Transport Layer: UDP (User Datagram Protocol)
Study-Ready Notes

Compiled by Andrew Photinakis

October 17, 2025

Contents

1 Introduction to Transport Layer

- Transport-layer services provide end-to-end communication between applications
- Key functions include multiplexing and demultiplexing
- Two main transport protocols: UDP (connectionless) and TCP (connection-oriented)
- Additional topics: reliable data transfer, congestion control principles

[Summary: The transport layer enables application-to-application communication across networks, with UDP providing lightweight connectionless service and TCP offering reliable connection-oriented service.]

2 UDP: User Datagram Protocol

2.1 Basic Characteristics

- **"No frills," "bare bones"** Internet transport protocol
- **Best effort service** - segments may be:
  - Lost during transmission
  - Delivered out-of-order to application
- **Connectionless** - no handshaking between UDP sender and receiver
- Each UDP segment handled independently of others

2.2 Why Use UDP?

- **No connection establishment** - avoids RTT delay
- **Simple** - no connection state maintained at sender or receiver
- **Small header size** - minimal overhead
- **No congestion control** - can transmit as fast as desired
- **Robust** - can function even when network service is compromised

[Mnemonic: UDP = Uncomplicated, Direct, Prompt - emphasizes its simplicity and speed]

3 UDP Applications and Use Cases

3.1 Common UDP Applications

- **Streaming multimedia apps** - loss tolerant, rate sensitive
- **DNS (Domain Name System)** - quick name resolution
- **SNMP (Simple Network Management Protocol)** - network monitoring
- **HTTP/3** - modern web protocol built on UDP

3.2 Reliability Over UDP

- If reliable transfer needed (e.g., HTTP/3):
  - Add reliability mechanisms at application layer
  - Implement congestion control at application layer
- Applications can customize reliability to their specific needs

4 UDP Protocol Specification [RFC 768]

4.1 Protocol Definition

- Standardized in RFC 768 (August 28, 1980)
- Provides datagram mode of packet-switched communication
- Assumes Internet Protocol (IP) as underlying protocol
- Transaction-oriented with minimal protocol mechanism

4.2 Service Guarantees

- **No guaranteed delivery** - packets may be lost
- **No duplicate protection** - may receive copies
- **No ordered delivery** - out-of-order arrival possible
- Applications needing reliable streams should use TCP instead

5 UDP Segment Format

5.1 Header Structure

- 32-bit (4-byte) header followed by data payload
- Header fields:
  - **Source Port** (16 bits) - sender's port number
  - **Destination Port** (16 bits) - receiver's port number
  - **Length** (16 bits) - total segment size in bytes (header + data)
  - **Checksum** (16 bits) - error detection field

5.2 Format Visualization

```
0               16              31
+---------------+---------------+
|  Source Port  | Destination Port|
+---------------+---------------+
|    Length     |   Checksum    |
+---------------+---------------+
|                                |
|          Data Payload         |
|                                |
+-------------------------------+
```

[Concept Map: UDP Segment → Header (Ports + Length + Checksum) + Data → IP Packet → Network Transmission]

6 UDP Transport Layer Actions

6.1 Sender Actions

1. Receives application-layer message
2. Determines UDP segment header field values
3. Creates UDP segment with header and data
4. Passes segment to IP layer for transmission

6.2 Receiver Actions

1. Receives segment from IP layer
2. Verifies UDP checksum header value

3. Extracts application-layer message from payload
4. Demultiplexes message to appropriate application via socket

6.3 SNMP Example

- SNMP client creates SNMP message
- UDP layer adds header and passes to IP
- SNMP server receives via IP, checks checksum, delivers to application

7 UDP Checksum Mechanism

7.1 Goal and Purpose

- **Primary goal**: Detect errors (flipped bits) in transmitted segments
- Provides basic data integrity verification
- Covers UDP header, data, and pseudo-IP header information

7.2 Sender Procedure

1. Treat UDP segment contents as sequence of 16-bit integers
2. Include UDP header fields and IP addresses in calculation
3. Compute one's complement sum of all 16-bit integers
4. Place resulting checksum value in UDP checksum field

7.3 Receiver Procedure

1. Compute checksum of received segment using same method
2. Compare computed checksum with received checksum field
3. **Not equal**: Error detected - segment may be corrupted
4. **Equal**: No error detected (but errors still possible due to checksum limitations)

8 Internet Checksum: Detailed Example

8.1 Calculation Process

- Add two 16-bit integers using one's complement arithmetic
- Example:
  - First number: 1110 0110 0110 0110 (binary)
  - Second number: 1101 0101 0101 0101 (binary)
  - Sum: 1101 1101 1101 1011 (with carry)
- **Wraparound**: Carry from most significant bit added back to result

8.2 Mathematical Representation

$$\begin{array}{cccccccccccccccc}
1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
\hline
1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\
1 & & & & & & & & & & & & & & &
\end{array}$$

8.3 Checksum Weaknesses

- **Limited error detection capability**
- May not detect certain error patterns (e.g., swapped bytes)
- Example: Even when numbers change due to bit flips, checksum may remain unchanged
- Not cryptographically secure - easy to forge

[Summary: UDP checksum provides basic error detection using one's complement addition, but has limitations and cannot guarantee complete data integrity.]

9 Comprehensive UDP Summary

9.1 Protocol Characteristics

- **"No frills"** protocol with minimal overhead
- Segments may be lost or delivered out-of-order
- Best effort service: "send and hope for the best"

9.2 UDP Advantages

- No setup/handshaking required (avoids RTT delay)
- Can function when network conditions are poor
- Provides basic error detection via checksum
- Enables application-specific reliability implementations

9.3 Modern Applications

- Foundation for HTTP/3 and other modern protocols
- Allows building customized transport functionality
- Ideal for applications preferring timeliness over reliability

[Mnemonic: UDP Benefits = FAST - Fast setup, Application control, Simple, Timely delivery]

10 Study Questions

10.1 Conceptual Questions

1. Compare and contrast UDP and TCP in terms of reliability, connection establishment, and overhead.
2. Explain why streaming applications often prefer UDP over TCP.
3. Describe the UDP checksum calculation process and its limitations.
4. What are the advantages of implementing reliability at the application layer rather than using TCP?

10.2 Technical Problems

1. Calculate the UDP checksum for a segment with source port 5200, destination port 53, length 32 bytes, and data ”Hello”.
2. Explain what happens when a UDP receiver detects a checksum error.
3. Design an application-layer protocol that adds reliability to UDP for file transfer.

10.3 Real-World Applications

1. Research and explain why HTTP/3 uses UDP instead of TCP.
2. Identify three real-world applications that use UDP and explain why it's appropriate for each.
3. Analyze the trade-offs between using raw UDP vs. building custom reliability vs. using TCP.

[Concept Map: Transport Layer → UDP vs TCP → UDP: Connectionless + Best Effort + Applications (DNS, Streaming) + Checksum → Modern Uses (HTTP/3)]