1) Differentiate between DOD and the OSI network Models

DOD: The Department of Defense four-layer model for the DARPA Internet Project, which gradually evolved into the internet, was developed in the '70s. This model is followed by the core internet protocols, while the OSI seven-layer model is preferred with respect to modern architectures.

From the bottom to the top, the four layers of the DoD model are:

* The Network Access Layer provides the data via the hardware media in use. Depending on the form of physical network, various protocols are chosen from this layer.
* The Internet Layer delivers data over a variety of physical networks linking a source and a destination system. Routing protocols, as the IP Protocol, are the fundamental protocol of the Internet, are most closely related to this layer.
* The Host-to - Host Layer manages connection appointments, flow control, data transmittal and other generic management of data flow. The most significant members of this layer are the mutually incompatible TCP and UDP protocols.
* The process layer includes protocols that execute functions on a user level, including mailing, file transfer and remote login.

The OSI model:

The Open Systems Interconnection (OSI) model is a conceptual model created by the International Organization for Standardization which enables diverse communication systems to communicate using standard protocols. In plain English, the OSI provides a standard for different computer systems to be able to communicate with each other.

The OSI model be a universal language for computer networking. It is based on the concept of splitting up a communication system into seven abstract layers, each one stacked upon the last.

1. Physical layer: Transmits raw bit stream over the physical medium.
2. Data link layer: Defines the format of data on the network.
3. Network layer: Decides which physical path the Data will take.
4. Transport layer: Transmits Data using transmission protocols including TCP and UDP.
5. Session layer: Maintains connections and is responsible for controlling ports and sessions
6. Presentation layer: Ensures data is in a usable format and is where data encryption occurs.
7. Application layer: Human-computer interaction layer, where applications can access the network services.

2) Identify Host-to-Host Layer Protocols:

**Host-To-Host Protocols:**

|  |  |
| --- | --- |
| **Transport Control Protocol (TCP):** | Allows users to access resources on another machine. All data is seen in clear text (not recommended for use) |
| **User Datagram Protocol (UDP):** | Similar to Telnet but it sets up a secure session (recommended over telnet). All data is encrypted during the session |

 3) Identify Internet Layer Protocols:

Internet Layer Protocols:

* **Internet Protocol (IP) – analyze each packet to decide where the packet is sent**
* **Internet Control Message Protocol (ICMP) – Use to gives status updates about a host or network**
* **Address Resolution Protocol (ARP) – resolves IP addresses to MAC addresses**

 4) Compare UDP and TCP Characteristics and Features

TCP vs UDP (Differences)

|  |  |  |
| --- | --- | --- |
| **Acronym for** | Transmission Control Protocol | User Datagram Protocol or Universal Datagram Protocol |
| **Connection** | Transmission Control Protocol is a connection-oriented protocol. | User Datagram Protocol is a connectionless protocol. |
| **Function** | As a message makes its way across the [internet](https://www.diffen.com/difference/Internet_vs_World_Wide_Web) from one computer to another. This is connection based. | UDP is also a protocol used in message transport or transfer. This is not connection based which means that one program can send a load of packets to another and that would be the end of the relationship. |
| **Usage** | TCP is suited for applications that require high reliability, and transmission time is relatively less critical. | UDP is suitable for applications that need fast, efficient transmission, such as games. UDP's stateless nature is also useful for servers that answer small queries from huge numbers of clients. |
| **Use by other protocols** | HTTP, HTTPs, FTP, SMTP, Telnet | DNS, DHCP, TFTP, SNMP, RIP, VOIP. |
| **Ordering of data packets** | TCP rearranges [data](https://www.diffen.com/difference/Data_vs_Information) packets in the order specified. | UDP has no inherent order as all packets are independent of each other. If ordering is required, it has to be managed by the application layer. |
| **Speed of transfer** | The speed for TCP is slower than UDP. | UDP is faster because error recovery is not attempted. It is a "best effort" protocol. |
| **Reliability** | There is absolute guarantee that the data transferred remains intact and arrives in the same order in which it was sent. | There is no guarantee that the messages or packets sent would reach at all. |
| **Header Size** | TCP header size is 20 bytes | UDP Header size is 8 bytes. |
| **Streaming of data** | Data is read as a byte stream, no distinguishing indications are transmitted to signal message (segment) boundaries. | Packets are sent individually and are checked for integrity only if they arrive. Packets have definite boundaries which are honored upon receipt, meaning a read operation at the receiver socket will yield an entire message as it was originally sent. |
| **Weight** | TCP is heavy-weight. TCP requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control. | UDP is lightweight. There is no ordering of messages, no tracking connections, etc. It is a small transport layer designed on top of IP. |
| **Data Flow Control** | TCP does Flow Control. TCP requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control. | UDP does not have an option for flow control |
| **Error Checking** | TCP does error checking and error recovery. Erroneous packets are retransmitted from the source to the destination. | UDP does error checking but simply discards erroneous packets. Error recovery is not attempted. |
| **Fields** | 1. Sequence Number, 2. AcK number, 3. Data offset, 4. Reserved, 5. Control bit, 6. Window, 7. Urgent Pointer 8. Options, 9. Padding, 10. Check Sum, 11. Source port, 12. Destination port | 1. Length, 2. Source port, 3. Destination port, 4. Check Sum |
| **Acknowledgement** | Acknowledgement segments | No Acknowledgment |
| **Handshake** | SYN, SYN-ACK, ACK | No handshake (connectionless protocol) |

(Similarities) **TCP vs UDP**

|  |  |  |
| --- | --- | --- |
| **Common Header Fields** | Source port, Destination port, Check Sum | Source port, Destination port, Check Sum |

  5) a) What are the ranges of port numbers

The port numbers in the range from 0 to 1023 (0 to 210 − 1) are the *well-known ports* or *system ports*.[[2]](https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers#cite_note-rfc6335-2)

b) Also name 3 types of port numbers.

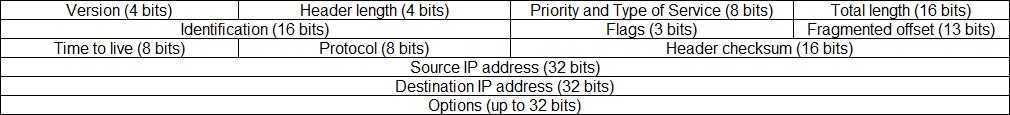
 The well-known **ports**

The registered **ports**

The dynamic or private **ports**

 6) Identify what is contained in the IP header

An **IP header** is a prefix to an IP packet that contains information about the IP version, length of the packet, source and destination IP addresses, etc. It consists of the following fields:

[](https://x9s4w2e2.stackpathcdn.com/wp-content/uploads/2016/03/ip_header.jpg)

Here is a description of each field:

* **Version** – the version of the IP protocol. For IPv4, this field has a value of 4.
* **Header length** – the length of the header in 32-bit words. The minumum value is 20 bytes, and the maximum value is 60 bytes.
* **Priority and Type of Service** – specifies how the datagram should be handled. The first 3 bits are the priority bits.
* **Total length** – the length of the entire packet (header + data). The minimum length is 20 bytes, and the maximum is 65,535 bytes.
* **Identification** – used to differentiate fragmented packets from different datagrams.
* **Flags** – used to control or identify fragments.
* **Fragmented offset** – used for fragmentation and reassembly if the packet is too large to put in a frame.
* **Time to live** – limits a datagram’s lifetime. If the packet doesn’t get to its destination before the TTL expires, it is discarded.
* **Protocol** – defines the protocol used in the data portion of the IP datagram. For example, TCP is represented by the number 6 and UDP by 17.
* **Header checksum** – used for error-checking of the header. If a packet arrives at a router and the router calculates a different checksum than the one specified in this field, the packet will be discarded.
* **Source IP address** – the IP address of the host that sent the packet.
* **Destination IP address** – the IP address of the host that should receive the packet.
* **Options** – used for network testing, debugging, security, and more. This field is usually empty.