OSI Transport Layer

Network Fundamentals – Chapter 4
Objectives

- Explain the role of Transport Layer protocols and services in supporting communications across data networks

- Analyze the application and operation of TCP mechanisms that support reliability, reassembly and manage data loss.

- Analyze the operation of UDP to support communicate between two processes on end devices
Transport Layer Role and Services

- The purpose of the Transport layer
Transport Layer Role and Services

- Provides the end-to-end transfer of data between applications.
Role of the Transport layer

- Encapsulating application data for use by the Network layer
- Enables multiple applications to communicate over the network at the same time on a single device
- Ensures that, if required, all the data is received reliably and in order by the correct application
- Employs error handling mechanisms
Responsibilities

- **Tracking** the individual communication between *applications* on the source and destination hosts *(Establishing a Session)*

- **Segmenting** data and managing each piece

- **Reassembling** the segments into streams of application data

- Identifying the **different** applications *(Port Number)*
Two TCP/IP Transport layer protocols

- There are multiple Transport layer protocols. Why?
  Because different applications have different requirements.
The two most common Transport layer protocols of TCP/IP protocol suite are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). Both protocols manage the communication of multiple applications. The differences between the two are the specific functions that each protocol implements.
Two TCP/IP Transport layer protocols:
TCP and UDP.

Some protocols TCP at the Transport layer provide:

- Connection-oriented conversations
- Reliable delivery
- Ordered data reconstruction
- Flow control
Basic characteristics of the UDP and TCP protocols

- Fast
- Low overhead
- Does not require acknowledgements
- Does not resend lost data
- Delivers data as it arrives

- Reliable
- Acknowledges data
- Resends lost data
- Delivers data in the order sent
Transport Layer Role and Services

- Supporting Reliable Communication

Transport Layer Protocols

- IP Telephony
- Streaming Video
- SMTP/POP (Email)
- HTTP

TCP/IP Model

OSL Model
- Application
- Presentation
- Session
- Transport
- Network
- Data Link
- Physical

Application

Network Access

Required Protocol Properties
- Fast
- Low overhead
- Does not require acknowledgements
- Does not resend lost data
- Delivers data as it arrives

Required Protocol Properties
- Reliable
- Acknowledge data
- Resend lost data
- Delivers data in order sent

Application developers choose the appropriate Transport Layer protocol based on the nature of the application.
UDP

- **UDP** is a simple, **connectionless** protocol
- **low overhead** data delivery.
- **datagrams** are sent as "**best effort**" by this Transport layer protocol.
Applications that use UDP include:

- Domain Name System (DNS)
- Video Streaming
- Voice over IP (VoIP)
Transmission Control Protocol (TCP)

- Connection-oriented protocol,
- Overhead
- Low control

**flow control**

Flow control is the management of data flow between devices in a network. It is used to avoid too much data arriving before a device can handle it, causing data overflow.
Applications that use TCP are:

- Web Browsers
- E-mail
- File Transfers
Transport Layer Headers

- segment or datagram?

<table>
<thead>
<tr>
<th>TCP SEGMENT &amp; HEADER FIELDS</th>
<th>Bit 0</th>
<th>Bit 15 Bit 16</th>
<th>Bit 31</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Port (16)</td>
<td></td>
<td>Destination Port (16)</td>
<td></td>
</tr>
<tr>
<td>Sequence Number (32)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Acknowledgement Number (32)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Header Length (4) Reserved (6) Code Bits (6)</td>
<td></td>
<td>Window (16)</td>
<td></td>
</tr>
<tr>
<td>Checksum (16)</td>
<td></td>
<td>Urgent (16)</td>
<td></td>
</tr>
<tr>
<td>Options (0 or 32 if any)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>APPLICATION LAYER DATA SEGMENT (Size varies)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UDP SEGMENT &amp; HEADER FIELDS</th>
<th>Bit (0)</th>
<th>Bit (15) Bit (16)</th>
<th>Bit (31)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Port (16)</td>
<td></td>
<td>Destination Port (16)</td>
<td></td>
</tr>
<tr>
<td>Length (16)</td>
<td></td>
<td>Checksum (16)</td>
<td></td>
</tr>
<tr>
<td>APPLICATION LAYER DATA SEGMENT (Size varies)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Port Number, Identifying the Conversations

- Port numbers are assigned in various ways, depending on whether the message is a request or a response. While server processes have **static port numbers** assigned to them, **clients dynamically choose** a port number for each conversation.
Port Number

- When a client application sends a request to a server application, the destination port contained in the header is the port number that is assigned to the service daemon running on the remote host. The client software must know what port number is associated with the server process on the remote host. This destination port number is configured, either by default or manually. For example, when a web browser application makes a request to a web server, the browser uses TCP and port number 80 unless otherwise specified. This is because TCP port 80 is the default port assigned to web-serving applications. Many common applications have default port assignments.
The source port in a segment or datagram header of a client request is randomly generated from port numbers greater than 1023.

The combination of the Transport layer port number and the Network layer IP address assigned to the host uniquely identifies a particular process running on a specific host device. This combination is called a socket.

192.168.1.20:80.
Port Number

Data for different applications is directed to the correct application because each application has a unique port number.
There are different types of port numbers:

- **Well Known** Ports (Numbers 0 to 1023)
  These numbers are reserved for **services and applications**. They are commonly used for applications such as HTTP (web server) POP3/SMTP (e-mail server) and Telnet.

- **Registered** Ports (Numbers 1024 to 49151)
  These port numbers are assigned to **user** processes or applications. These processes are primarily individual applications that a user has chosen to install rather than common applications that would receive a Well Known Port. When not used for a server resource, these ports may also be used dynamically selected by a client as its source port.
**Dynamic or Private Ports** (Numbers 49152 to 65535) these are usually assigned dynamically to client applications when initiating a connection.
Using both TCP and UDP

- Some applications may use both TCP and UDP. For example, the low overhead of UDP enables DNS to serve many client requests very quickly. Sometimes, however, sending the requested information may require the reliability of TCP. In this case, the well known port number of 53 is used by both protocols with this service.
TCP Ports
**UDP Ports**

<table>
<thead>
<tr>
<th>Port Number Range</th>
<th>Port Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 to 1023</td>
<td>Well Known (Contact) Ports</td>
</tr>
<tr>
<td>1024 to 49151</td>
<td>Registered Ports</td>
</tr>
<tr>
<td>49152 to 65535</td>
<td>Private and/or Dynamic Ports</td>
</tr>
</tbody>
</table>

**Registered UDP Ports:**
- 1812 RADIUS Authentication Protocol
- 5004 RTP (Voice and Video Transport Protocol)
- 5060 SIP (VoIP)

**Well Known UDP Ports:**
- 88 TFTP
- 520 RIP
Netstat

- Netstat lists the protocol in use, the local address and port number, the foreign address and port number, and the state of the connection.
Segments

- Dividing application data into pieces both ensures that data is transmitted within the limits of the media and that data from different applications can be multiplexed on to the media.
Segments

The Transport layer divides the data into pieces and adds a header for delivery over the network.

**Transport Layer Functions**

**APPLICATION LAYER DATA**

<table>
<thead>
<tr>
<th>Piece 1</th>
<th>Piece 2</th>
<th>Piece 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Datagram</td>
<td>Or</td>
<td>TCP Segment</td>
</tr>
</tbody>
</table>

**UDP Header provides for:**
- Source and destination (ports)

**TCP Header provides for:**
- Source & destination (ports)
- Sequencing for same order delivery
- Acknowledgement of received segments
- Flow control and congestion management

© 2007 Cisco Systems, Inc. All rights reserved.
TCP Segment Header

The fields of the TCP header enable TCP to provide connection-oriented, reliable data communications.
Application and Operation of TCP Mechanisms
- port numbers in establishing TCP sessions and directing segments to server process
TCP Connection Establishment and Termination

- When two hosts communicate using TCP, a connection is established before data can be exchanged.
- After the communication is completed, the sessions are closed and the connection is terminated.
- The connection and session mechanisms enable TCP's reliability function.
Handshake in the establishment of TCP sessions

TCP Connection Establishment and Termination

1. Send SYN
   (SEQ=100 CTL=SYN)

2. SYN received
   Send SYN, ACK
   (SEQ=300 ACK=101 CTL=SYN, ACK)

3. SEQ=101 ACK=301 CTL=ACK
   Established

CTL = Which control bits in the TCP header are set to 1
Handshake in the termination of TCP sessions

TCP Connection Establishment and Termination

1. Send FIN
2. FIN received
3. Send ACK
4. ACK received
5. Send FIN
6. FIN received
7. Send ACK
8. ACK received
TCP 3 way handshake

1. Send SYN
   
   (SEQ=100 CTL=SYN)

2. SYN received

3. CTL = Which control bits in the TCP header are set to 1

   A sends SYN request to B.
2

1. Send SYN
   (SEQ=100 CTL=SYN)

2. SYN received
   Send SYN, ACK
   (SEQ=300 ACK=101 CTL=SYN, ACK)

CTL = Which control bits in the TCP header are set to 1

B sends ACK response and SYN request to A.
3

**Diagram:**

1. Send SYN (SEQ=100 CTL=SYN)
2. SYN received
3. SYN received
4. Send SYN,ACK (SEQ=300 ACK=101 CTL=SYN,ACK)
5. ESTABLISHED

**CTL = Which control bits in the TCP header are set to 1**

A sends ACK response to B.
- The sequence number is the relative number of bytes that have been transmitted in this session plus 1.

- acknowledgement number indicate the next byte in this session that the receiver expects to receive.
Managing TCP Sessions

- TCP sequence numbers are used to reconstruct the data stream with segments placed in the correct order.
Managing TCP Sessions

- Trace the steps used by the TCP protocol in which sequence numbers and acknowledgement numbers are used to manage exchanges in a conversation.
The amount of data that a source can transmit before an acknowledgement must be received is called the window size. **Window Size** is a field in the TCP header that enables the management of lost data and flow control.
TCP Retransmission

- For example, if segments with sequence numbers 1500 to 3000 and 3400 to 3500 were received, the acknowledgement number would be 3001. This is because there are segments with the sequence numbers 3001 to 3399 that have not been received.

- When TCP at the source host has not received an acknowledgement after a predetermined amount of time, it will go back to the last acknowledgement number that it received and retransmit data from that point forward.
Flow Control

- window size, and congestion during a session

TCP Congestion and Flow Control

Sender

- Sequence number 1
- Sequence number 1501
- Receive Acknowledge
- Sequence number 3001
- Sequence number 4501
- Receive Acknowledge

Window size = 3000

1500 bytes
1500 bytes

Receive 1501 - 3000
Receive 1501 - 3000
Acknowledgement number 3001

X

Receiver

- Receive 1501 - 3000
- Acknowledgement number 3001
- Window size = 1500

If segments are lost because of congestion, the Receiver will acknowledge the last received sequential segment and reply with a reduced window size.
UDP Protocol

UDP Low Overhead Data Transport

UDP does not establish a connection before sending data.
- UDP is a simple protocol that provides the basic Transport layer functions.

- It has a much lower overhead than TCP, since it is not connection-oriented and does not provide the sophisticated retransmission, sequencing, and flow control mechanisms.
key Application layer protocols that use UDP include:

- Domain Name System (DNS)
- Simple Network Management Protocol (SNMP)
- Dynamic Host Configuration Protocol (DHCP)
- Routing Information Protocol (RIP)
- Trivial File Transfer Protocol (TFTP)
- Online games
UDP Reassembles

- UDP does not keep track of sequence numbers the way TCP does. UDP has no way to reorder the datagrams into their transmission order.

- UDP simply reassembles the data in the order that it was received and forwards it to the application.

- If the sequence of the data is important to the application, the application will have to identify the proper sequence of the data and determine how the data should be processed.
UDP Protocol

Different datagrams may take different routes.

Data

Data is divided into datagrams.

Datagram 1
Datagram 2
Datagram 3
Datagram 4
Datagram 5
Datagram 6

Having taken different routes to the destination, datagrams arrive out of order.

Datagram 1
Datagram 2
Datagram 5
Datagram 6

Out of order datagrams are not re-ordered.

Datagram 4

Lost datagrams are not re-sent.
UDP Protocol – Port Number

- servers use port numbers to identify a specified application layer process and direct segments to the proper service or application.
UDP Protocol

- Trace the steps as the UDP protocol and port numbers are utilized in client-server communication.

Clients Sending UDP Requests

Server DNS response:
- Source Port 53
- Destination Port 49152

Server RADIUS Response:
- Source Port 1812
- Destination Port 51152

DNS: Port 53
RADIUS: Port 1812

Server response to UDP clients use well-known port numbers as the source port.

Client 1 waiting for server DNS response on Port 49152

Client 2 waiting for server RADIUS response on Port 51152
Quiz

At the transport layer, which of the following controls is used to avoid a transmitting host overflowing the buffers of a receiving host?

- [ ] best effort
- [ ] encryption
- [ ] flow control
- [ ] compression
- [ ] congestion avoidance
End systems use port numbers to select the proper application. What is the smallest port number that can be dynamically assigned by a host system?

- 1
- 64
- 128
- 256
- 512
- 1024
During data transfer, what are the main responsibilities of the receiving host? (Choose two.)

- throughput
- encapsulation
- acknowledgment
- bandwidth
- segmentation
- reassembly
At which layer of the TCP/IP model does TCP operate?

- session
- transport
- network
- data link
What determines how much data a sending station running TCP/IP can transmit before it must receive an acknowledgment?

- segment size
- transmission rate
- bandwidth
- window size
- sequence number
What is the purpose of the sequence number in the TCP header?

- reassemble the segments into data
- identify the application layer protocol
- indicate the number of the next expected byte
- show the maximum number of bytes allowed during a session
Which acknowledgement number should be sent by the receiver shown in the graphic?

Sender:
- send 1
- send 2
- send 3
- send 4
- send 5
- send 6

Receiver:
- receive 1
- receive 2
- receive 3
- receive 4
- receive 5
- receive 6
- send ACK ___

Options:
- 3
- 4
- 6
- 7
What is the purpose of TCP/UDP port numbers?

- indicate the beginning of a three-way handshake
- reassemble the segments into the correct order
- identify the number of data packets that may be sent without acknowledgment
- track different conversations crossing the network at the same time