

- TCP and UDP -

Transport Layer Protocols

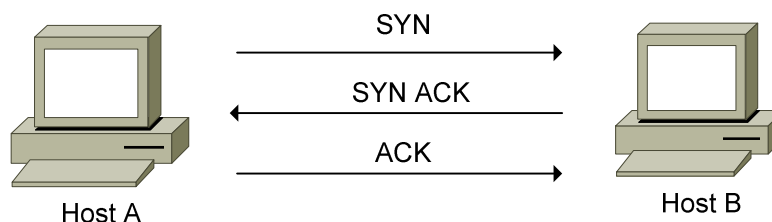
The **Transport** layer of the OSI model (or, the **Host-to-Host** layer of the DoD model) is concerned with the reliable transfer of data between devices. It ensures (or in some cases, does *not* ensure) that a packet arrives at its destination without corruption or data loss.

However, protocols at the transport layer do not actually *send* or *route* packets. Network layer protocols, such as IP, route packets from one network to another. In the TCP/IP protocol suite, TCP and UDP are transport layer protocols.

Transmission Control Protocol (TCP)

The **Transmission Control Protocol (TCP)** is defined as a reliable, **connection-oriented** transport protocol. Parameters must be agreed upon by both parties before a connection is established.

TCP utilizes a **three-way handshake** to accomplish this. Control messages are passed between two devices as the connection is set up:



- Host A sends a **SYN** (short for **synchronize**) message to Host B to initiate a connection
- Host B responds with an **ACK** (short for **acknowledgement**) to Host A's SYN message, and sends its own **SYN** message (both messages are combined to form a SYN+ACK)
- Host A completes the three-way handshake by sending an **ACK** to Host B's SYN.

The TCP header contains both a SYN flag and an ACK flag. Thus, when a particular message needs to be sent, the appropriate flag is marked as on (in other words, changed from a "0" to a "1"). A SYN+ACK message has both flags set to on (1).

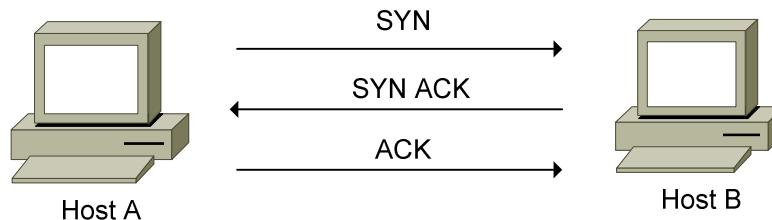
* * *

All original material copyright © 2006 by Aaron Balchunas (aaron@routeralley.com),
unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

Transmission Control Protocol (TCP) (continued)

Additionally, TCP **segments** data into smaller pieces for transport. Segments are assigned a sequence number, so that the receiving device can then reassemble this data in order upon arrival.



- Host A sends an **initial sequence number (ISN)** with its SYN message. This number is chosen from a random timer – we’ll assume an ISN of 4000.
- Host B responds to this sequence number with an acknowledgment number, which is always one more than the sequence number. Thus, Host B’s acknowledgment number is 4001.
- Additionally, Host B sends an initial sequence number with *its* SYN message. We’ll assume Host B’s ISN is 6000.
- Host A responds to this sequence with an acknowledgment number of 6001.

After a TCP connection is established, each segment is tagged with a sequence number. TCP detects that a segment has been lost when it does *not* receive a corresponding acknowledgement of receipt. It must not only receive an **ACK**, but it must receive an ACK with the appropriate acknowledgement number.

(Reference: http://www.tcpipguide.com/free/t_TCPConnectionEstablishmentSequenceNumberSynchroniz.htm)

Additionally, TCP incorporates **windowing** for **flow control**. When flow control is enabled, both the sending and receiving devices must agree on the amount of data being sent in between acknowledgements. This helps prevent data loss due to one side of the connection being overloaded.

(Reference: http://www.tcpipguide.com/free/t_TCPSlidingWindowAcknowledgmentSystemForDataTranspo.htm)

* * *

All original material copyright © 2006 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

The TCP Header

The TCP header has **12 fields**:

<i>Field</i>	<i>Length</i>	<i>Description</i>
Source Port	16 bits	<i>Source TCP Port</i>
Destination Port	16 bits	<i>Destination TCP Port</i>
Sequence Number	32 bits	<i>Initial Sequence Number</i>
Ack Number	32 bits	<i>Acknowledgement Number</i>
Data Offset	4 bits	<i>Indicates where the data begins in a TCP segment</i>
Reserved	6 bits	<i>Always set to 0</i>
Control Bits	6 bits	<i>URG, ACK, PSH, RST, SYN, and FIN flags</i>
Window	16 bits	<i>Used for Flow Control</i>
Checksum	16 bits	<i>Used for Error-Checking</i>
Urgent Pointer	16 bits	
Options	Variable	
Padding	Variable	<i>To ensure the TCP header ends at a 32 bit boundary</i>

User Datagram Protocol (UDP)

The **User Datagram Protocol (UDP)** is defined as an unreliable, **connectionless** transport protocol. It is essentially a stripped-down version of TCP, and thus has *far less latency* than TCP.

UDP provides no three-way handshake, no flow-control, no sequencing, and no acknowledgment of data receipt. However, UDP does provide basic error-checking using a checksum.

The UDP header has only **4 fields**:

<i>Field</i>	<i>Length</i>	<i>Description</i>
Source Port	16 bits	<i>Source UDP Port</i>
Destination Port	16 bits	<i>Destination UDP Port</i>
Length	16 bits	<i>Length of the header and the data</i>
Checksum	16 bits	<i>Used for Error-Checking</i>

3

All original material copyright © 2006 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

Comparison of TCP versus UDP

TCP	UDP
Connection-oriented	Connection-less
Guaranteed Delivery	No Guaranteed Delivery
Sends Acknowledgments	Does not send Acknowledgments
Reliable, but slow	Unreliable, but fast
Segments and Sequences Data	Does NOT segment/sequence data
Flow Control	No Flow Control
Performs CRC on data	Performs CRC on data

TCP/UDP Ports

TCP and UDP **ports** identify services that run on a specific logical address. Otherwise, there would be no way to distinguish data destined for one service or another on a device. For example, port numbers allow both a web and email server to operate simultaneously on the same address.

An IP address combined with a TCP or UDP port forms a **socket**. A socket is written out as follows:

10.50.1.1:80

Specific ports (1-1024) have been reserved for specific services, and are recognized as **well-known** ports. Below is a table of several common TCP/UDP ports:

20, 21	TCP	FTP
22	TCP	SSH
23	TCP	Telnet
25	TCP	SMTP
53	UDP	DNS
80	TCP	HTTP
110	TCP	POP3
443	TCP	SSL
666	TCP	Doom

For a complete list of port numbers, refer to the IANA website:
<http://www.iana.org/assignments/port-numbers>.

* * *