IP Packet Header and Fragmentation

CME451 Tutorial 4

Hao Zhang
(Graduate Teaching Fellow)

Department of Electrical & Computer Engineering
University of Saskatchewan

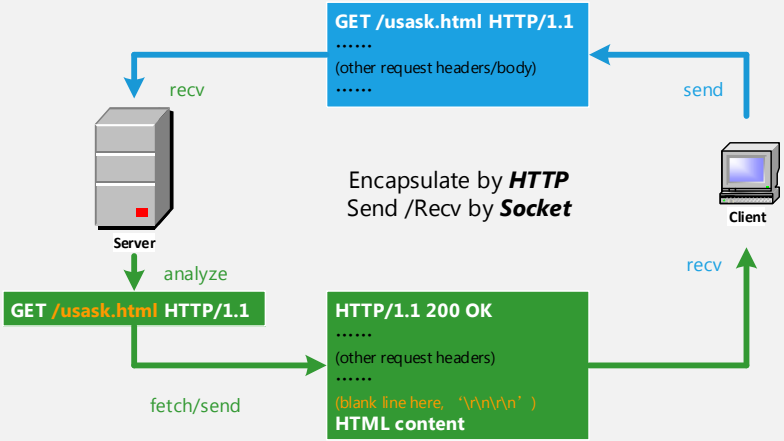
Jan 27, 2017

Review

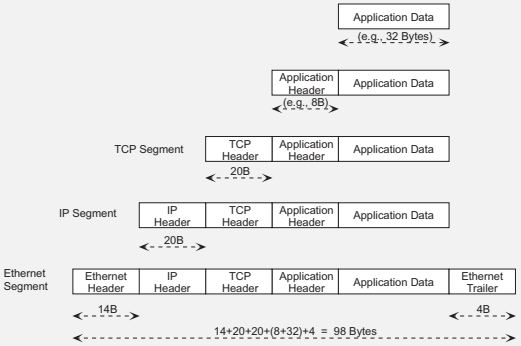
- Application-Layer Programming:
  - Open and download webpage.
  - Analyze HTML file content.
  - Fetch resources from server – *Find the correct resource path.*
- Socket Programming:
  - HTTP, TCP ,IP protocols.
  - Socket concepts.
  - Simple socket server and client.
  - Web socket server and client – *HTTP request/response header.*

Review

- Web Server and Client



Protocol Headers in Each Layer



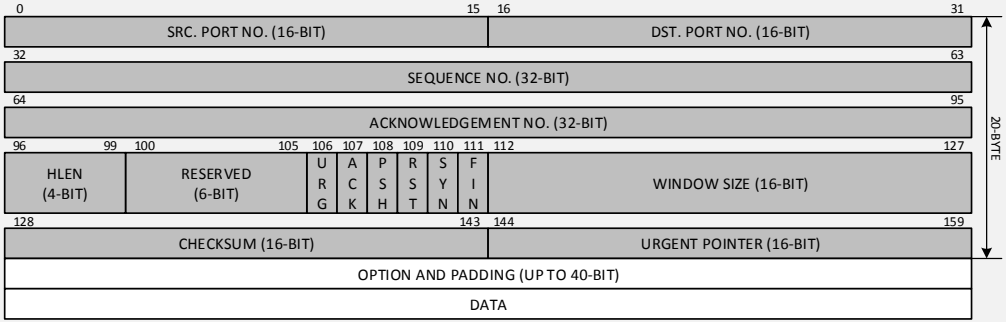
(**Network Infrastructure and Architecture: Designing High-Availability Networks*, 1st Ed., Wiley 2008, pp.142)

- Application Layer:
  - HTTP request/response header
- Transport Layer:
  - TCP header: port number, Seq number, ACK number, ...
- Network Layer:
  - IP header: IP address, ...
- Data Link Layer:
  - Ethernet header: MAC address, CRC checksum ...

TCP Header

Overall Format

- TCP Header Format:



TCP Header

Each Component

- SRC. PORT NO.: 16-bit
  - The source port number.
  - Support Port: 0 - 65535
- DST. PORT NO.: 16-bit
  - The destination port number.
  - Support Port: 0 - 65535
- SEQUENCE NO.: 32-bit
  - The sequence number of the first data octet in this segment.
  - If SYN is present, the sequence number is the initial sequence number (ISN) and the first data octet is ISN + 1.
- ACKNOWLEDGEMENT NO.: 32-bit
  - If the ACK bit is set, this field contains the value of the next sequence number the sender of the segment is expecting to receive.

TCP Header

Each Component

- `HLEN`: 4-bit
  - The number of 32-bit words (4 Bytes) in the TCP header.
  - Indicate the where data begins.
  - TCP header is always an integral number of 32-bit long.
- `RESERVED`: 6-bit
  - Reserved for future use. Must be all zeros.
- `Control Bits`: 6-bit
  - URG: Urgent pointer field significant.
  - ACK: Acknowledgement field significant.
  - PSH: Push function.
  - RST: Reset the connection.
  - SYN: Synchronize sequence numbers.
  - FIN: No more data from sender.

TCP Header

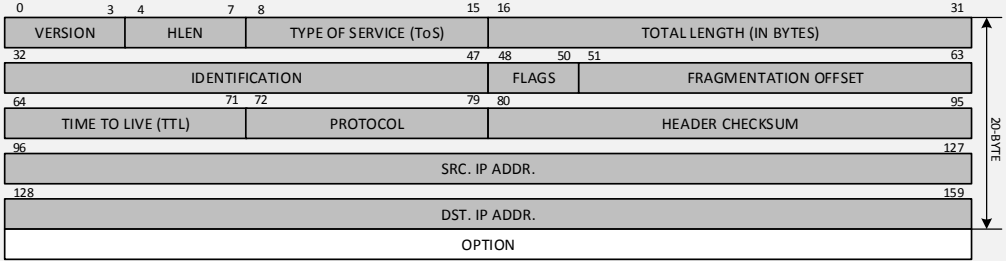
Each Component

- `WINDOW SIZE`: 16-bit
  - The number of data octets beginning with the one indicated in the `ACKNOWLEDGEMENT NO.` field which the sender of this segment is willing to accept.
- `CHECKSUM`: 16-bit
  - The checksum field is the 16 bit one's complement of the one's complement sum of all 16 bit words in the header and text.
- `URGENT POINTER`: 16-bit
  - Communicates the current value of the urgent pointer as a positive offset from the sequence number in this segment.
  - The urgent pointer points to the sequence number of the octet following the urgent data.
  - Only be interpreted in segments with the `URG` control bit set.
- `OPTION AND PADDING`:
  - Options can be added at the end of the TCP header.
  - Padding is used to ensure the TCP header ends and data begins on a 32 bit boundary. Padding is composed of zeros.

IP Packet Header

Overall Format

- IP Header Format:



IP Packet Header

Each Component

- `VERSION`: 4-bit
  - The version of IP protocol.
  - Currently 4 (`0010` in binary) for IPv4. 6 (`0110` in binary) for IPv6.
- `HLEN`: 4-bit
  - Total length of the datagram header in 4-byte words.
  - Default value is 5 indicating header length is 20-byte.

IP Packet Header

Each Component

- `TYPE OF SERVICE (TOS)`: 8-bit
  - Service type: 3-bit precedence, 4-bit ToS bits, last bit not used.
  - Differentiated services: 6-bit codepoint, last two bits not used.

- Service Type:
  - Precedence: defines the priority of the datagram in case of congestion. Range from 000 to 111 in binary. Never used!
  - ToS bits: One and only one of the 4-bit can have a value of 1.

ToS Bits		Description
0000	all zeros	normal (default)
0001	C = 1	minimize cost
0010	R = 1	maximize reliability
0100	T = 1	maximize throughput
1000	D = 1	minimize delay

IP Packet Header

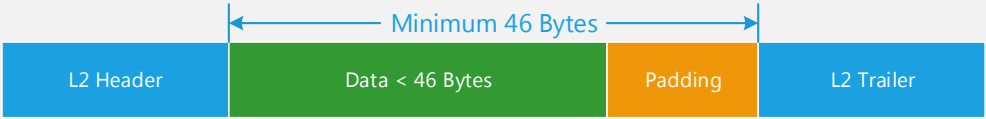
Each Component

Protocol	ToS Bits	Description
ICMP	0000	Normal
BOOTP	0000	Normal
NNTP	0001	Minimize cost
IGP	0010	Maximize reliability
SNMP	0010	Maximize reliability
TELNET	1000	Minimize delay
FTP (data)	0100	Maximize throughput
FTP (control)	1000	Minimize delay
TFTP	1000	Minimize delay
SMTP (command)	1000	Minimize delay
SMTP (data)	0100	Maximize throughput
DNS (UDP query)	1000	Minimize delay
DNS (TCP query)	0000	Normal
DNS (zone)	0100	Maximize throughput

- Differentiated Services:
  - When 3 rightmost bits are all 0, the 3 leftmost bits are precedence bits.
  - When 3 rightmost bits are not all 0, the 6-bit codepoint defines 64 services.
    - Category 1: 32 services, assigned by IETF.
    - Category 2: 16 services, local authorities.
    - Catelogy 3: 16 services, temporary or experimental use.

Category	Codepoint	Assigning Authority
1	xxxxx0	Internet
2	xxxx11	Local
3	xxxx01	Temporary or Experimental

- `TOTAL LENGTH`: 16-bit
  - Define the packet total length (header + data) in bytes.
  - Length of Data = Total Length - Header Length
  - Total length of IPv4 datagram is limited to 65535 ($2^{16} - 1$) bytes.
  - Total length is necessary when datagram smaller than 46 bytes.
    - Ethernet protocol requires datagram between 46 to 1500 bytes.



- `TIME TO LIVE (TTL)`: 8-bit
  - The maximum number of hops (routers) visited by the datagram.
  - Avoid infinite looping in the network.
  - At each hop, this value is decremented by one.
  - The packet is discarded when its value reaches zero.
- `PROTOCOL`: 8-bit
  - Defines the higher level protocol.

Value	Protocol
1	ICMP
2	IGMP
6	TCP
17	UDP
89	OSPF

IP Packet Header

Each Component

- `HEADER CHECKSUM`: 16-bit
  - The entire header is divided into 16-bit sections and added together.
  - The checksum is obtained by taking 1's complement of the sum.
  - In receiver side, when the result of the calculated sum plus the checksum is all zeros, the data is error free.
- `SRC. IP ADDR.`: 32-bit
  - 32-bit IPv4 address of the source.
  - Must be unchanged during the transmission.
- `DST. IP ADDR.`: 32-bit
  - 32-bit IPv4 address of the destination.
  - Must be unchanged during the transmission.

IP Fragmentation

- Maximum Transfer Unit (MTU)
  - Defined in data link layer protocol.
  - Defines the maximum size of the data field.
  - When encapsulation, the total size of datagram must be less than MTU.
  - **Fragmentation**: divide the datagram to meet MTU.

Protocol	**MTU**
Hyperchannel	65,535
Token Ring (16 Mbps)	17,914
Token Ring (4 Mbps)	4,464
FDDI	4,352
Ethernet	1,500
X.25	576
PPP	296

IP Fragmentation

- When a datagram is fragmented, each fragment has its own header with most of the fields repeated.
- Three fields must be changed:
  - `FLAGS`
  - `FRAGMENTATION OFFSET`
  - `TOTAL LENGTH`
- Other fields must be copied. The checksum must be recalculated.

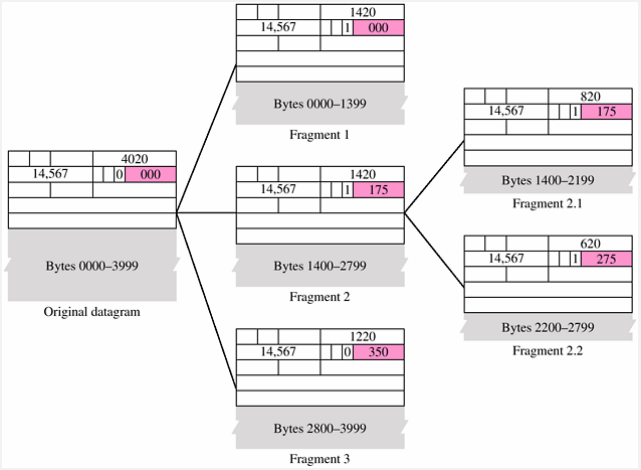
IP Fragmentation

Related IP Header Component

- `IDENTIFICATION`: 16-bit
  - Identifies a datagram originating from the source host.
- `FLAGS`: 3-bit
  - First bit is reserved.
  - Second bit is `Do not fragment` bit.
    - Datagram cannot be fragmented if `Do not fragment = 1`.
    - Datagram is discarded if cannot pass the physical network.
  - Third bit is `More fragment` bit.
    - `More fragment = 1`: more fragments after this one.
    - `More fragment = 0`: the last or the only fragment.
- `FRAGMENTATION OFFSET`: 13-bit
  - Relative position of this fragment with respect to the whole datagram.
  - Measured in units of 8 bytes.

IP Fragmentation

Example

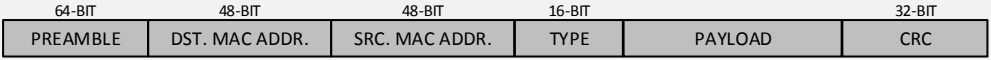


(**Data Communications and Networking*, 4th Ed., Chapter 20.)

IP Defragment

- Refer to three components:
  - `IDENTIFICATION`
  - `FRAGMENTATION OFFSET`
  - `FLAG – More fragment`
- The fragments with same `IDENTIFICATION` can be reassembled.
- The first fragment has `FRAGMENTATION OFFSET = 0`
- Second fragment offset: Divide data length of first fragment by 8. Find this fragment.
- Third fragment offset: Divide data length of first and second fragment by 8.
- The last fragment has a `More fragment = 0`.

Ethernet Frame



- Data link layer format.
- Payload comes from Network Layer.
- Preamble will be added in Physical Layer.
- `TYPE`:

TYPE	Upper Level Protocol
0x0800	IPv4
0x8100	802.1Q tagged frame
0x0806	ARP
0x86DD	IPv6

Appendix I

Wireshark

- Wireshark is an open-source network packet analyzer.
- Capture packets passing through a network interface.
- Display the contents of various protocol fields.
- Refer to lab manual for more detail.

Appendix II

Python Number System Conversion

```
>>> hex_data = "01F6"
>>> bin_data = (bin(int(hex_data, 16))[2:]).zfill(len(hex_data)*4)
>>> bin_data
'0000000111110110'
```

- `int(x, base)` return an integer constructed from number or string `x`.
- `bin(x)` convert an integer number to binary string. Return format `'0bxxxxx'`, actual number starting at index 2.
- `zfill(length)` Pad a numeric string on the left with zeros to fill `length`.

Appendix III

Python Dictionary

- Dictionary stores (key, value) pairs.
- Unordered.
- Can contain multiple data types.
- Keys must be unique in one dictionary.

Appendix III

Python Dictionary

```
>>> dic = {}
>>> dic['lab1'] = 'basics'
>>> dic['lab2'] = 2
>>> dic['lab3'] = 'IP'
>>> dic
{'lab1':'basics', 'lab3':'IP', 'lab2':2}
>>> dic.keys()
dict_keys(['lab1', 'lab3', 'lab2'])
>>> sorted(dic.keys())
dict_keys(['lab1', 'lab2', 'lab3'])
>>> del dic['lab2']
>>> dic['lab1']
'basics'
>>> dic.keys()
dict_keys(['lab1', 'lab3'])
>>> 'lab3' in dic
True
```

Appendix IV

HTML Table

```
<!DOCTYPE html>
<html>
<head>
<style>
table, th, td {
    border: 1px solid black;
    border-collapse: collapse;
}
th, td {
    padding: 5px;
    text-align: center;
}
</style>
</head>
```

Appendix IV

HTML Table

```
<!-- continue -->
<body>

<h2>Cell that spans two columns:</h2>
<table style="width:100%">
  <tr>
    <th>Name</th>
    <th colspan="2">Telephone</th>
  </tr>
  <tr>
    <td>Bill Gates</td>
    <td>55577854</td>
    <td>55577855</td>
  </tr>
</table>

</body>
</html>
```