UDP (User Datagram Protocol) - Complete Study Notes

1. Introduction to UDP

User Datagram Protocol (UDP) is one of the two main transport layer protocols in the TCP/IP suite, alongside TCP. UDP provides a simple, connectionless communication service between applications.

Key Point

In the transport layer, two protocols are critically important:

- **TCP (Transmission Control Protocol)**
- **UDP (User Datagram Protocol)**

---

2. UDP Characteristics

2.1 Connectionless Protocol

UDP is a **connectionless protocol**, which means:

- No connection establishment required before data transmission
- Data can be sent immediately without handshaking

Real-World Analogy: Postal Service vs. Telephone

TCP (Connection-oriented) = Telephone Service:

- First dial the number
- Wait for dial tone
- Establish connection
- Then start conversation

UDP (Connectionless) = Postal Service:

- Write a letter (data)
- Put it in an envelope (header)
- Drop it in the mailbox
- No prior connection needed

Example Scenario

Client wants to send data to Server:

TCP Approach:
1. Client → Server: "Can we connect?" (SYN)
2. Server → Client: "Yes, let's connect" (SYN-ACK)
3. Client → Server: "Connection established" (ACK)
4. Data transfer begins

UDP Approach:
1. Client → Server: [Data packet sent immediately]

2.2 Unreliable Protocol

Unreliable means UDP doesn't guarantee:

- Packet delivery
- Packet order
- Duplicate detection

Example Scenario

Sender sends 5 packets: [P1, P2, P3, P4, P5]

Possible UDP delivery outcomes:
✓ All packets delivered: [P1, P2, P3, P4, P5]
✗ Some packets lost: [P1, P3, P5] (P2, P4 lost)
✗ Out of order: [P3, P1, P5, P2, P4]
✗ Duplicates: [P1, P2, P2, P3, P4, P5]

UDP Response: "I don't care about these issues"

2.3 No Ordering Guarantee

UDP doesn't use **sequence numbers**, so packets can arrive out of order.

Example Scenario

Sender transmits: Packet1 → Packet2 → Packet3 → Packet4

Possible receiver sequence:
- Packet3 arrives first
- Packet4 arrives second
- Packet2 arrives third
- Packet1 arrives last

Final order: [P3, P4, P2, P1] ≠ [P1, P2, P3, P4]

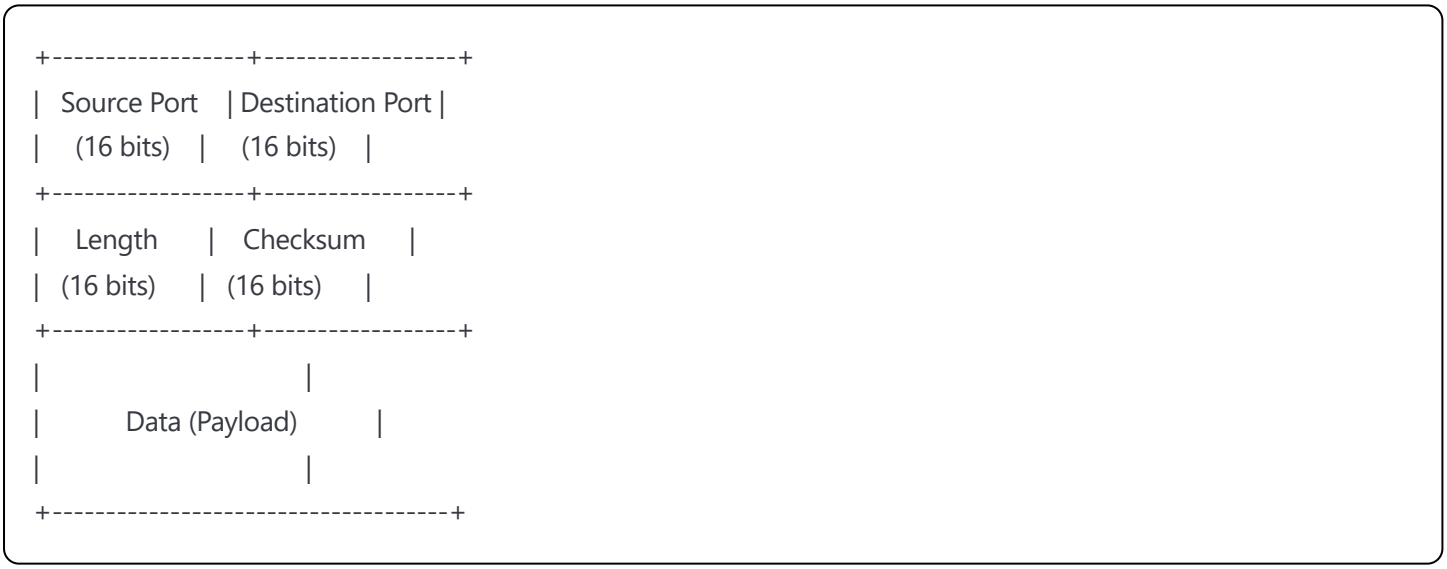
Why does this happen?

- Different packets may take different network routes
- Network congestion varies per path
- No connection state to maintain order

---

3. UDP Header Structure

UDP header contains only **4 fields** (compared to TCP's many fields):



3.1 Source Port (16 bits)

Purpose: Identifies the sending application on the source machine.

Port Number Ranges:

- **Total range:** 0 to 65,535 (2^16 - 1)
- **Well-known ports:** 0 to 1,023 (system services)
- **Registered ports:** 1,024 to 49,151 (applications)
- **Dynamic/Private ports:** 49,152 to 65,535 (temporary use)

Example Scenario

User opens web browser on their computer:

1. Browser starts → OS assigns port 52,341
2. Browser connects to Google → Source Port: 52,341
3. Multiple browser tabs → Each gets different port
   - Tab 1: Port 52,341
   - Tab 2: Port 52,342
   - Tab 3: Port 52,343

3.2 Destination Port (16 bits)

Purpose: Identifies the target application on the destination machine.

Common Service Ports:

- **HTTP:** Port 80
- **HTTPS:** Port 443
- **DNS:** Port 53
- **SMTP (Email):** Port 25
- **FTP:** Port 21

Example Scenario

Email Application Sending Mail:

Source: Your computer (Port 54,321)
Destination: Mail server (Port 25)

UDP Header:
- Source Port: 54,321
- Destination Port: 25

Critical Point: Source and Destination ports are **mandatory** for end-to-end delivery, which is the transport layer's responsibility.

3.3 Length Field (16 bits)

Purpose: Specifies the total length of the UDP segment (header + data).

Length Calculation:

- **Field size:** 16 bits

- **Maximum value:** 2^16 - 1 = 65,535 bytes
- **Represents:** Total segment length

Mathematical Example

16-bit number maximum values:
- 4 bits can represent: 2^4 = 16 numbers (0-15)
- 16 bits can represent: 2^16 = 65,536 numbers (0-65,535)

Maximum segment size = 65,535 bytes

Length Components:

Total Length = UDP Header + Payload Data

Where:
- UDP Header = Fixed 8 bytes
- Maximum Payload = 65,535 - 8 = 65,527 bytes

Example Scenarios

Scenario 1: Small message
- Data: "Hello World" (11 bytes)
- Header: 8 bytes
- Total Length: 19 bytes

Scenario 2: Maximum size
- Data: 65,527 bytes
- Header: 8 bytes
- Total Length: 65,535 bytes

Scenario 3: Cannot exceed
- If you try to send 65,528 bytes of data
- Total would be 65,536 bytes
- This exceeds 16-bit limit → Not allowed

3.4 Checksum Field (16 bits)

Purpose: Error detection mechanism to verify data integrity.

Checksum Calculation Process:

1. **Take UDP Header values:**
    - Source Port

- Destination Port
- Length
- (Checksum field set to 0 during calculation)

2. **Add Payload Data:**
   - All data bytes from application layer
3. **Add IP Pseudo Header:**
   - Source IP Address
   - Destination IP Address
   - Protocol field (UDP = 17)
   - UDP Length

Why Pseudo Header?

The pseudo header includes **fixed IP fields** that don't change during transmission:

- Source IP: Remains same throughout journey
- Destination IP: Remains same throughout journey
- Protocol: Always UDP (17)

Variable IP fields excluded: TTL, Flags, Checksum (these change at routers)

Checksum Working Example

Sender Side:
1. Combine: UDP Header + Data + IP Pseudo Header
2. Calculate hash value: 0x4A7B
3. Insert 0x4A7B in checksum field
4. Send packet

Receiver Side:
1. Receive packet with checksum: 0x4A7B
2. Extract same components: UDP Header + Data + IP Pseudo Header
3. Calculate hash value: 0x4A7B
4. Compare: 0x4A7B == 0x4A7B ✓ No error detected

Error Scenario:
1. Packet corrupted during transmission
2. Receiver calculates: 0x3F82
3. Compare: 0x4A7B != 0x3F82 ✗ Error detected

Checksum Behavior Differences:

- **IPv4:** Checksum is **OPTIONAL**
- **IPv6:** Checksum is **MANDATORY**

Example: Optional Checksum in IPv4

Valid UDP packets in IPv4:
✓ With checksum: Normal error detection
✓ Without checksum: Checksum = 0, no error detection

Invalid in IPv6:
✗ Must always include checksum

---

4. UDP vs TCP Routing Behavior

4.1 TCP Connection-Based Routing

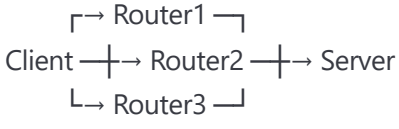
TCP Established Connection:
Client ←→ Router1 ←→ Router2 ←→ Server

All packets follow SAME path:
Packet1: Client → Router1 → Router2 → Server
Packet2: Client → Router1 → Router2 → Server
Packet3: Client → Router1 → Router2 → Server

Result: Ordered, reliable delivery

4.2 UDP Connectionless Routing

UDP Independent Routing:



Different packets follow DIFFERENT paths:
Packet1: Client → Router1 → Server
Packet2: Client → Router3 → Server
Packet3: Client → Router2 → Server

Result: Possible packet loss, out-of-order delivery

---

5. Real-World Applications of UDP

5.1 When to Use UDP Despite Its "Unreliability"

Suitable Applications:

1. **Live Video Streaming**
   - Lost frames acceptable
   - Speed more important than perfection
   - Example: YouTube Live, Twitch
2. **Online Gaming**
   - Player position updates
   - Old position data becomes irrelevant quickly
   - Example: First-person shooters
3. **DNS Queries**
   - Small request/response
   - Can retry if needed
   - Speed critical
4. **IoT Sensor Data**
   - Continuous data stream
   - Missing one reading acceptable
   - Example: Temperature sensors

Example Gaming Scenario:

```
Player Movement Updates:
Time 0ms: Player at position (10, 20)
Time 10ms: Player at position (12, 25)
Time 20ms: Player at position (15, 30)

If 10ms packet lost:
- Still have current position (15, 30)
- Old position (12, 25) not needed
- Game continues smoothly

UDP Perfect Choice: Speed > Reliability
```

---

6. Complete UDP Segment Example

Example: DNS Query

Application: Web browser resolving "google.com"
Source: Computer (192.168.1.100:54321)
Destination: DNS Server (8.8.8.8:53)

UDP Header (8 bytes):

```
+----------------+----------------+
| 54321 (0xD431) | 53 (0x0035)    | Source/Dest Port
+----------------+----------------+
| 40 (0x0028)    | 0x1A2B         | Length/Checksum
+----------------+----------------+
```

Payload (32 bytes): DNS query for "google.com"

Total segment: 8 + 32 = 40 bytes

Checksum Calculation Detail:

Components for checksum:
1. UDP Header: 54321 + 53 + 40 + 0
2. DNS Query Data: [32 bytes of query]
3. IP Pseudo Header:
   - Source IP: 192.168.1.100
   - Dest IP: 8.8.8.8
   - Protocol: 17 (UDP)
   - Length: 40

Hash Result: 0x1A2B (example)

---

7. Summary and Key Takeaways

UDP Characteristics Summary:

Feature	UDP	TCP
Connection	Connectionless	Connection-oriented
Reliability	Unreliable	Reliable
Ordering	No guarantee	Guaranteed
Header Size	8 bytes (fixed)	20+ bytes (variable)
Speed	Fast	Slower
Error Recovery	None	Automatic

When to Choose UDP:

- ✅ Speed is critical
- ✅ Some data loss acceptable
- ✅ Real-time applications
- ✅ Simple request-response
- ✅ Broadcast/multicast needed

When to Avoid UDP:

- ❌ Data integrity critical
- ❌ File transfers
- ❌ Financial transactions
- ❌ Email delivery
- ❌ Web page loading

Memory Aid: "UDP = Unreliable, Datagram, Protocol"

- **U**nreliable: No delivery guarantee
- **D**atagram: Independent packet delivery
- **P**rotocol: Transport layer service

---

8. Practice Questions

1. **Calculate maximum payload size if UDP header is 8 bytes and total segment size is limited by 16-bit length field.**
2. **Explain why UDP checksum includes IP pseudo header fields.**
3. **Give three real-world scenarios where UDP is preferred over TCP and justify each choice.**
4. **If a UDP packet has source port 8080, destination port 53, total length 64 bytes, what is the payload size?**

Answers:

1. Max payload = 65,535 - 8 = 65,527 bytes
2. To detect errors that might occur in IP layer fields that shouldn't change
3. Gaming (speed), streaming (acceptable loss), DNS (simple query)
4. Payload = 64 - 8 = 56 bytes