Chapter 11

User Datagram Protocol (UDP)
CONTENTS

• PROCESS-TO-PROCESS COMMUNICATION
• USER DATAGRAM
• CHECKSUM
• UDP OPERATION
• USE OF UDP
• UDP PACKAGE
Position of UDP in the TCP/IP protocol suite

- **Application layer**
  - SMTP
  - FTP
  - TFTP
  - DNS
  - SNMP
  - ... (ellipses)
  - BOOTP

- **Transport layer**
  - TCP
  - UDP

- **Network layer**
  - IGMP
  - ICMP
  - IP
  - ARP
  - RARP

- **Data link layer**

- **Physical layer**
  - Underlying LAN or WAN technology
UDP Services

- Process-to-process communication using “Ports.”
  Fundamentally, this is the only thing it adds on top of IP.
- Control:
  - No flow control or acknowledgment
  - Minimal error control: simply detect and drop.
- Connectionless & Unreliable transport.
- Minimal overhead, compared to TCP
11.1 PROCESS TO PROCESS COMMUNICATION

UDP versus IP

Processes (Running application programs)

Internet

Processes (Running application programs)

Domain of IP protocol

Domain of UDP protocol
Client-Server Paradigm & Ports

- A *Client* process on a local host sends a request to a *Server* process on a remote host.
- Client process identification:
  - Local host IP
  - *Ephemeral port number* (0-65,535) randomly chosen by UDP
- Server process identification
  - Remote host IP
  - *Well-known* port number
Role of IP addresses vs. Port numbers

- IP address selects the host
- Port number selects the process

Diagram showing the relationship between IP addresses and port numbers in network communication.
IANA Port Number Ranges

<table>
<thead>
<tr>
<th>Port</th>
<th>Protocol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Echo</td>
<td>Echoes a received datagram back to sender</td>
</tr>
<tr>
<td>13</td>
<td>Daytime</td>
<td>Returns the date and time</td>
</tr>
<tr>
<td>111</td>
<td>RPC</td>
<td>Remote procedure call</td>
</tr>
<tr>
<td>123</td>
<td>NTP</td>
<td>Network Time Protocol</td>
</tr>
<tr>
<td>161</td>
<td>SNMP</td>
<td>Simple Network Management Protocol</td>
</tr>
</tbody>
</table>
Socket addresses

**IP Header**

IP address

200.23.56.8

**UDP Header**

Port number

69

Socket address

200.23.56.8

69
### 11.2 UDP Packets or *User Datagram*

**Max UDP Data length**

\[
= 65535 - \text{UDP Header (8)} - \text{Smallest IP header (20)} \\
= 65,507
\]
11.3 Checksum

• Checksum covers three sections:
  1. Pseudoheader *(which does not physically exist as part of the user datagram)*
  2. UDP header
  3. Data coming from Application Layer whose length must be an even number of Bytes. Padding may be added to help compute the checksum, then eliminated be.
## Pseudoheader added to the UDP datagram

<table>
<thead>
<tr>
<th>Pseudoheader</th>
<th>32-bit source IP address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>32-bit destination IP address</td>
</tr>
<tr>
<td>All 0s</td>
<td>8-bit protocol (17)</td>
</tr>
<tr>
<td></td>
<td>16-bit UDP total length</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Header</th>
<th>Source port address 16 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination port address 16 bits</td>
</tr>
<tr>
<td></td>
<td>UDP total length 16 bits</td>
</tr>
<tr>
<td></td>
<td>Checksum 16 bits</td>
</tr>
</tbody>
</table>

**Data**

*Padding must be added to make the data a multiple of 16 bits*
Checksum calculation of a simple UDP user datagram

<table>
<thead>
<tr>
<th>153.18.8.105</th>
<th>171.2.14.10</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>All 0s</strong></td>
<td><strong>17</strong></td>
</tr>
<tr>
<td>1087</td>
<td>13</td>
</tr>
<tr>
<td>15</td>
<td><strong>All 0s</strong></td>
</tr>
<tr>
<td><strong>T</strong></td>
<td><strong>E</strong></td>
</tr>
<tr>
<td><strong>I</strong></td>
<td><strong>N</strong></td>
</tr>
</tbody>
</table>

10011001 00010010 → 153.18
00001000 01101001 → 8.105
10101011 00000010 → 171.2
00001110 00001010 → 14.10
00000000 00010001 → 0 and 17
00000000 00001111 → 15
00000100 00111111 → 1087
00000000 00001101 → 13
00000000 00001111 → 15
00000000 00000000 → 0 (checksum)
01010100 01000101 → T and E
01010011 01010100 → S and T
01001001 01001110 → I and N
01000111 00000000 → G and 0 (padding)

10010110 11101011 → Sum
01101001 00010100 → Checksum
11.4 UDP Operation

- Its Connectionless:
  - Independent user datagrams, even if from same source socket to same destination socket. No streaming is supported.
  - Data must be small enough to fit in one user datagram.

- No flow or error control
  - If checksum fail, destination silently drops the datagram.
  - Source gets no idea whether the user datagram has arrived.
Encapsulation and decapsulation

- Pair of sockets
- Data length

Sender/Receiver IP

Protocol = 17

UDP header
UDP data
IP header
IP data
Frame header
Frame data

Message from process → Process → Message to process

a. Encapsulation
b. Decapsulation
Queues in UDP

Daytime client

Outgoing queue

Incoming queue

Port 52000

UDP

Daytime server

Outgoing queue

Incoming queue

Port 13

UDP
Multiplexing and demultiplexing

Processes

UDP (Multiplexer)

IP

UDP (Demultiplexer)

Processes


© Adapted for use at JMU by Mohamed Aboutabl, 2003
11.5 Use of UDP

- Simple request-response communication (no bulk data)
- Processes with Internal flow and error-control mechanisms (e.g. Trivial FTP).
- Multicasting and Broadcasting
- Management processes (e.g. SNMP)
- Routing Information Protocol (RIP) for route updating.
11.6 UDP Package
**Examples**

*Control-block table at the beginning*

<table>
<thead>
<tr>
<th>State</th>
<th>Process ID</th>
<th>Port Number</th>
<th>Queue Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN-USE</td>
<td>2,345</td>
<td>52,010</td>
<td>34</td>
</tr>
<tr>
<td>IN-USE</td>
<td>3,422</td>
<td>52,011</td>
<td></td>
</tr>
<tr>
<td>FREE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IN-USE</td>
<td>4,652</td>
<td>52,012</td>
<td>38</td>
</tr>
<tr>
<td>FREE</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example 1

The first activity is the arrival of a user.
Example 2

After a few seconds, a process starts. It asks
## Modified table after Example 2

<table>
<thead>
<tr>
<th>State</th>
<th>Process ID</th>
<th>Port Number</th>
<th>Queue Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN-USE</td>
<td>2,345</td>
<td>52,010</td>
<td>34</td>
</tr>
<tr>
<td>IN-USE</td>
<td>3,422</td>
<td>52,011</td>
<td></td>
</tr>
<tr>
<td>IN-USE</td>
<td>4,978</td>
<td>52,014</td>
<td></td>
</tr>
<tr>
<td>IN-USE</td>
<td>4,652</td>
<td>52,012</td>
<td>38</td>
</tr>
<tr>
<td>FREE</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example 3

A user datagram now arrives for port 52,011.
## Modified table after Example 3

<table>
<thead>
<tr>
<th>State</th>
<th>Process ID</th>
<th>Port Number</th>
<th>Queue Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN-USE</td>
<td>2,345</td>
<td>52,010</td>
<td>34</td>
</tr>
<tr>
<td>IN-USE</td>
<td>3,422</td>
<td>52,011</td>
<td>43</td>
</tr>
<tr>
<td>IN-USE</td>
<td>4,978</td>
<td>52,014</td>
<td></td>
</tr>
<tr>
<td>IN-USE</td>
<td>4,652</td>
<td>52,012</td>
<td>38</td>
</tr>
<tr>
<td>FREE</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example 4

After a few seconds, a user datagram arrives for port 52,222. The input module checks the table and cannot find the entry for this destination. The user datagram is dropped and a request is made to ICMP to send an “unreachable port” message to the source.
Example 5

After a few seconds, a process needs to send a user datagram. It delivers the data to the output module which adds the UDP header and sends it.