# Introduction to TCP/IP

punkis@attrition.org

# Intro to TCP/IP

## What this lecture is

- This lecture will give a general low-level overview of TCP/IP
- This lecture is geared toward novices
- If you want a high level overview of TCP/IP go buy TCP/IP Illustrated

# Intro to TCP/IP

## History - The Beginning

- 1969 - **A**dvanced **R**esearch **P**roject **A**gency (ARPA) funds research and development of an experimental packet-switching network (ARPANET)
- ARPA's goal was to study techniques for providing robust, vendor independent data communications

# Intro to TCP/IP

## History

- ARPANET was so successful that many organizations attached to it began to use it on a daily basis
- 1975 - ARPANET converted from an experimental network to an operational network when the **D**efense **C**ommunications **A**gency (DCA) took control of it

# Intro to TCP/IP

## History - 1983

- TCP/IP protocols developed as Military Standards. All hosts on the network were required to convert to the new protocols
- DARPA funded the implementation of TCP/IP in Berkley (BSD) Unix
- The term *internet* came into common use

# Intro to TCP/IP

## History

- 1983 - ARPANET is split into MILNET and a new smaller ARPANET
- 1985 - The **N**ational **S**cience **F**oundation (NSF) creates NSFNet and connects it to the internet
- 1987 - NSF creates a new faster backbone and a three-tiered topology that includes backbone, regional networks, and local networks

# Intro to TCP/IP

## History

- 1990 - ARPANET passes out of existence.
- 1995 - NSFNet ceases its role as primary backbone for the internet

# Intro to TCP/IP

## History - In a Nutshell

- What has come to be known as the internet was originally an experiment used primarily by researchers and developers
- The internet has grown much larger than it was originally designed for
- The original networks and agencies involved in the creation of the internet no longer play an essential role

# Intro to TCP/IP

## Myths

- Contrary to what he may think, Al Gore did NOT invent the internet.
- Mr. Gore was 21 years old when ARPA funded research and development of ARPANET

# TCP/IP
## Defined

- **T**ransmission **C**ontrol **P**rotocol/**I**nternet **P**rotocol:

  The suite of networking protocols that have been used to construct the global Internet. Also referred to as the DoD or ARPANET protocol suite because their early development was funded by the Advanced Research Projects Agency (ARPA) of the US Department of Defense (DoD).

# TCP/IP

## In a Nutshell:

- The series of protocols that allow computers to communicate with one another regardless of Operating System or vendor

# The 4 layers of TCP/IP

| | |
|---|---|
| **Application** | **Telnet, FTP, mail, etc** |
| **Transport** | **TCP, UDP** |
| **Network** | **IP, ICMP, IGMP** |
| **Link** | **Device Driver and Interface Card** |

# The 4 layers of TCP/IP

- *Link layer* (Data link layer)

  This layer includes the device driver in the OS and the corresponding network interface card in the computer. Handles the hardware details of physically interfacing with the network.

# The 4 layers of TCP/IP

- *Network Layer* (Internet Layer)

  Handles the movement and routing of packets around the network

# The 4 layers of TCP/IP

- *Transport Layer*

  Provides a flow of data between two hosts for the *Application Layer* above. Two different Transport protocols are used at this level:

  **TCP: Reliable. Breaks data passed from the *Application Layer* above into chunks for the *Network Layer* below, acknowledging received packets setting timeouts, etc.**

  **UDP: Unreliable. Sends packets of data (Datagrams) from one host to another with no guarantee packets will reach their destination.**

# The 4 layers of TCP/IP

- *Application Layer*

  This layer handles the details of the particular application being used. Some standard TCP/IP applications include:

  - Telnet
  - FTP
  - SMTP
  - SNMP

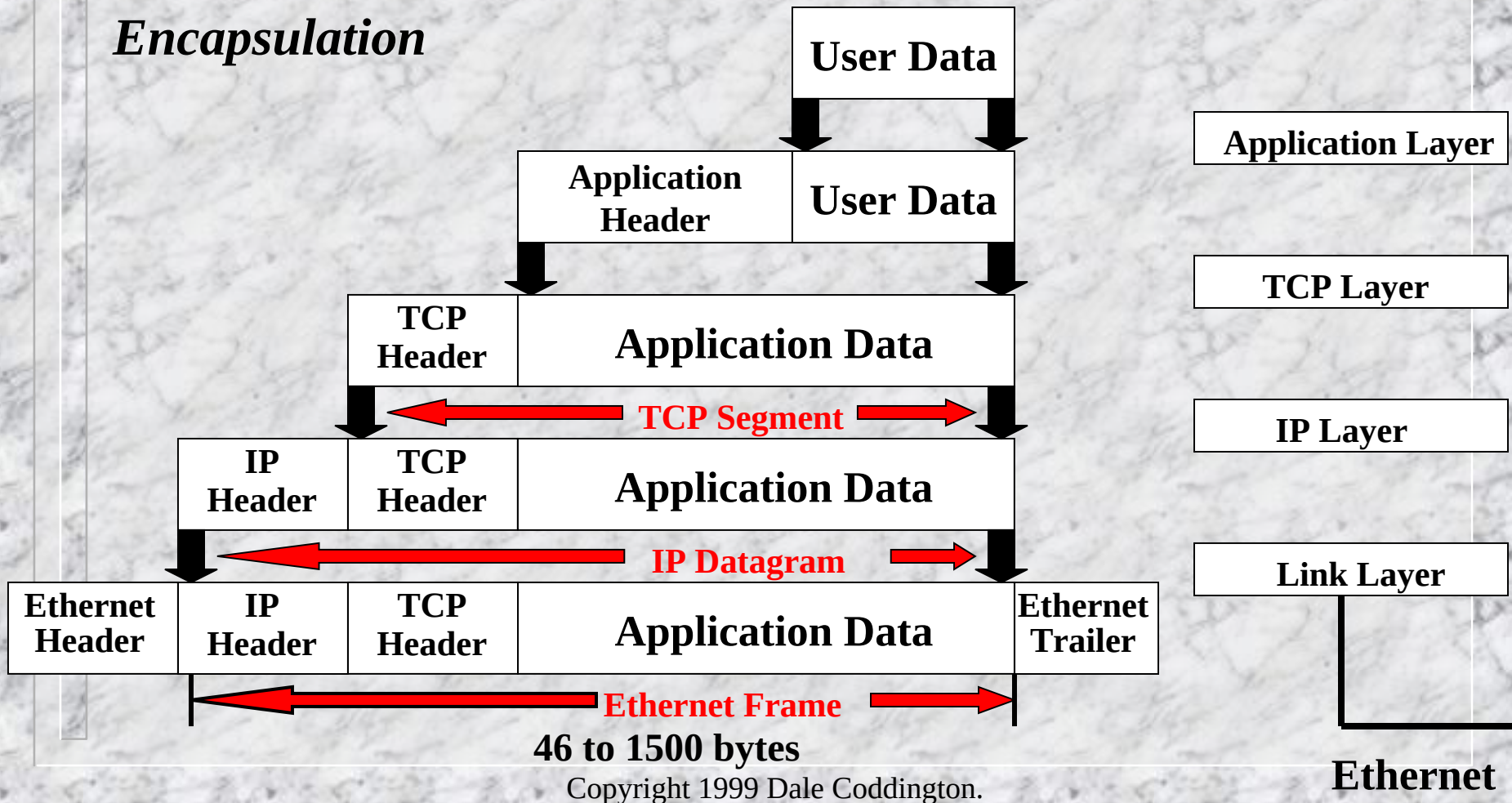
# The 4 layers of TCP/IP

- *Encapsulation*

  When an an application sends data using TCP, it is sent through each layer in the protocol stack.

  - Each layer adds information to the data by adding a header and sometimes a footer.
  - The data is then sent as a stream of bits across the network

# The 4 layers of TCP/IP



# The 4 layers of TCP/IP

- *Demultiplexing*
  - When an ethernet frame is received by a host it starts its way back up the protocol stack
  - Each layer looks at its respective header and decides what to do with the data before passing it up to the next layer

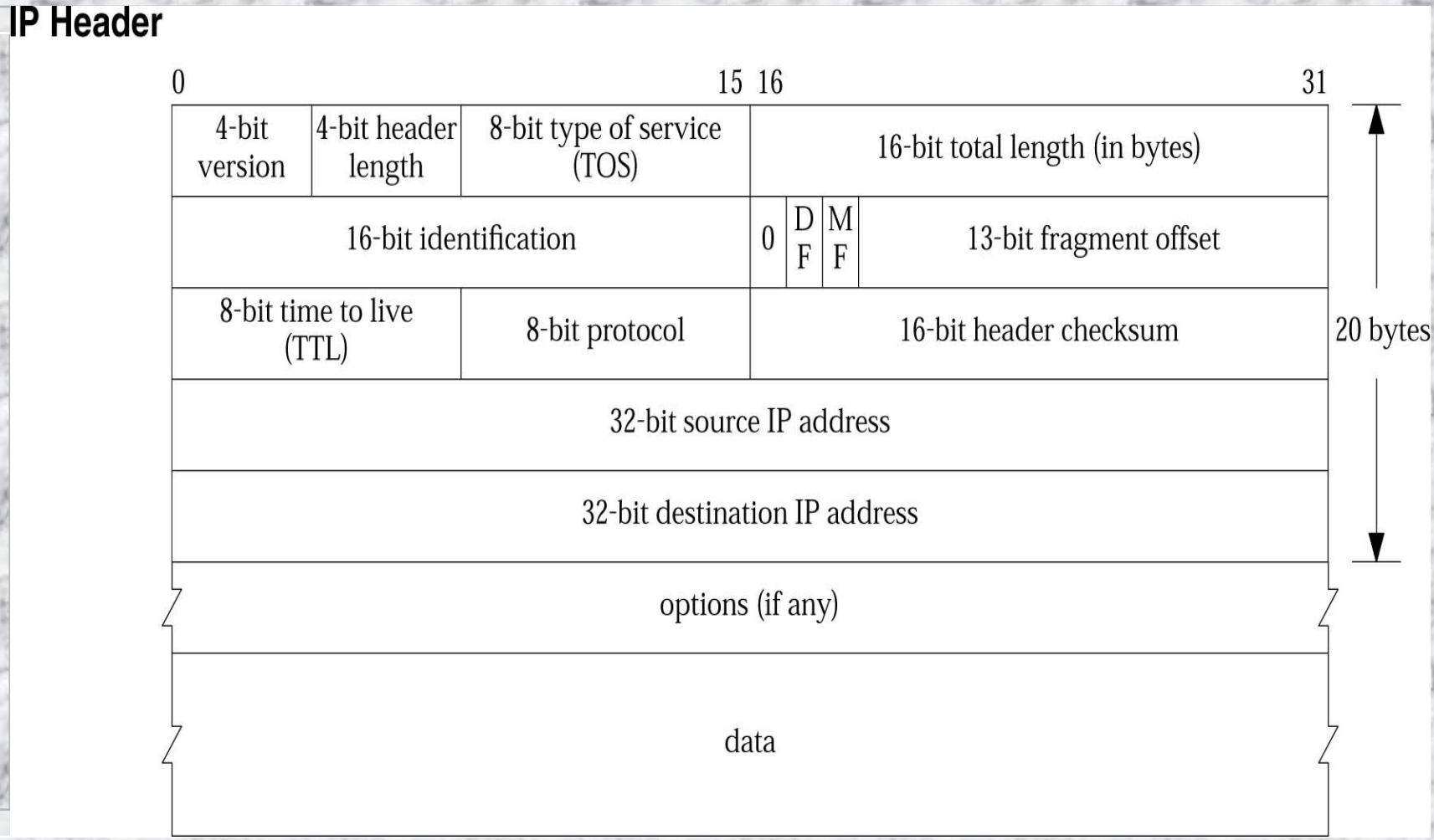
# TCP/IP Networking Protocols

# Internet Protocol

## IP - Features

- The dominant network layer protocol used by the TCP/IP suite of protocols
- IP defines the rules for packaging network traffic into IP datagrams and also defines the rules for moving these datagrams across a network
- IP is also responsible for fragmenting data wherever necessary and to properly reassemble the datagrams at the other end

# Internet Protocol



# Internet Protocol

## IP Datagram Fields

- Version: Indicates which version of IP is being used (typically 4)
- Header Length: Indicates how many 4-byte words are in the header
- Type of Service (**TOS**): Indicates the level of service the IP datagram should be assigned
- Datagram Length: The length of the entire datagram including the header (max size 65,535 bytes)
- Datagram Identification: Uniquely identifies each datagram sent by a host

# Internet Protocol

## IP Datagram Fields

- Flags: The first of three flags are unused. The Don't Fragment (**DF**) and More Fragment (**MF**) flags control the way a datagram is fragmented
- Fragment Offset: Indicates how many units from the start of the original datagram the current datagram is
- Time to Live (**TTL**): Indicates how many routers a datagram may traverse before being dropped (max TTL is 255)
- Protocol: Identifies which protocol handed the IP to data
- Header Checksum: A check on the IP header only to ensure the header is not corrupted

# Internet Protocol

## IP Datagram Fields

- Source/Destination IP Addresses: 32 bit IP addresses of originating host and destination host
- Options: Currently defined options are security and handling restrictions, record route, timestamp, loose source routing, and strict source routing. These options are rarely used
- Data

# Internet Protocol

## Packet Analysis

```
IP Header
  Version:                    4
  Header Length:              20 bytes
  Service Type:               0x00
  Datagram Length:            40 bytes
  Identification:             0x5850
  Flags:                      MF=off, DF=on
  Fragment Offset:            0
  TTL:                        32
  Encapsulated Protocol:      TCP
  Header Checksum:            0x9658
  Source IP Address:          172.16.10.2 (broken)
  Destination IP Address:     172.16.10.5 (testbed)
```

# Traceroute

- There is no guarantee that two connective IP datagrams from the same source to the same destination will take the same route but they usually do
- Traceroute is a tool that help to trace the flow of IP datagrams from one host to another

# Traceroute - How it works

- Sends an IP datagram with a TTL of 1 to the destination host
- The first router to handle the datagram decrements the TTL to 0, throws away the datagram, and sends an ICMP time exceeded back to the host
- Traceroute then sends another datagram with a TTL of 2 and we find the IP address of the second router
- This continues until the datagram reaches the host

# Traceroute

- Sample traceroute output:

```
# traceroute victim.com
traceroute to victim (172.16.1.10), 30 hops max, 40 byte
   packets
1 satan  (172.16.1.66) 20ms 10ms 10ms
2 victim (172.16.1.10) 120ms 120ms 120ms
```

- For each TTL 3 datagrams are sent. These values are recorded in the output.

# Transmission Control Protocol

## TCP - Features

- TCP is a transport layer protocol
- Provides a way to connect hosts across an network reliably
- Provides a ‘virtual circuit’ between two hosts
- Communicating hosts are required to acknowledge receipt of network traffic

# Transmission Control Protocol

## TCP - Features

- TCP packages its data into segments which contain both data and session control information
- Since segments traversing a network may arrive out of order TCP provides proper reassembly of these segments
- Sequence numbers are used to properly reassemble segments

# Sequence Numbers

## Packet Analysis

```
Packet 49
 TCP:   port ftp-data -> 26410 seq=1326731397 ack=1518678629
DATA:   1460 bytes
-------------------------------------------------------------------------
Packet 50
 TCP:   port ftp-data -> 26410 seq=1326732857 ack=1518678629
DATA:   1460 bytes
-------------------------------------------------------------------------
Packet 51
 TCP:   port ftp-data -> 26410 seq=1326734317 ack=1518678629
DATA:   1460 bytes
-------------------------------------------------------------------------
Packet 52
 TCP:   port ftp-data -> 26410 seq=1326735777 ack=1518678629
DATA:   1460 bytes
```

ftp transfer tcpdump output

# Transmission Control Protocol

## TCP - Features

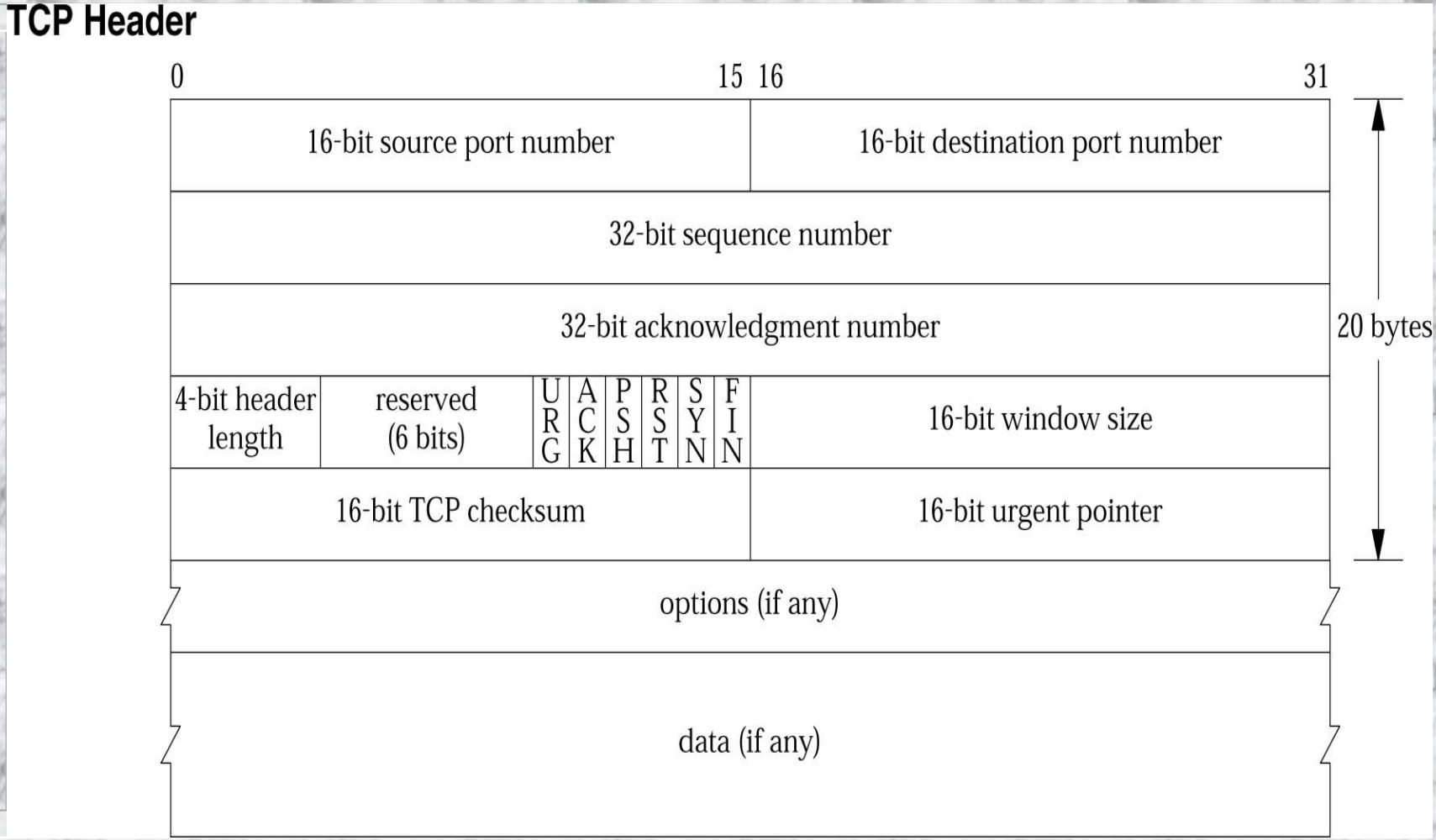
- Maximizes performance of a connection by ensuring TCP segments are neither too large or too small

# Transmission Control Protocol

## TCP - In a Nutshell

- Virtual circuits - TCP connections behave like a live two-way connection
- Reliable connections - TCP segments are guaranteed to reach their destination, if they are not the user is notified
- Performance optimization - TCP can modify transmission variables depending on network conditions

# Transmission Control Protocol



# Transmission Control Protocol

## TCP Header Fields

- Source Port/Destination Port: 16-bit port number or originating host and destination host
- Sequence Number: 4-byte number assigned by TCP starting with an randomly chosen number. This number is used to determine how many bytes have been transmitted across the network
- Acknowledgement Number: Acknowledges the last segment sent by the host
- Header Length: Measures the header length in 4-byte words

# Transmission Control Protocol

## TCP Header Fields

- Flags: used when negotiating and managing a connection:

**URG**: Indicates segment being sent is urgent

**ACK**: Indicates ack number in segment header is valid

**PSH**: Pass the data to the application as soon as possible

**RST**: Resets the connection

**SYN**: Synchronize sequence numbers to initiate a connection

**FIN**: The sender is finished sending data

- 16-Bit Window Size: The number of bytes the receiving host is willing to accept

# Transmission Control Protocol

## TCP Header Fields

- 16-Bit TCP Checksum: A checksum of the TCP Header and data
- 16-Bit Urgent Pointer: used only if the **URG** flag is set
- Options: The most commonly used option is the Maximum Segment Size (MSS) option. Determines the maximum sized segment the sender is willing to receive
- Data: This portion of the TCP segment is optional. When connections are being established or terminated no data is sent

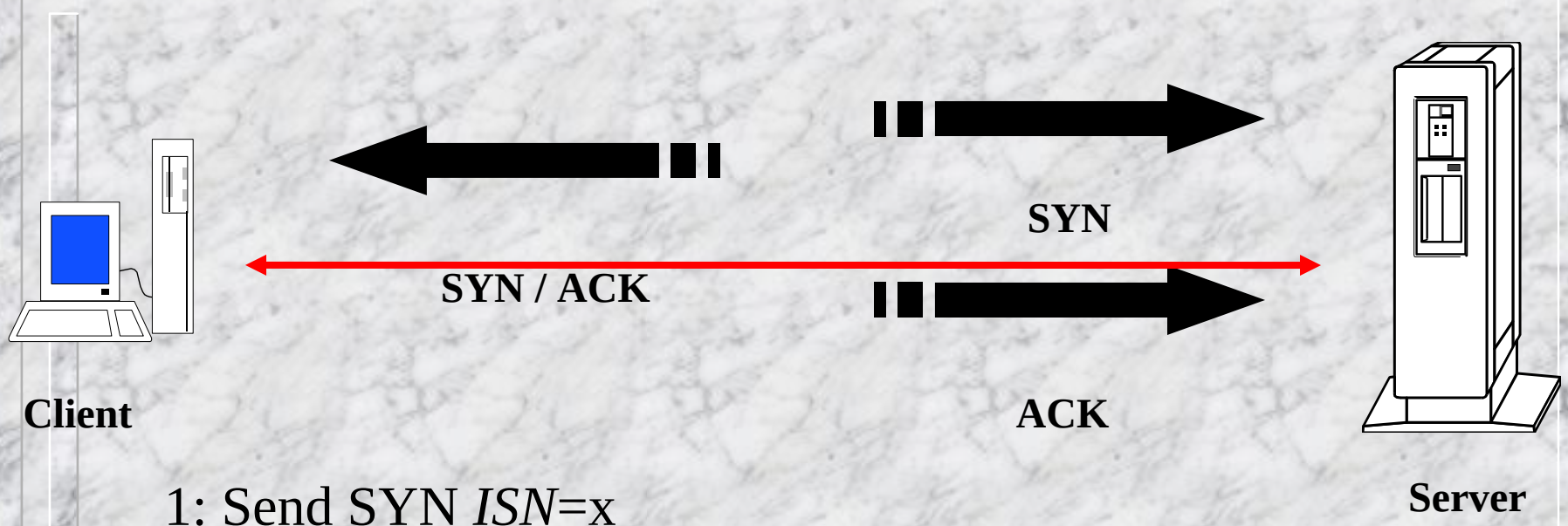
# Transmission Control Protocol

## Packet Analysis

```
TCP Header
 Source Port:                    22 (ssh)
 Destination Port:               1714 (<unknown>)
 Sequence Number:                1937534412
 Acknowledgement Number:         0104479939
 Header Length:                  20 bytes (data=0)
 Flags:                          URG=off, ACK=on,  PSH=off
                                 RST=off, SYN=off, FIN=off
 Window Advertisement:           32736 bytes
 Checksum:                       0xD102
 Urgent Pointer:                 0
```

# TCP - Establishing a Connection

## Three Way Handshake




1: Send SYN *ISN*=x

2: Send SYN *ISN*=y, ACK x+1

3: Send ACK *ISN* y+1

4: Full connection established

# TCP Three Way Handshake

## Packet Analysis - Part 1

```
-------------------------------------------------------------------------
Packet 1
TIME:   19:50:32.912582 (0.040960)
LINK:   00:40:05:E3:09:D0 -> 00:00:C5:38:0D:27 type=IP
IP:   strife -> testbed hlen=20 TOS=00 dgramlen=44 id=2864
      MF/DF=0/0 frag=0 TTL=64 proto=TCP cksum=E641
TCP:   port 24616 -> ftp seq=2735221453 ack=0000000000
      hlen=24 (data=0) UAPRSF=000010 wnd=512 cksum=FBEC urg=0
DATA:   <No data>
-------------------------------------------------------------------------
```

The requesting client sends a *SYN* (synchronize) segment specifying the **port number** of the server it wishes to connect to and the client's ***ISN*** (Initial Sequence Number).

# TCP Three Way Handshake

## Packet Analysis - Part 2

```
-------------------------------------------------------------------------
Packet 2
TIME:   19:50:32.912792 (0.000210)
LINK:   00:00:C5:38:0D:27 -> 00:40:05:E3:09:D0 type=IP
  IP:   testbed -> strife hlen=20 TOS=00 dgramlen=44 id=5FF4
        MF/DF=0/1 frag=0 TTL=64 proto=TCP cksum=6EB1
 TCP:   port ftp -> 24616 seq=2809565737 ack=2735221454
        hlen=24 (data=0) UAPRSF=010010 wnd=17520 cksum=7FCB urg=0
DATA:   <No data>
-------------------------------------------------------------------------
```

The server responds with a *SYN* segment including the servers own ***ISN***. An ***ACK*** (acknowledge) is also sent with the clients ISN plus one.

# TCP Three Way Handshake

## Packet Analysis - Part 3

```
-------------------------------------------------------------------------
Packet 3
TIME:   19:50:32.913768 (0.000976)
LINK:   00:40:05:E3:09:D0 -> 00:00:C5:38:0D:27 type=IP
  IP:   strife -> testbed hlen=20 TOS=00 dgramlen=40 id=2865
        MF/DF=0/1 frag=0 TTL=64 proto=TCP cksum=A644
 TCP:   port 24616 -> ftp seq=2735221454 ack=2809565738
        hlen=20 (data=0) UAPRSF=010000 wnd=32120 cksum=5E80 urg=0
DATA:   <No data>
-------------------------------------------------------------------------
```

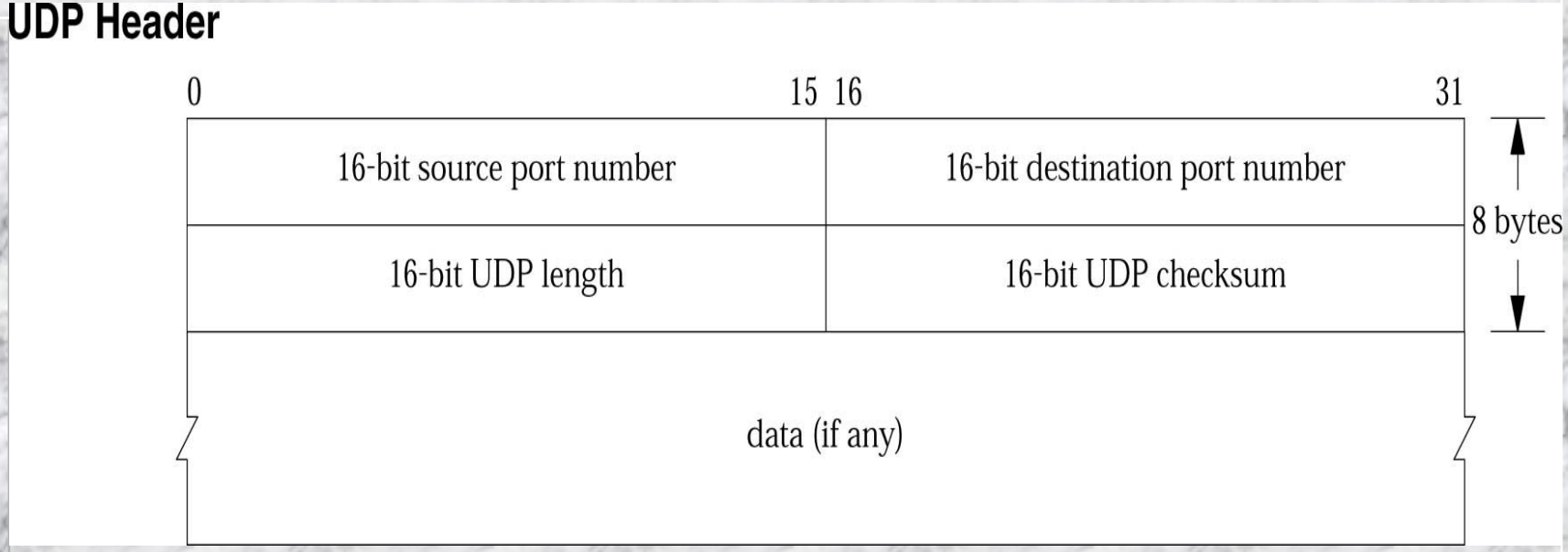
The client acknowledges the servers *SYN* and sends an ***ACK*** segment with the Servers *ISN* plus one.

# User Datagram Protocol

## UDP - Features

- UDP is a transport layer protocol
- Does not use the benefit of error detection, error correction, handshaking, or verification of delivery like TCP
- Provides a connectionless delivery system between two hosts
- UDP has low overhead

# User Datagram Protocol



# User Datagram Protocol

## UDP Datagram Format

- Source Port Number: 16-bit port number of originating host. Uses ephemeral ports (greater than 1024)
- Destination Port Number: 16-bit port number of destination host. Uses well-known port numbers
- 16-Bit UDP Length: Indicates the length of the length of entire UDP datagram, including header
- 16-Bit UDP Checksum: A checksum of the entire UDP datagram

# UDP

## Packet Analysis

```
UDP Header
   Source Port:              2167 (<unknown>)
   Destination Port:         53 (domain)
   Datagram Length:          37 bytes (Header=8, Data=29)
   Checksum:                 0xD5B0
```

# Intro to TCP/IP

## References

- Books

• TCP/IP Illustrated Volume 1
W. Richard Stevens
Addison-Wesley
1994
ISBN:0-201-63346-9

• TCP/IP Network Administration - 2nd Edition
Craig Hunt
O'Reilly & Associates
1998
ISBN: 1-56592-322-7

# Intro to TCP/IP

## References

- WWW

• TCP/IP FAQ Frequently Asked Questions (1999-07) Part 1 of 2

**http://www.cis.ohio-state.edu/hypertext/faq/usenet/internet/tcp-ip/tcp-ip-faq/part1/faq.html**

• TCP/IP FAQ Frequently Asked Questions (1999-07) Part 1 of 2

**http://www.cis.ohio-state.edu/hypertext/faq/usenet/internet/tcp-ip/tcp-ip-faq/part1/faq.html**

# Intro to TCP/IP

## References

### Tools

• tcpshow

Tcpshow reads a tcpdump(1) savefile and provides a reasonably complete decode of Ethernet, ARP, RARP, IP, ICMP, UDP and TCP headers, in packets that match the boolean expression. The data belonging to these packets is displayed in ASCII.

ftp://ftp.freebsd.org/pub/FreeBSD/packages/net/tcpshow-1.74.tgz

• tcpdump

Tcpdump prints out the headers of packets on a network interface that match the boolean expression.

ftp://ftp.ee.lbl.gov/tcpdump.tar.Z