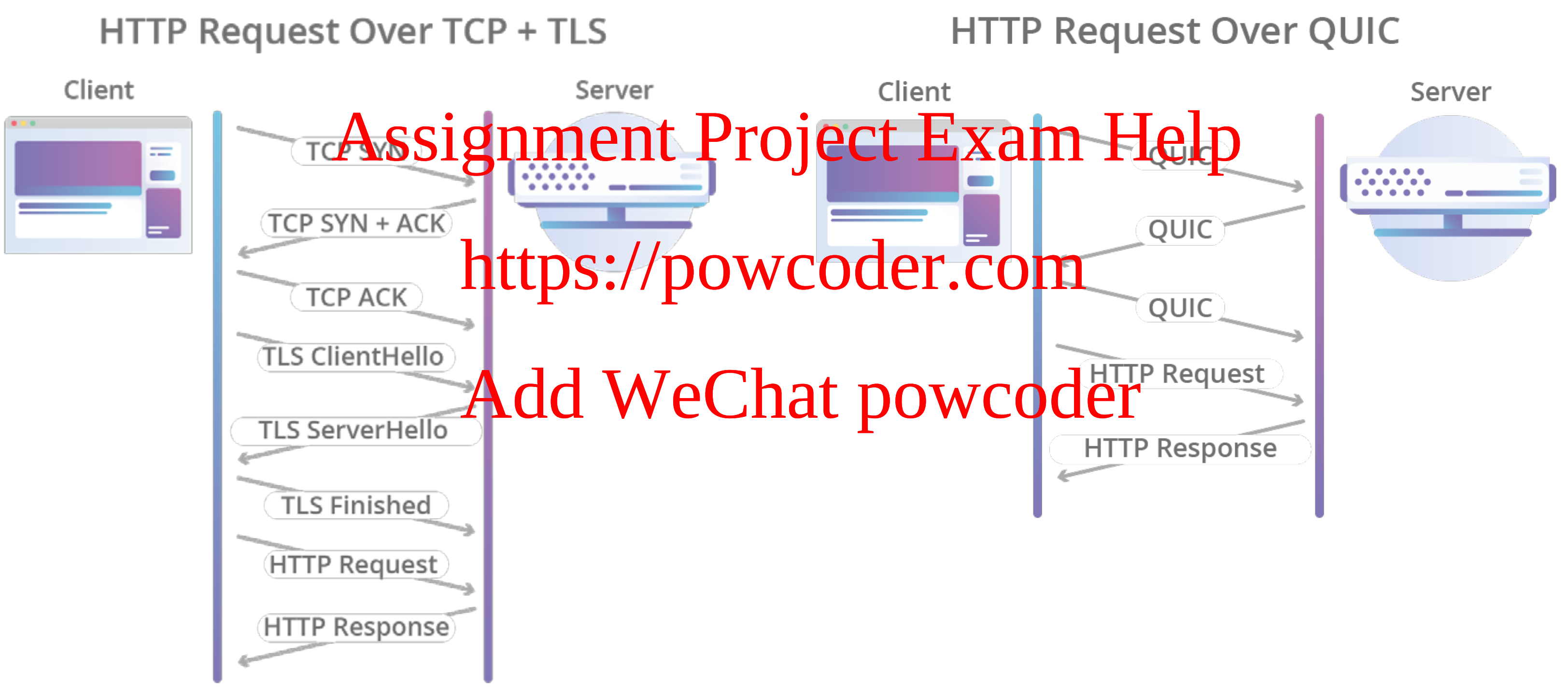
QUIC overview



https://www.ietf.org/proceedings/98/slides/slides-98-edu-sessf-quic-tutorial-00.pdf

THE UNIVERSITY OF SYDNEY

Built-in security (and performance)

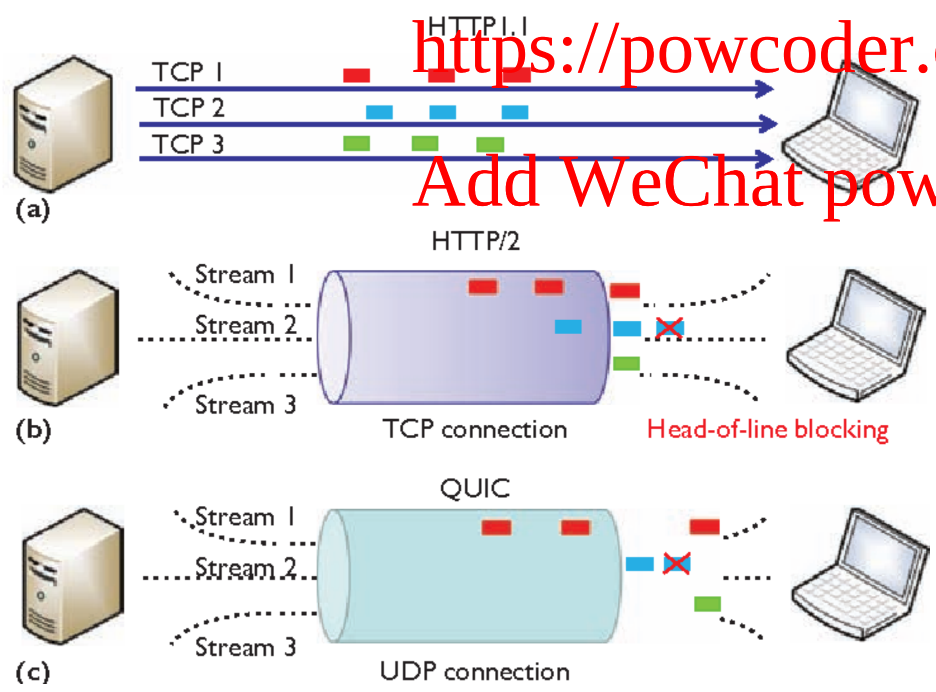


https://blog.cloudflare.com/the-road-to-quic/

Head-of-line blocking

Multiple streams of data to reach all the endpoints independently, and hence independent of packet losses involving other streams.

Avoid head-of-line blocking.

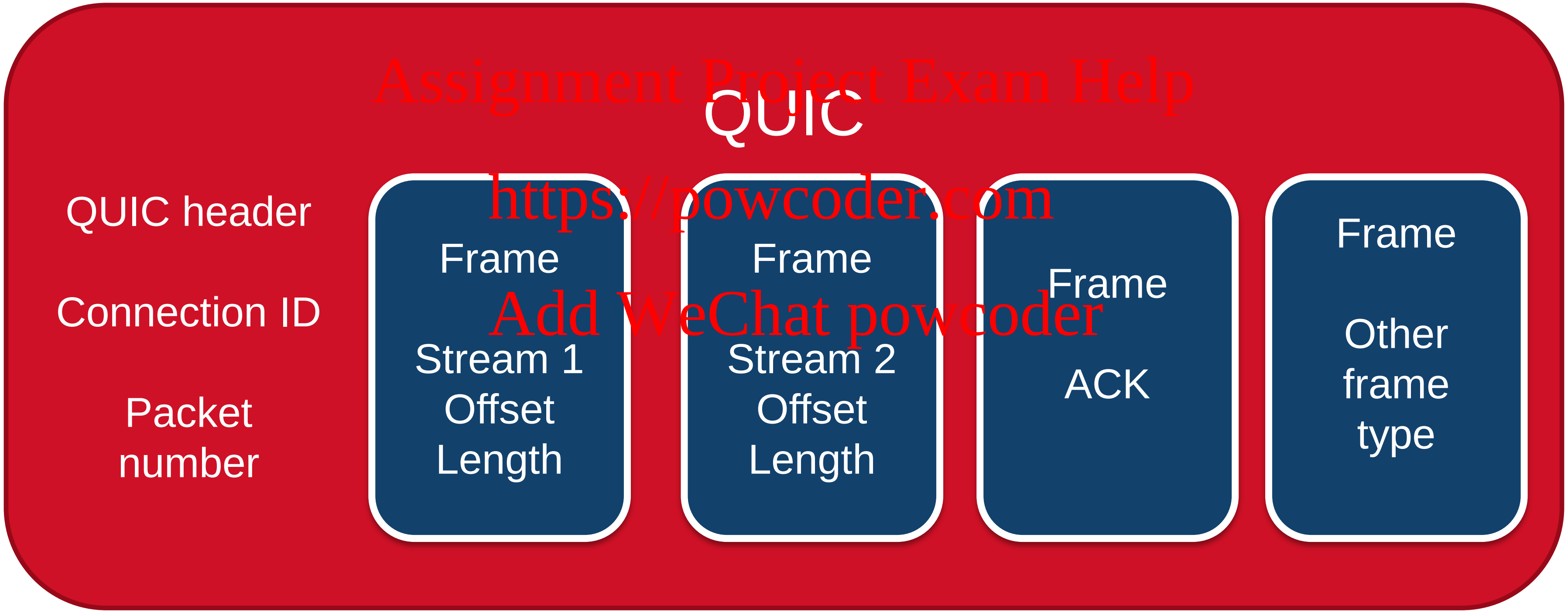


UDP does not care about ordering of packet and if packet get lost.

QUIC is solving this issue and it will take care of packet lost in particular stream.

https://medium.com/faun/http-2-spdy-and-http-3-quic-bae7d9a3d484

Connection ID



Connection ID

Connection ID: identifier that is used to identify a QUIC connection

Review:

TCP uses (source IP, destination IP, source port number, destination port number) to identify socket.

UDP uses (destination IP, destination port number) to identify socket.

What happens device is migrated (e.g., from 4G to WiFi)

Connection ID for smooth handover.

Packet Number and Offset

Packet number: monotone, non-repeating

Offset + length: Protect the order of the stream

Packet is lost: Application decides if retransmit the lost frames.

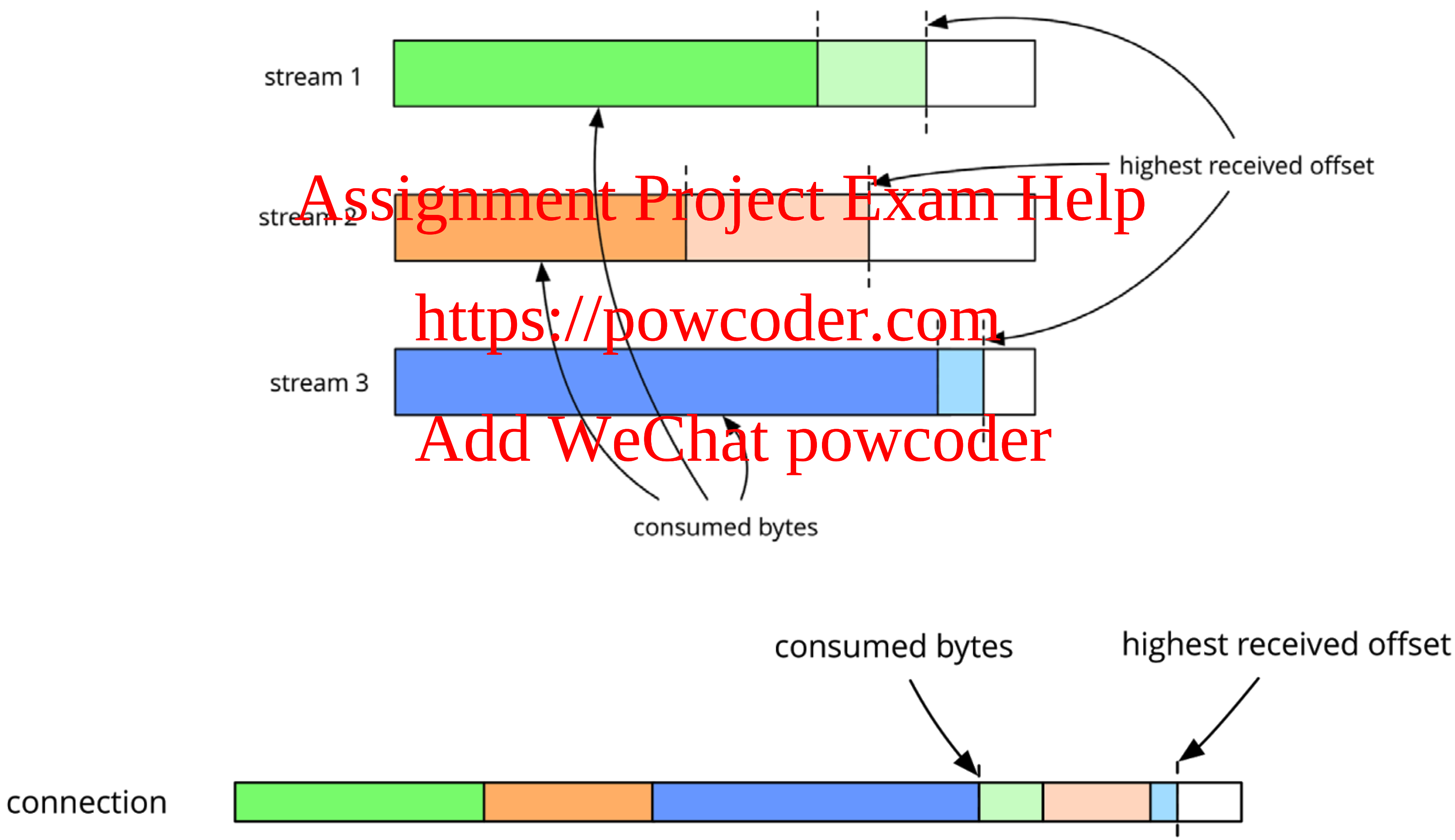
Loss detection separates from loss recovery

Other frame types

Examples:

MAX_STREAM_DATA: connection level flow control
MAX_STREAM_ID: stream level flow control
PING/PONG: to verify that their peers are still alive
CONNECTION_CLOSE: the connection is being closed.

Connection-level and stream-level flow control



Congestion control

- Similar to TCP but more advanced.
  - Monotone packet number: No RTT estimation ambiguity if packet is lost.
  - Timeout:
    smoothed_rtt + max(4*rttvar, kGranularity) + max_ack_delay

    kGranularity: timer granularity, 1ms
    max_ack_delay: the maximum amount of time by which the receiver intends to delay acknowledgments for packets

Congestion control

- Slow start
- Congestion avoidance (linear increase).
- Recovery period (halve window size).

- Loss detection:

  - By ACK (similar to 3 duplicate ACKs)
  - By timeout

  - Loss causes recovery

- Persistent Congestion causes “slow start”
  - A sender establishes loss of all in-flight packets sent over a long enough duration, the network is considered to be experiencing persistent congestion.

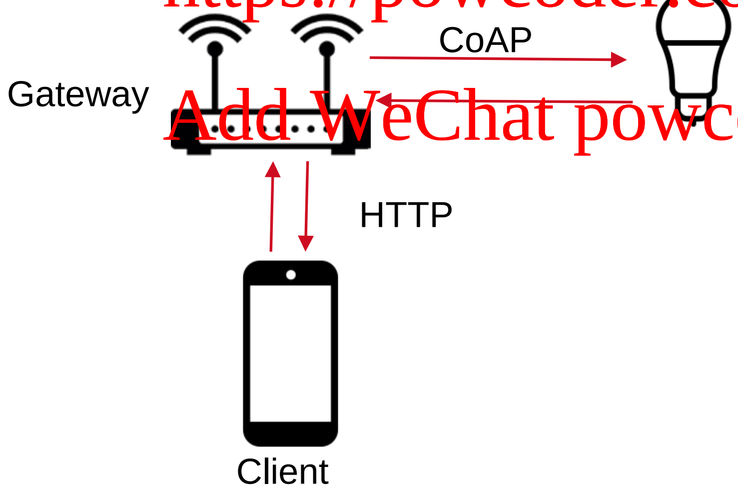
CoAP

Constrained Application Protocol

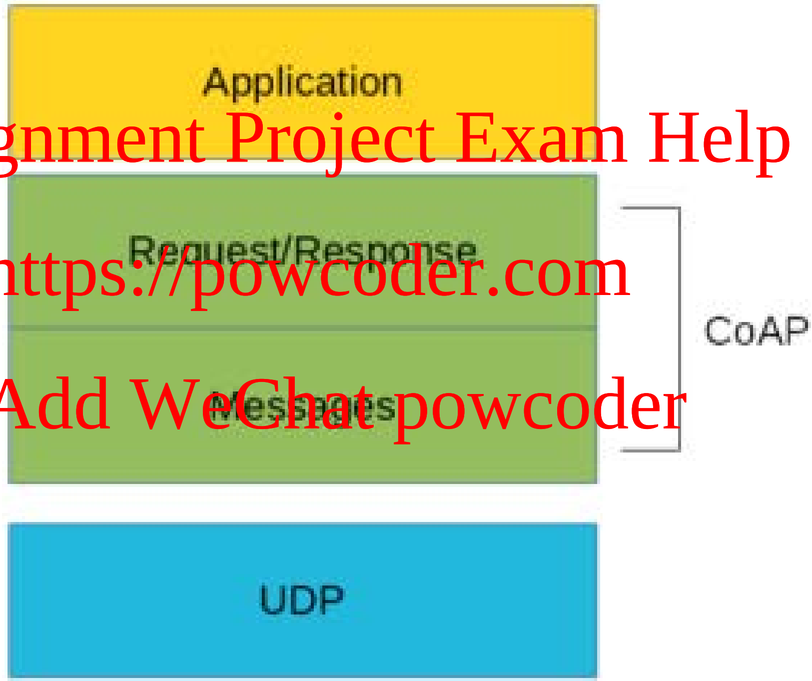
IETF RFC 7252

CoAP for IoT

- CoAP provides a request/response interaction like HTTP.
- Smaller messages than HTTP and with very low overhead.
- Suitable for IoT devices (sensors and actuators) with limited memory and storage.
  - For example, to obtain a current temperature, send a GET request.
  - To turn on/off or toggle LEDs we use PUT requests.



CoAP



CoAp

- Has a scheme coap://
- Based on UDP.
- Has a well known port.

- GET, PUT, DELETE

- Confirmable messages (CON) requires an ACK with message ID. The message ID of the ACK matches the message ID of the confirmable message.

- Non-confirmable (NON) messages do not require an ACK. Less reliable.
- Responses are matched with requests via the client generated Token.
- Example:

CoAP Client   CoAP Server

---->   CON {id} GET /basement/light   Confirmable request has an ID

<----   ACK {id} 2.05 Content {“status” : “on”}   Piggy back response and same ID

CoAP message types

- CoAP supports different message types:
  - Confirmable (CON)
    - Reliable message, need ACK. CON and ACK have the same ID.
  - Non-confirmable
    - No need ACK
  - Acknowledgment
  - Reset
    - Server has troubles managing the incoming request.

CoAP Uses Timeouts over UDP

CoAP Client CoAP Server

----> CON {id} GET /basement/light lost request

timeout

-- --> CON {id} GET /basement/light finally arrives

<---- ACK {id} Content {“status” : “on”}

The {id} allows us to detect duplicates.

CoAP Request/Acknowledge/Callback

CoAP Client CoAP Server

----> CON {id} PUT /basement/cleanFloor Token: 0x22 Needs time

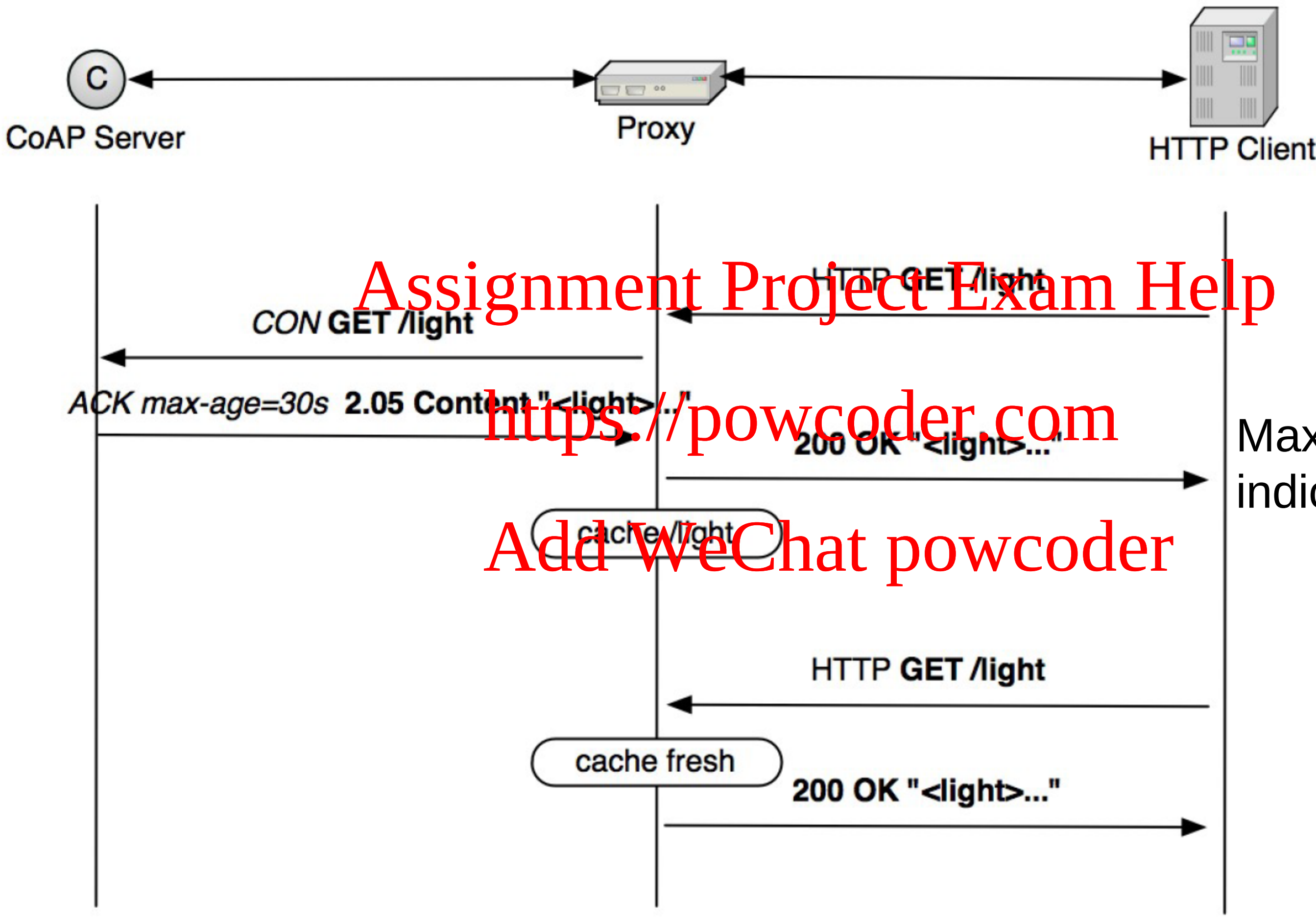
<---- ACK {id} I am on it!

<----- CON {newID} Content /basement/cleanFloor Token: 0x22 Done

----> ACK {newID}

The same token is used to identify this request and the service response.

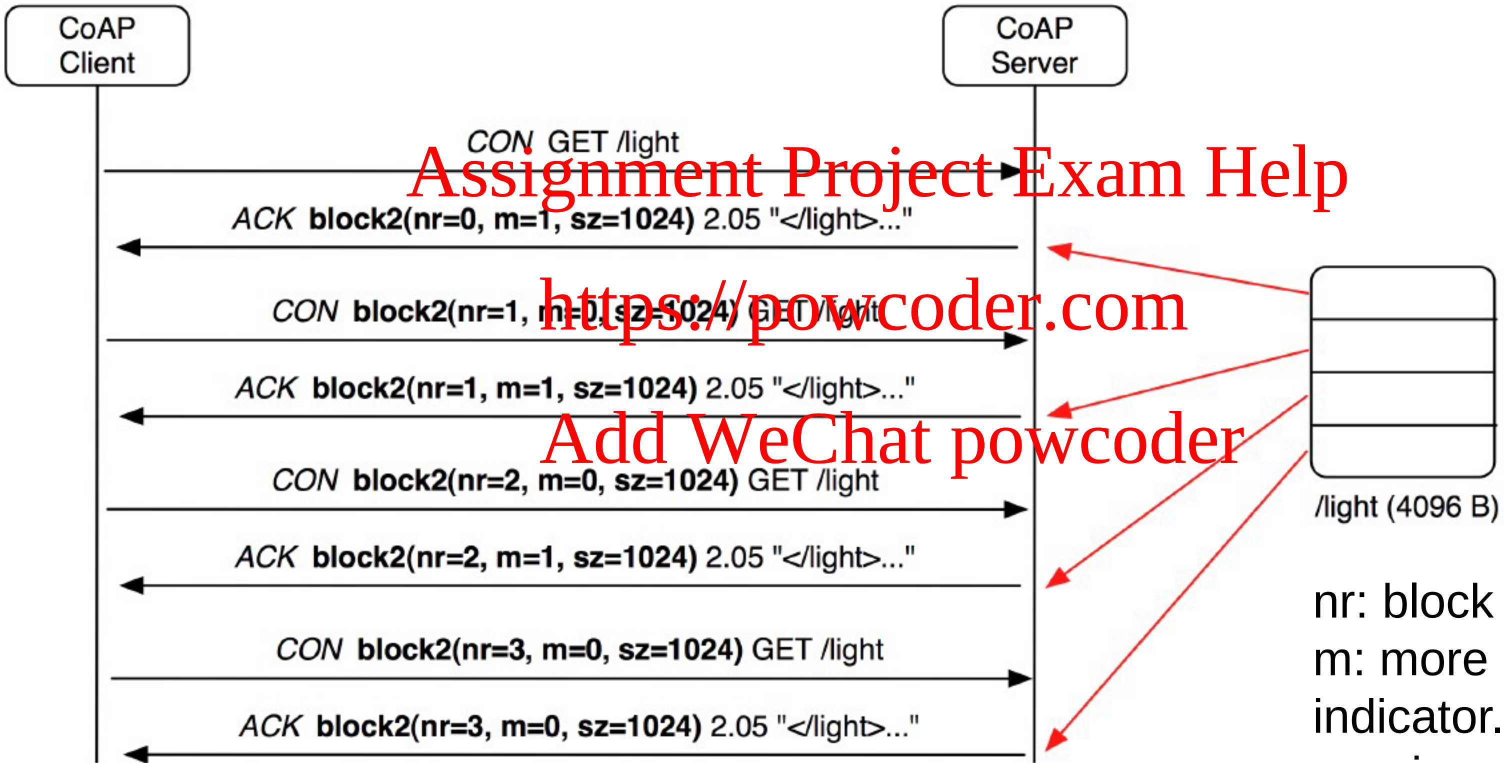
Cache



Max-Age option indicates cache lifetime

https://community.arm.com/cfs-file/__key/telligent-evolution-components-attachments/01-1996-00-00-00-00-53-31/ARM-CoAP-Tutorial-April-30-2014.pdf

Block-Wise Transfer



nr: block number.
m: more block indicator.
sz: size
block1: for request.
block2: for response.

https://community.arm.com/cfs-file/__key/telligent-evolution-components-attachments/01-1996-00-00-00-00-53-31/ARM-CoAP-Tutorial-April-30-2014.pdf

CoAP Publish/Subscribe

The GET includes an “Observe” message to establish a subscription request.
The response includes an “Observe” to say this is a publication.
The value included with Observe response is there for possible re-orderings.

Token matches

CoAP Client CoAP Server

----> CON GET /basement/light Observe: 0 Token: 0x22 Registration

<---- ACK Observe: 27 Token 0x22 Current state

<---- CON Observe: 28 Token: 0x22 {“light” : ”off”}

-----> ACK Token: 0x22

<---- CON 200 Observe: 29 Token: 0x22 {“light” : ”on”} Notification of stage change

-----> ACK Token: 0x22

CoAP Resource Discovery

CoAP Client | CoAP Server

----> CON {id} GET /.well-known/core

<---- ACK {id} Content “/sensor/temp /sensor/light”

----> CON {id} GET /sensor/light

<---- ACK {id} Content “dim”

----> CON {id} GET /sensor/temp

<---- ACK {id} Content “72”

CoAP Packet Format

CoAP Header

Offsets	Octet	0								1								2								3							
Octet	**Bit**	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
4	**32**	VER		Type		Token Length				Request/Response Code								Message ID															
8	**64**	Token (0 - 8 bytes)																															
12	**96**																																
16	**128**	Options (If Available)																															
20	**160**	1	1	1	1	1	1	1	1	Payload (If Available)																							

Version (VER)
Indicates the CoAP version number.
Type (2 bits)
Request
0 : Confirmable : This message expects a corresponding Acknowledgement message.
1 : Non-confirmable : This message does not expect a confirmation message.
Response
2 : Acknowledgement : This message is a response that acknowledge a confirmable message
3 : Reset : This message indicates that it had received a message but could not process it.
Token Length (4 bits)
Indicates the length of the variable-length Token field
Request/Response Code (8 bits)
For example 2.05 Content similar to HTTP 200. 4.00 Bad request
Message ID (16 bits)
Token

REST (Representational state transfer)

A set of operations to be used for creating Web services

Server provides access to resources and client accesses and modifies the resources: stateless operations.

GET: read information

PUT: update information

POST: create information

DELETE: delete information

Both HTTP and CoAP are based on REST.

MQTT

Message Queuing Telemetry Transport

ISO/IEC 20922

MQTT

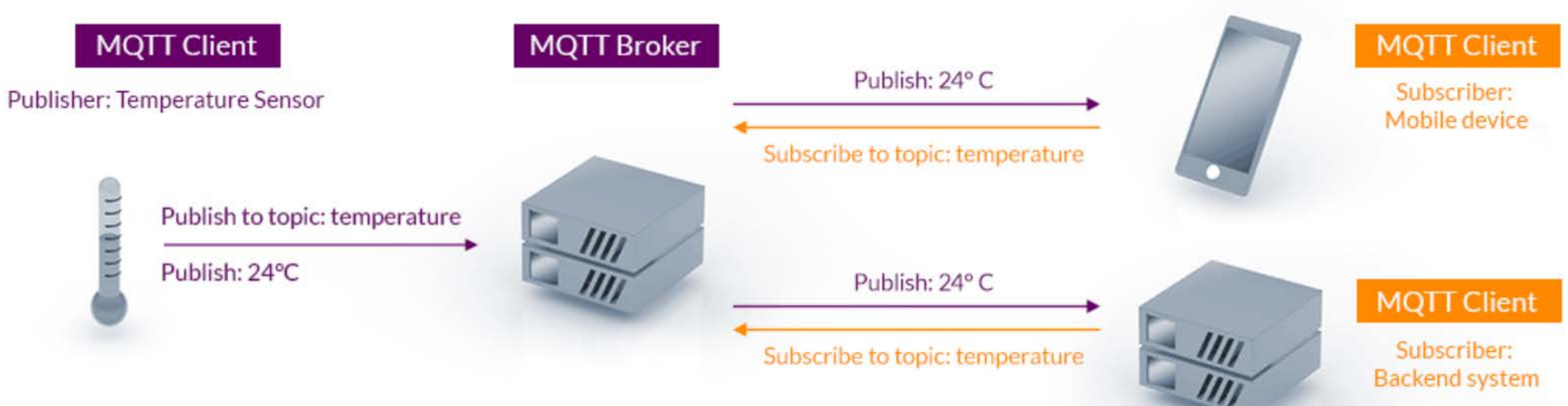
MQTT: Lightweight, publish-subscribe network protocol that transports messages between devices.

- Runs over TCP/IP
- For IoT networks

Two types of entities:

- Broker: server receives and forwards messages
- Client: device connected to broker

Purpose: publish-subscribe information.



https://mqtt.org/

MQTT Topics

Information is organized as topics.

Publish

- Subscriber subscribes topics.
- Publisher has a new item of data to distribute.
- Publisher sends to broker.
- Broker distributes to clients subscribed to the topic.

Publisher does not need to know number/location of the subscribers.

Subscriber does not need to configure publishers.

MQTT Topics

Topics are structured in hierarchy, using / as delimiter

e.g.,house/room1/maindoor

Wildcard for multiple topics
multiple-level topics must be in the end
house/#
    house/room1/maindoor
    house/room1/window
    house/room2/maindoor
    house/room2/window
    house/maindoor
+ single-level topics
house/+/maindoor
    house/room1/maindoor
    house/room2/maindoor

MQTT Packet

Fixed header, present in all MQTT Control Packets
Variable header, present in some MQTT Control Packets
Payload, present in some MQTT Control Packets

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type				Flags specific to each MQTT Control Packet type			
byte 2…	Remaining Length							

Fixed header

MQTT Packet

Type:
CONNECT: connection request
CONNACK: connection ACK
PUBLISH: publish message
SUBCRIBE: subscribe request
SUBACK: subscribe ACK
UNSUBSCRIBE: unsubscribe request
UNSUBSCRIBEACK: unsubscribe request
others

Flag: mostly reserved.
For PUBLISH packets, it contains duplicate transmission flag and QoS level.

Remaining length:
Length of the packet (variable header + payload)

Variable Header

THE UNIVERSITY OF SYDNEY

MQTT QoS level

Data loss can sill occur if TCP connection is down and messages in transit is lost.

QoS 0: At most once - the message is sent only once and the client and broker take no additional steps to acknowledge delivery (fire and forget).

e.g., temperature sensor

QoS 1: At least once - the message is re-tried by the sender multiple times until acknowledgement is received (acknowledged delivery).

e.g., door sensor (status of door)

QoS 2: Exactly once - the sender and receiver engage in a two-level handshake to ensure only one copy of the message is received (assured delivery).

e.g., smoke sensor (alarm signal)

MQTT QoS level

QoS 0: At most once - the message is sent only once and the client and broker take no additional steps to acknowledge delivery (fire and forget).

https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/

MQTT QoS level

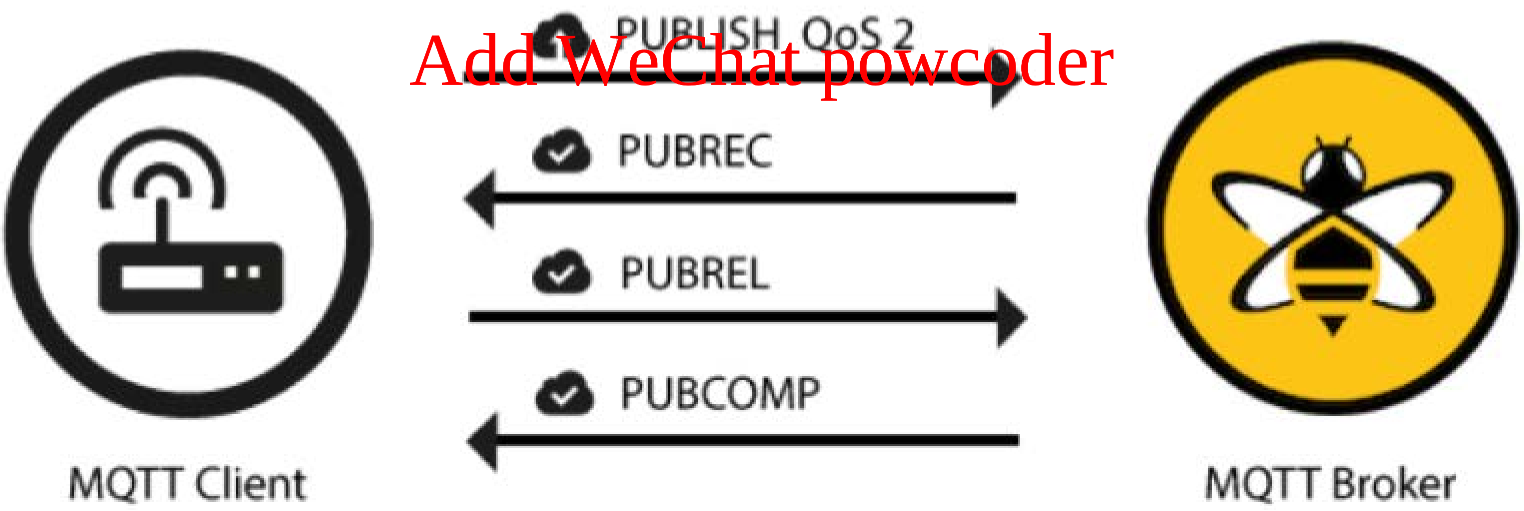
QoS 1: At least once - the message is re-tried by the sender multiple times until acknowledgement is received (acknowledged delivery).

https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/

MQTT QoS level

QoS 2: Exactly once - the sender and receiver engage in a two-level handshake to ensure only one copy of the message is received.

PUBREC: publication received.
PUBREL: publication released.
PUBCOM: publication completed.
(Other MQTT packet types.)



https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/

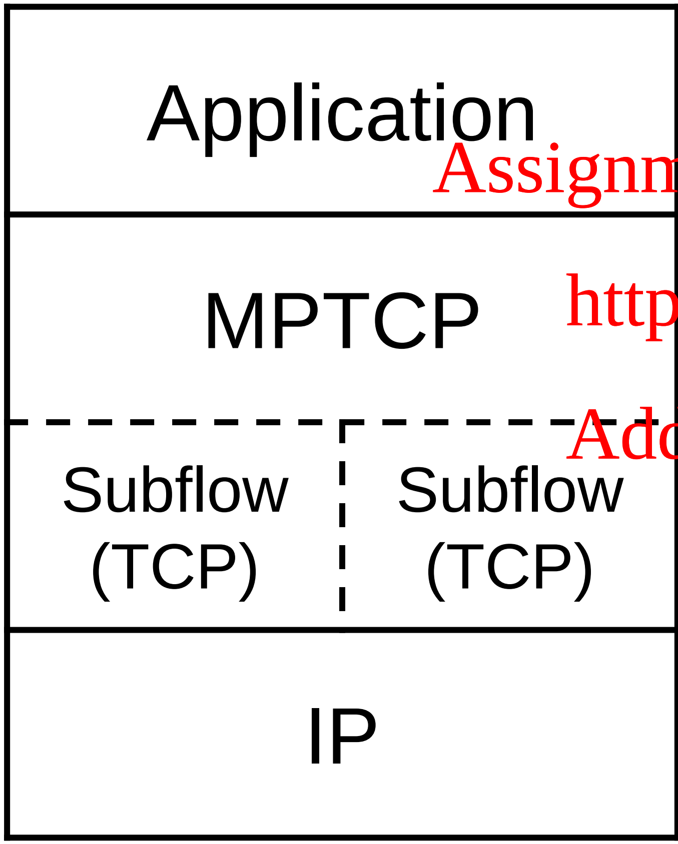
MPTCP

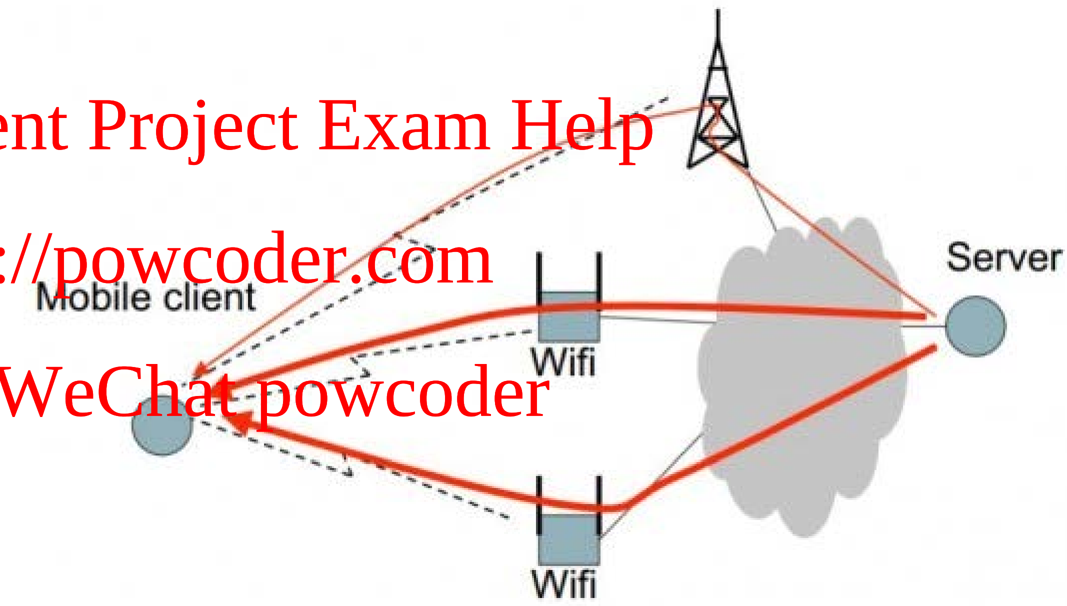
Multi-path TCP

IETF RFC 6824 (older version)

IETF RFC 8684 (latest version)

MPTCP in Stack

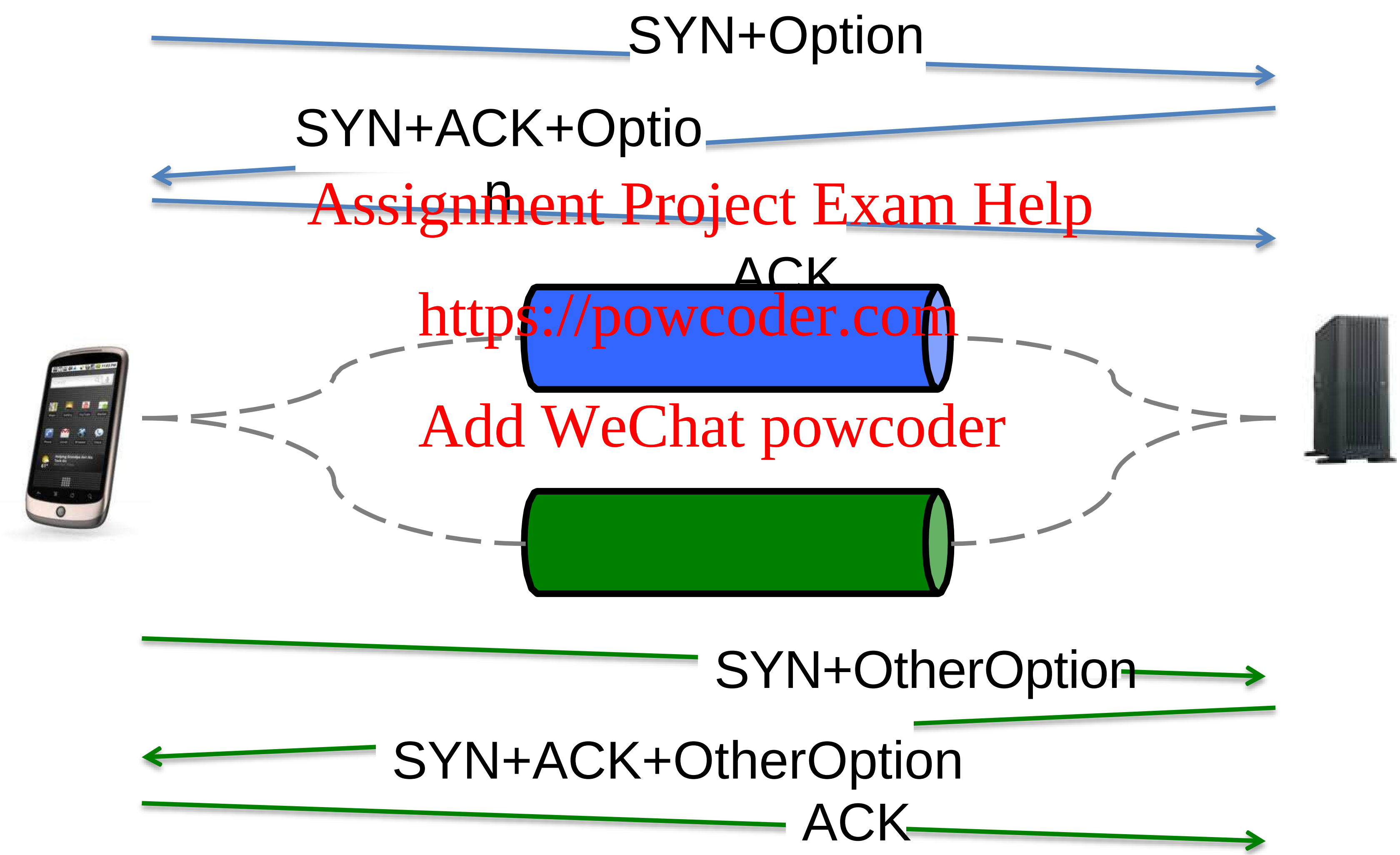




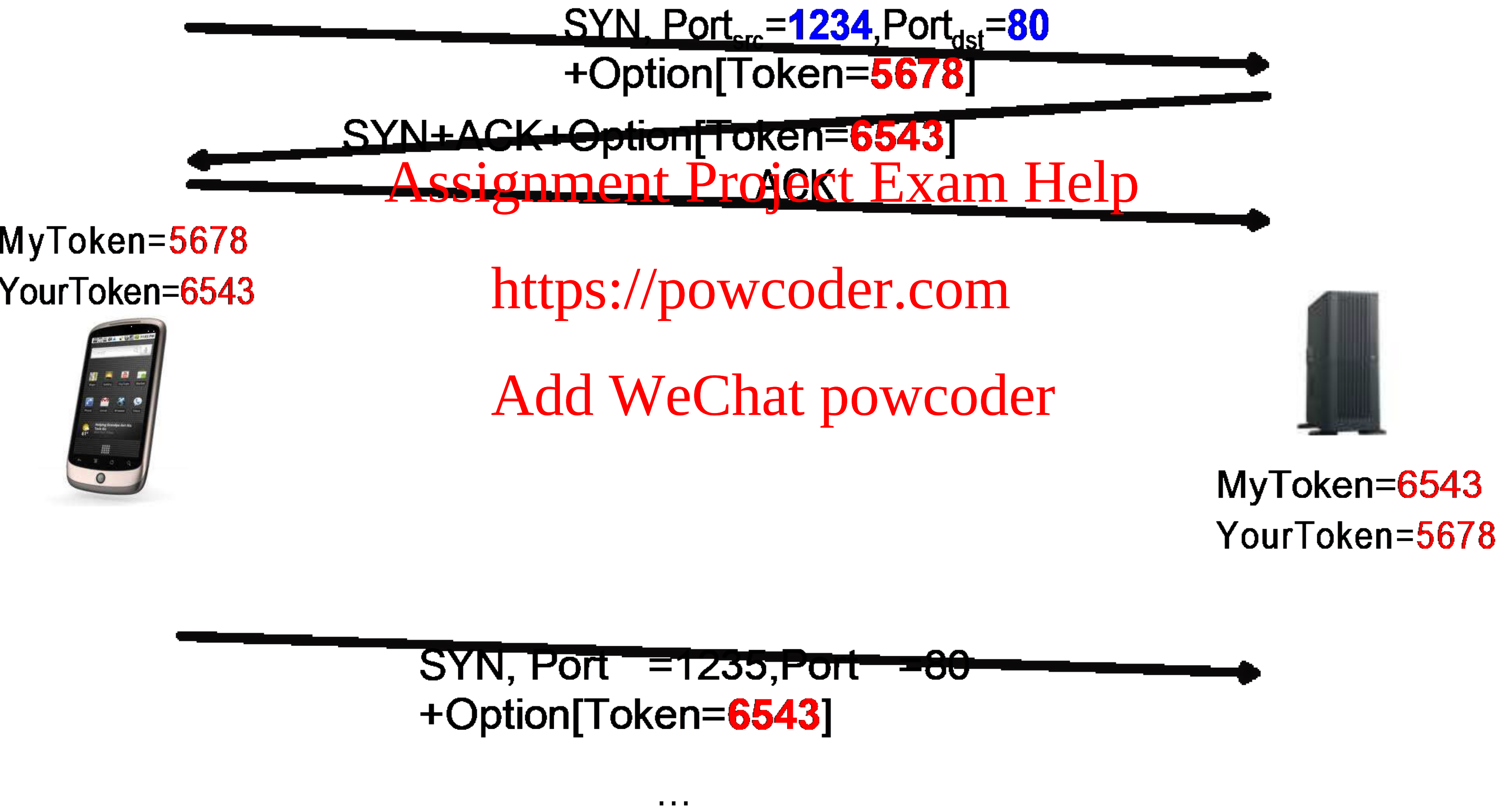
https://pocketnow.com/multipath-tcp

Different IP addresses for different subflows.

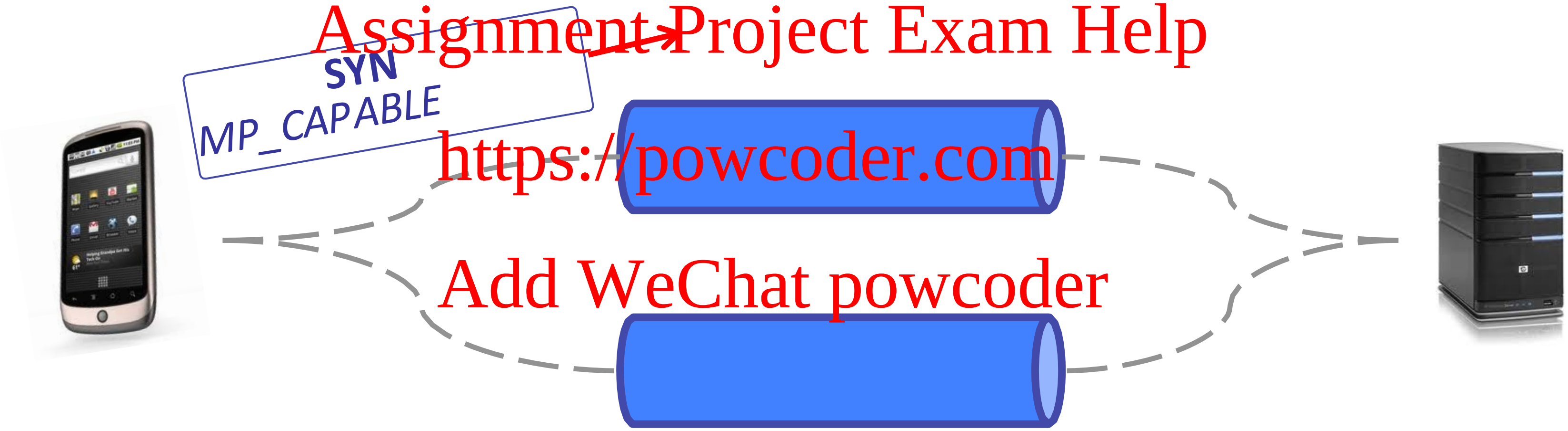
Option Field



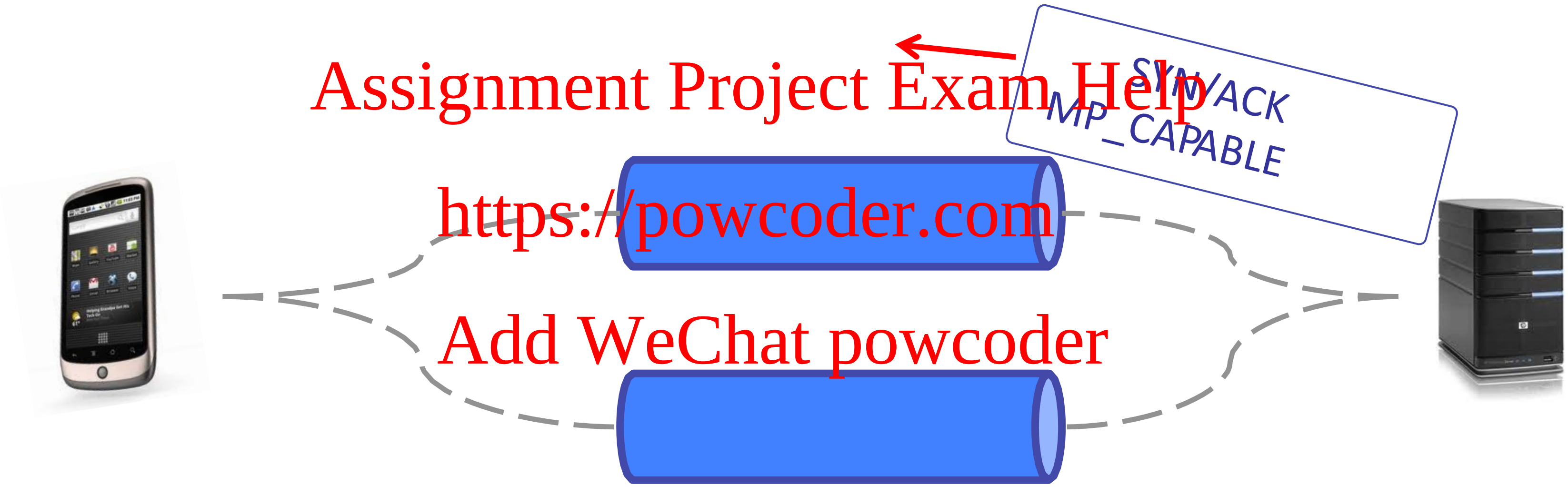
Token


SYN, Port$_{src}$=1234,Port$_{dst}$=80
+Option[Token=5678]
SYN+ACK+Option[Token=6543]
ACK
MyToken=5678
YourToken=6543
MyToken=6543
YourToken=5678
SYN, Port =1235,Port =80
+Option[Token=6543]
…

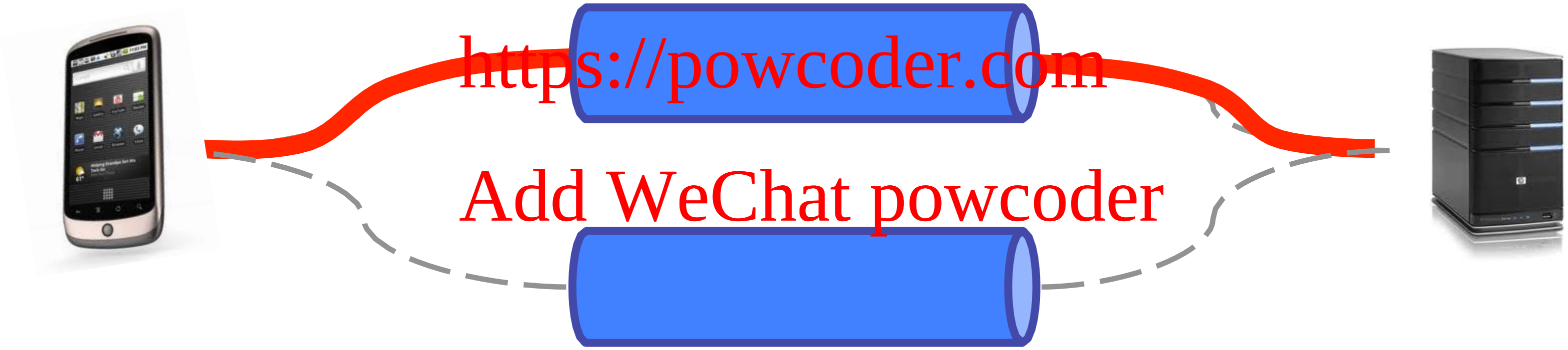
Initiation and Join



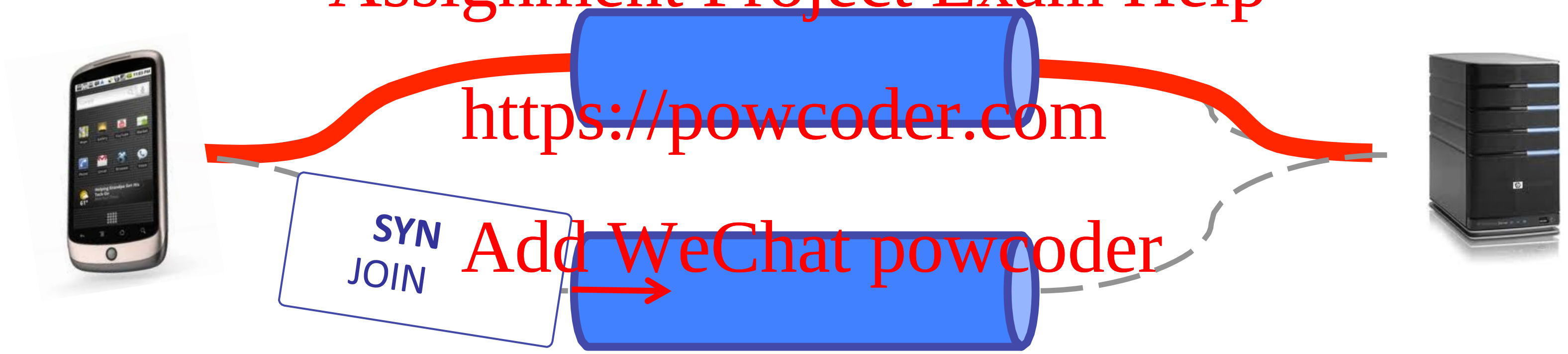
Initiation and Join



Initiation and Join



Initiation and Join



Initiation and Join

MPTCP sequence numbers

- MPTCP uses a Data Sequence Number (DSN) to number all data sent over the MPTCP connection.

- Each subflow has its own sequence number space.

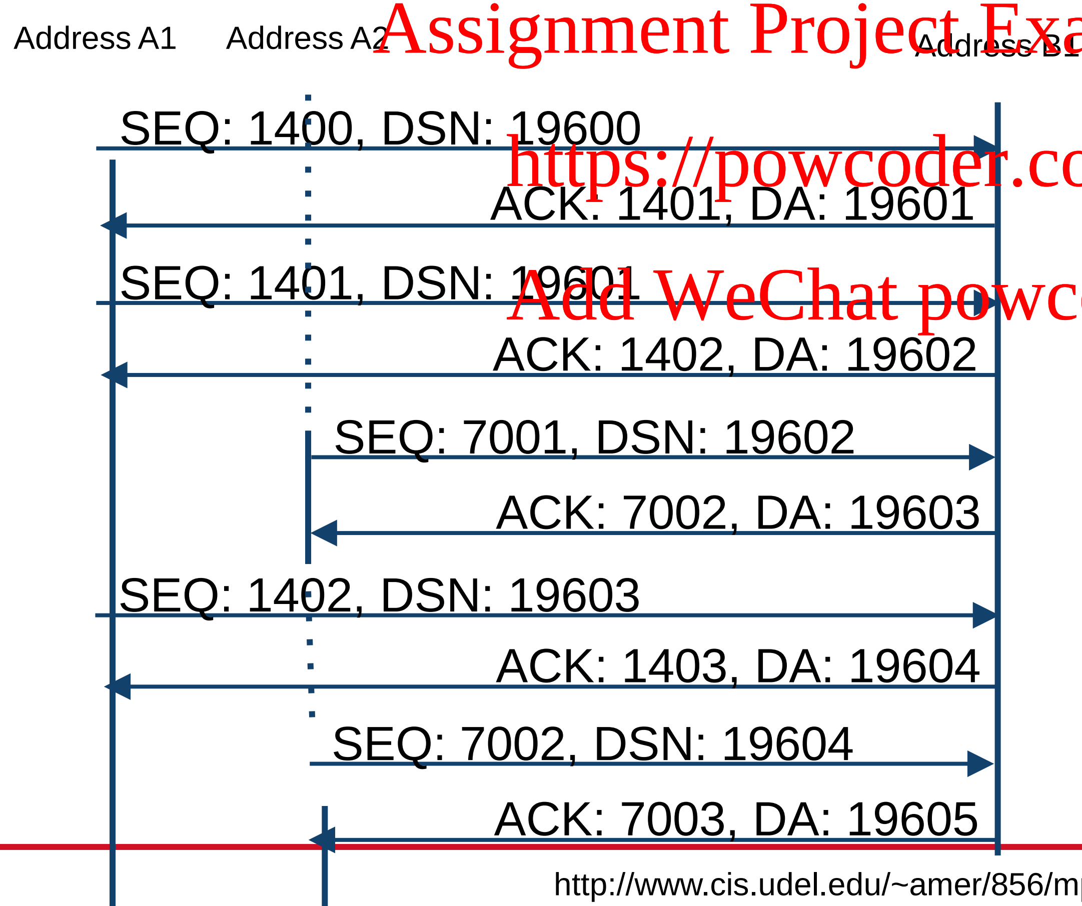
MPTCP sequence numbers

Host A

DSN	19600	19601	19602	19603	19604
subflow1	1400	1401		1402	
subflow2			7001		7002

Host B

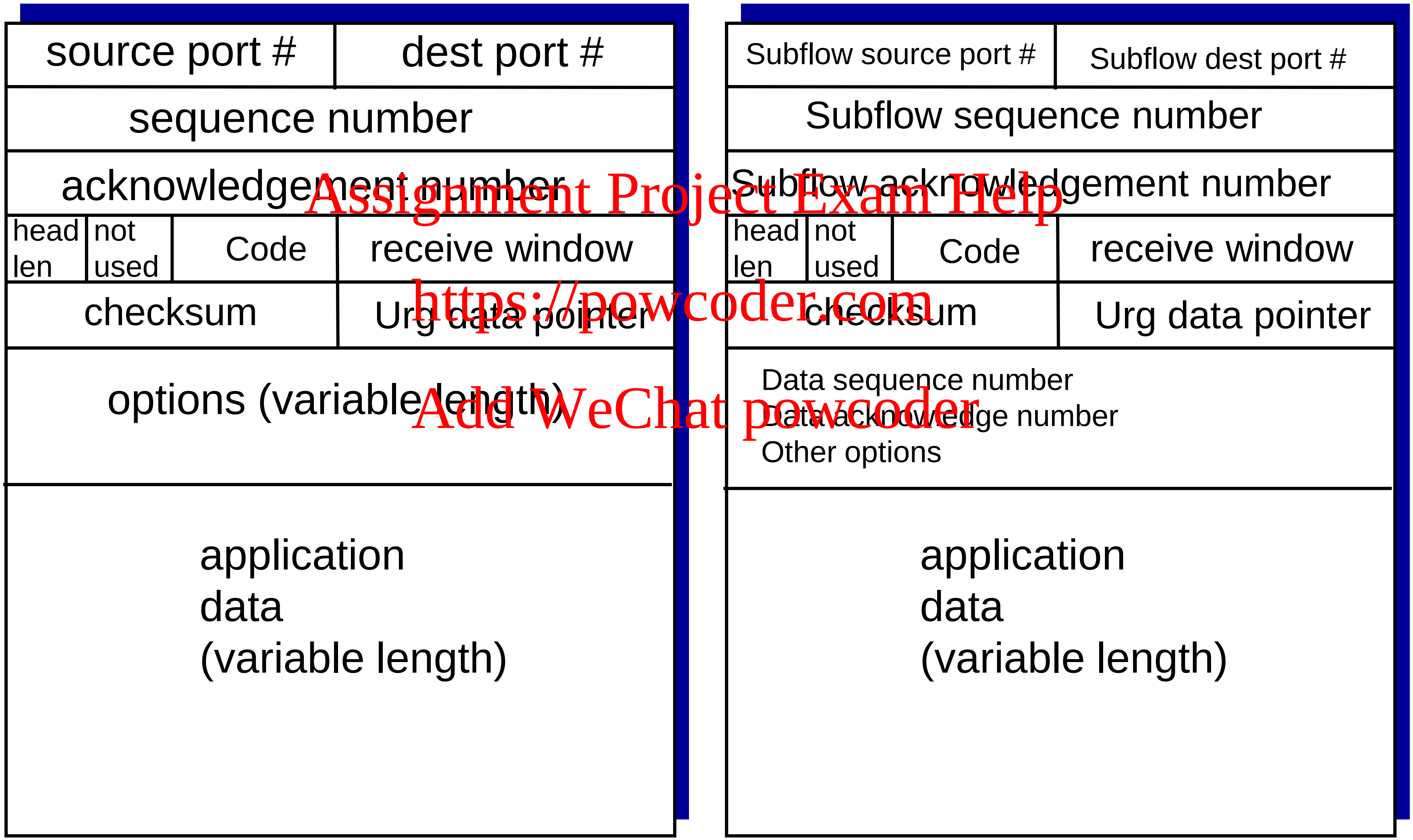
DSN	19600	19601	19602	19603	19604
subflow1	1400	1401		1402	
subflow2			7001		7002



If one subflow fails, the other subflow can be used for retransmission.

http://www.cis.udel.edu/~amer/856/mptcp.12f.pptx

MPTCP sequence numbers



Receive Window: The receive window in the TCP header indicates the amount of free buffer space for the whole data-level connection (as opposed to the amount of space for this subflow) that is available at the receiver.

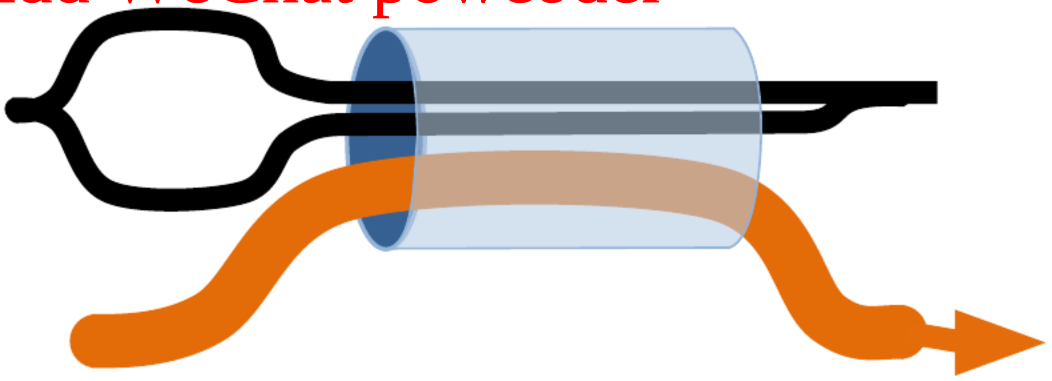
Can we run regular TCP congestion control on each subflow?

No. Not fair.

MPTCP should take as much capacity as TCP at a bottleneck link, no matter how many subflows MPTCP is using.

A MPTCP with two subflows

A regular TCP



Congestion Control

For each ACK received on subflow i, increase cwnd_i by

$$min\left(\frac{\alpha \cdot bytes\_acked \cdot MSS_i}{cwnd_{total}}, \frac{bytes\_acked \cdot MSS_i}{cwnd_i}\right)$$

$$\alpha = cwnd_{total} \cdot \frac{max_i\left(cwnd_i / {RTT_i}^2\right)}{\left(\sum_i cwnd_i / RTT_i\right)^2}$$

For each packet loss, halve the window size cwnd_i

α: aggressiveness of the multipath flow
bytes_acked: number of bytes newly acknowledged
cwnd_total: sum of the congestion windows of all subflows
(need to use ssthresh_i instead of cwnd_i if subflow is in fast retransmission)
RTT_i: round-trip time (smoothed round-trip time estimate used by TCP) of subflow i.
MSS_i: maximum segment size on subflow i
cwnd_i: congestion windows of subflow i

More details: see RFC 6356