Lecture 12b - TCP Flow Control

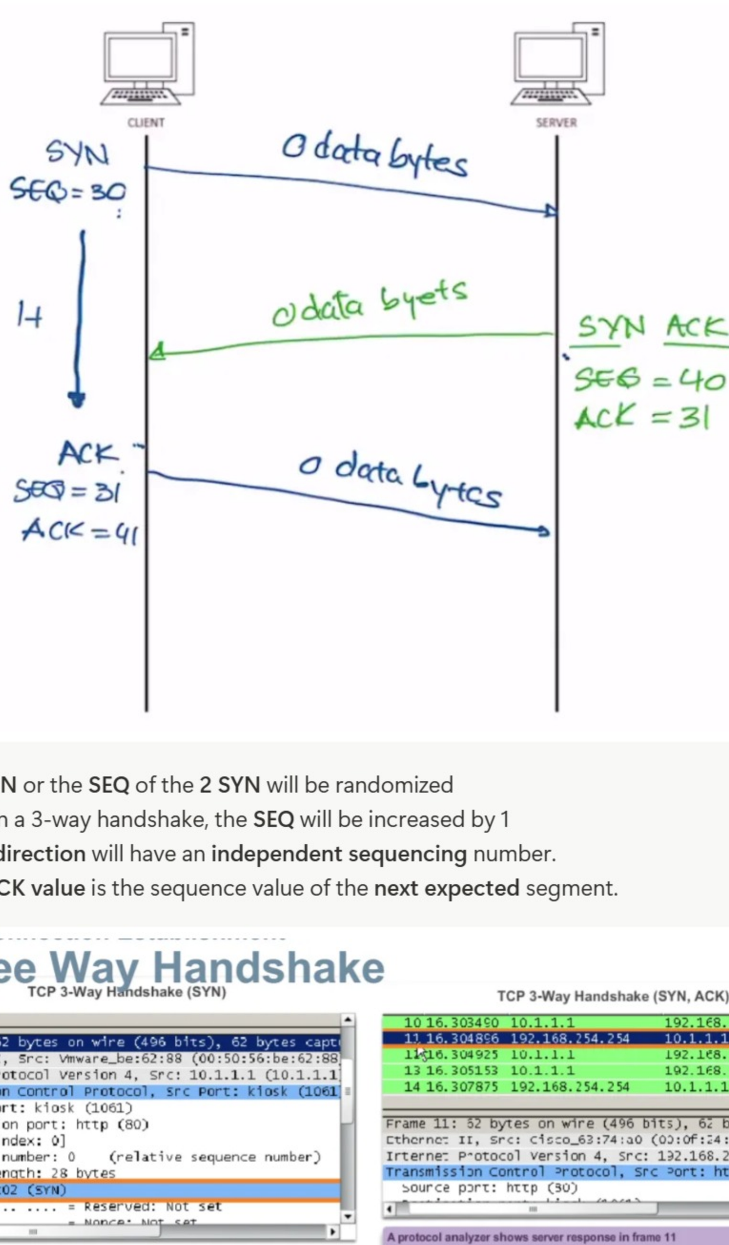
Type: Lecture

Materials: Empty

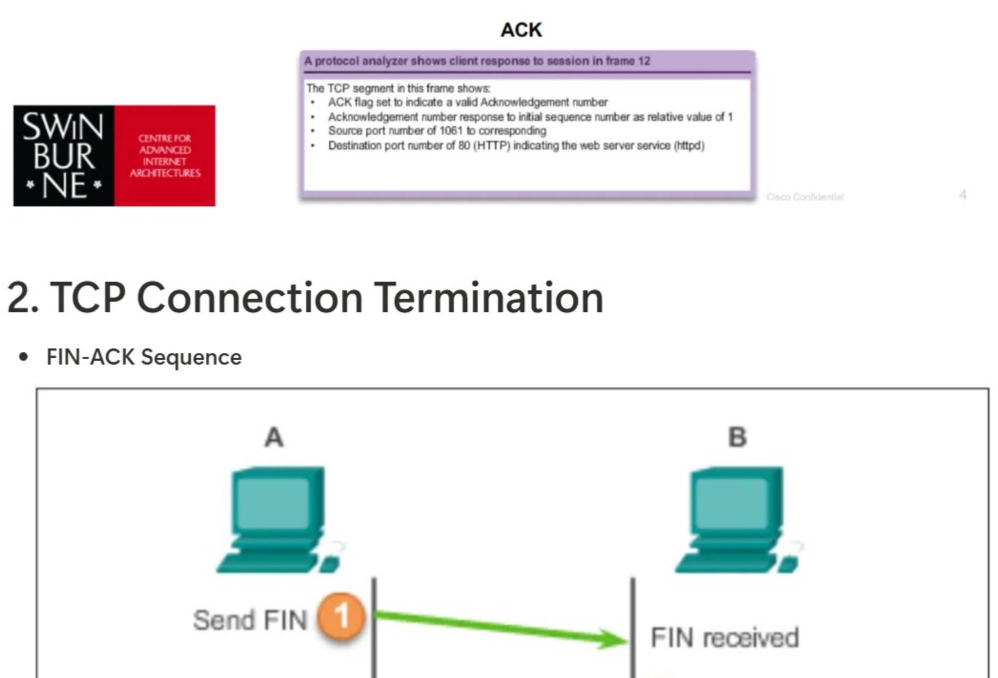
Reviewed: ☑

1. TCP Connection Establishment

- **Three-way handshake: ⇒ Connection-oriented**
    - **Establishes** that the **destination device** is **available** and has an **active service** on the **destination port number**
    - **Agrees** on **initial sequence numbers** `(ISN)` to use for **actual transmission**
    - Happens **before** any data is **transferred**

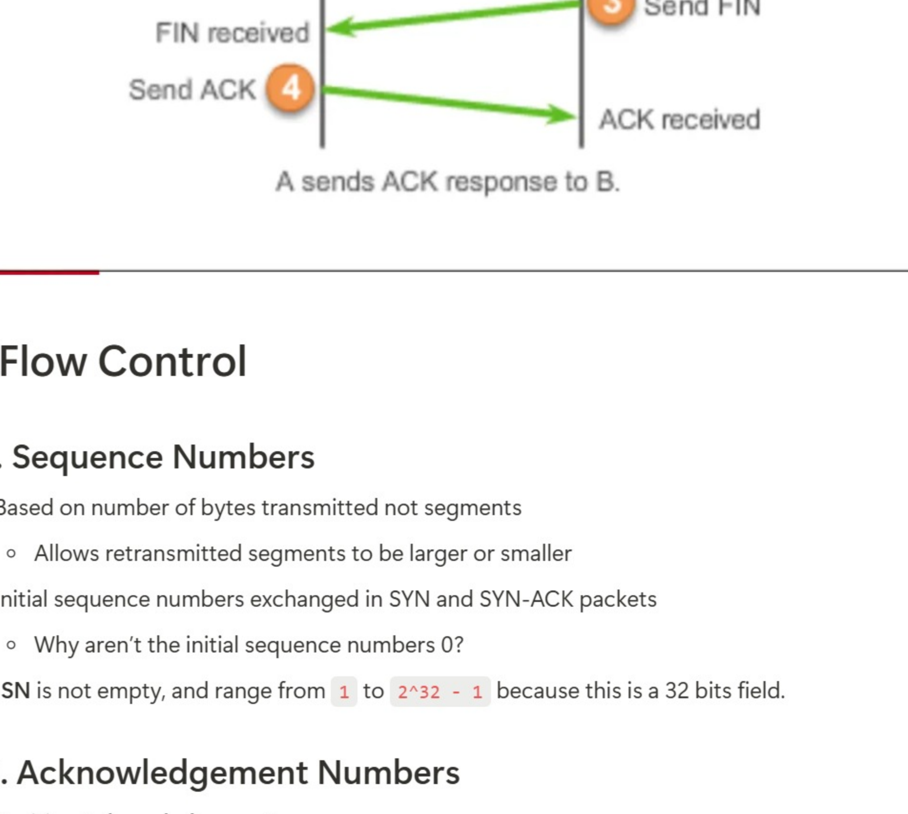


💡 - The **ISN** or the **SEQ** of the **2 SYN** will be randomized
- Only in a 3-way handshake, the **SEQ** will be increased by 1
- **Each direction** will have an **independent sequencing** number.
- The **ACK value** is the sequence value of the **next expected** segment.



2. TCP Connection Termination

- FIN-ACK Sequence



3. Flow Control

3.1. Sequence Numbers

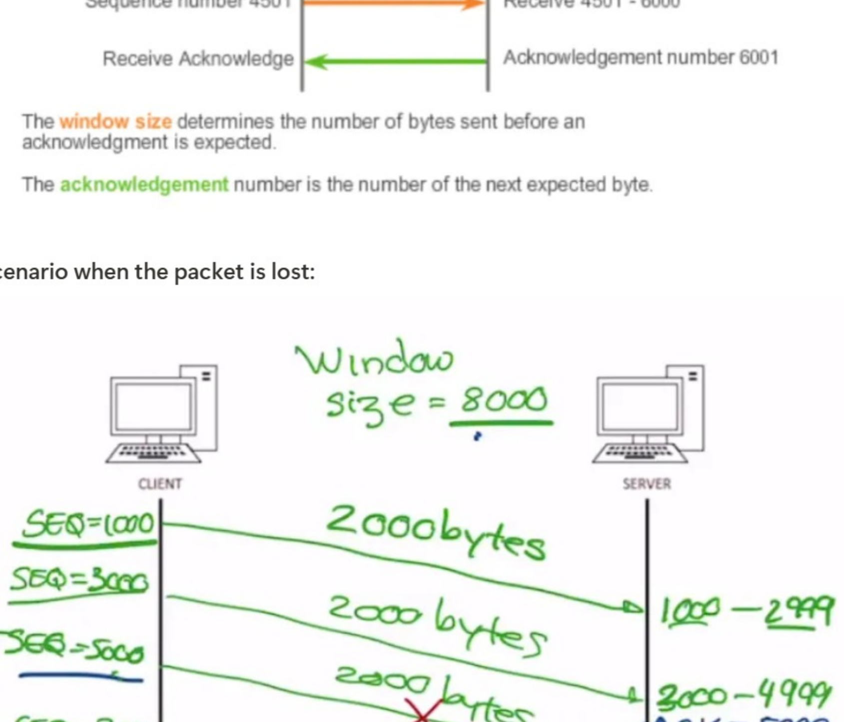
- Based on number of bytes transmitted not segments
    - Allows retransmitted segments to be larger or smaller
- Initial sequence numbers exchanged in SYN and SYN-ACK packets
    - Why aren't the initial sequence numbers 0?
- **ISN** is not empty, and range from `1` to `2^32 - 1` because this is a 32 bits field.

3.2. Acknowledgement Numbers

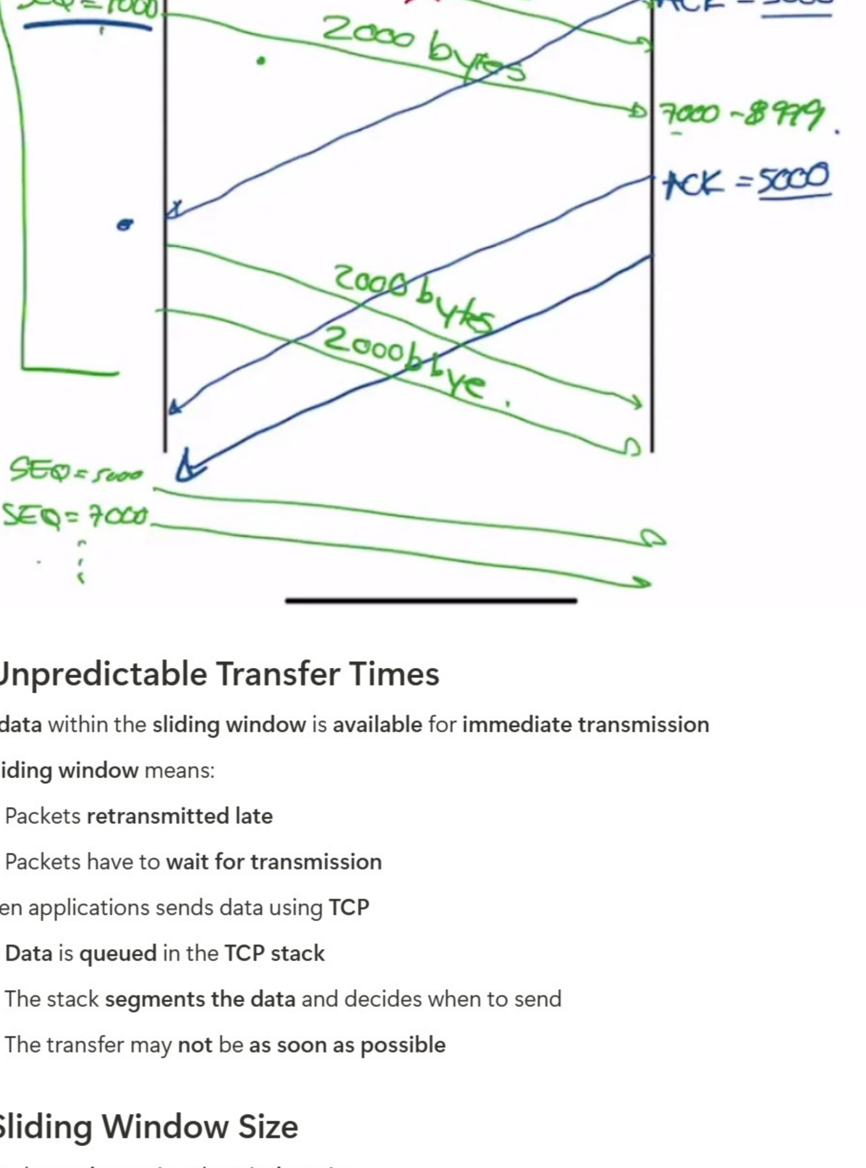
- **Positive Acknowledgement**
    - Acknowledge **next byte** we are expecting
    - Do **not** acknowledge **last byte** received
    - Included in every packet, including those with data
- **No acknowledgment** is used as a **signal of loss** and data is retransmitted

3.3. Window Size

- The packet size will depend on the **maximum transfer unit (MTU)** of the L**ayer 3** protocol
- The **Layer 3 protocol** will inform the **Layer 4** protocol how many **bytes** of data can be carried **in a single packet**? (in this example is **1500 bytes**)
- And that's how the **window size** or the **number of packets** that can be sent are determined based on the **window size** and the **maximum transfer unit**.



- **A scenario when the packet is lost:**



3.4. Unpredictable Transfer Times

- **All** data within the **sliding window** is **available** for **immediate transmission**
- A **sliding window** means:
    - Packets **retransmitted late**
    - Packets have to **wait for transmission**
- When applications sends data using **TCP**
    - **Data** is **queued** in the **TCP stack**
    - The stack **segments the data** and decides when to send
    - The transfer may **not** be **as soon as possible**

3.5. Sliding Window Size

- How do we **determine** the **window size**
    - **High-bandwidth** or **long transmission delay links** mean **more unacknowledged segments** can be in **transit** at the **same time**
    - We want to **maximize throughput**
- Window size **too small**
    - Network **bandwidth not utilised**
- Window size **too large**
    - Offering **too much data** concurrently
    - **Queues** will form
    - **Packets** will be **dropped**

3.6. Adjustable Window

- **Problem is addressed** with a Window size that is **adjustable**
- Initially TCP was programmed to **increase** the Window size **slowly** and try to **stop** at the **ideal size** to **maximise throughput**
    - If **segments** are **acknowledged**, **increase** window size
    - As soon as **first segment** is **dropped**, **fix** window size
- This has **problems**

3.7. Congestion Types

Receiver Congestion

- **End-to-End** Flow Control
- Receiver informs source of optimal sliding window size
- Receiver not swamped with more datagrams than it can handle

Network Congestion

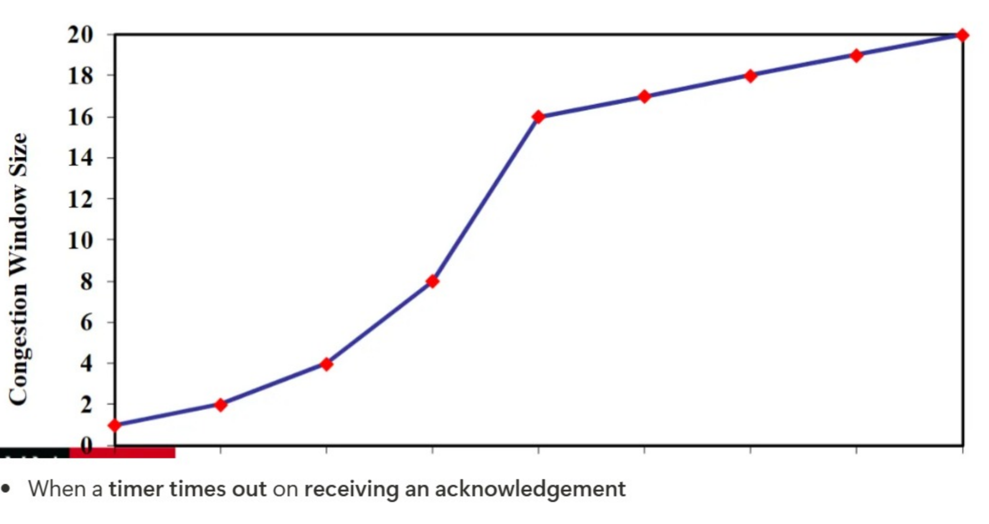
- Handled by the **Slow Start and Congestion Avoidance** algorithms
- Protocol maintains a congestion window size
- This size is modified by the Slow Start and Congestion Avoidance algorithms in response to network congestion

- **Sliding Window Size** = **min**(**receiver** window size, **congestion** window size)

3.8. Congestion Window Management

- The congestion window **size begins at one**
- When an **acknowledgement** is **received** the **window size** is **incremented by 1**
    - This causes the **window size** to **increase exponentially**!
    - Once window size **reaches** a **threshold** value or **half** the **previous maximum value** (whichever is greater):
        - **Size** is **incremented** by **one** only if **all segments** in the window are **acknowledged**
    - This **slows down** the **rate of increase**

Slow Start with Congestion Avoidance



- When a **timer times out** on **receiving an acknowledgement**
    - **Congestion window size** is **halved**
    - **Timeout** wait period is **extended**
- This has the immediate effect of **exponentially reducing transmission bit rate** and **alleviating** network **congestion**
- **Congestion** has
    - **Exponential** traffic **back-off**
    - **Exponential** traffic **rises up** to a **certain value** and then **linear**

3.9. Traffic Characteristics

- **TCP flows adjust** to
    - **Transmit** at the **available bandwidth**
    - **Share** bandwidth **evenly**
- Continuous work to define **new Congestion Control Schemes**
    - **Improve** overall **throughput without affecting TCP fairness**

4. Interaction of Traffic Classes

- TCP flows behave nicely in a congested network
- What about UDP?
- How does **UDP** traffic affect **TCP** traffic?

  ⇒ If **Too much UDP** traffic that taking too much of the network bandwidth, **TCP** will **adjust** its **window size** to the **available bandwidth**

  ⇒ Result in **slow connection**
- How does **TCP** traffic affect **UDP** traffic?

  ⇒ **TCP** has the **mechanism** to keep **increasing** the **window size** until network **congestion** is **detected**

  ⇒ Result in **UDP packet loss**.