**EO 001.04.02 – IP Network Protocol Reading Assignment**

**Transmission Control Protocol (TCP)**

A connection-oriented communications protocol that facilitates the exchange of messages between computing devices in a network. It is the most common protocol in networks that use the Internet Protocol (IP); together they are sometimes referred to as TCP/IP.

TCP takes messages from an application/server and divides them into packets, which can then be forwarded by the devices in the network – switches, routers, security gateways – to the destination. TCP numbers each packet and reassembles them prior to handing them off to the application/server recipient. Because it is connection-oriented, it ensures a connection is established and maintained until the exchange between the application/servers sending and receiving the message is complete.

For example, when an email (using the simple mail transfer protocol – SMTP) is sent from an email server, the TCP layer in that server will divide the message up into multiple packets, number them and then forward them to the IP layer for transport. At the IP layer, each packet will be transported to the destination email server. While each packet is going to the same place, the route they take to get there may be different. When it arrives, the IP layer hands it back to the TCP layer, which reassembles the packets into the message and hands it to the email application, where it shows up in the Inbox.

**What is User Datagram Protocol (UDP/IP)?**

The User Datagram Protocol, or UDP, is a communication protocol used across the Internet for especially time-sensitive transmissions such as video playback or DNS lookups. It speeds up communications by not formally establishing a connection before data is transferred. This allows data to be transferred very quickly, but it can also cause packets to become lost in transit — and create opportunities for exploitation in the form of DDoS attacks.

**How does UDP work?**

Like all networking protocols, UDP is a standardized method for transferring data between two computers in a network. Compared to other protocols, UDP accomplishes this process in a simple fashion: it sends packets (units of data transmission) directly to a target computer, without establishing a connection first, indicating the order of said packets, or checking whether they arrived as intended. (UDP packets are referred to as ‘datagrams’.)

UDP is faster but less reliable than TCP, another common transport protocol. In a TCP communication, the two computers begin by establishing a connection via an automated process called a ‘handshake.’ Only once this handshake has been completed will one computer actually transfer data packets to the other.

UDP communications do not go through this process. Instead, one computer can simply begin sending data to the other:

**TCP vs UDP Communication**

In addition, TCP communications indicate the order in which data packets should be received and confirm that packets arrive as intended. If a packet does not arrive — e.g. due to congestion in intermediary networks — TCP requires that it be re-sent. UDP communications do not include any of this functionality.

These differences create some advantages. Because UDP does not require a ‘handshake’ or check whether data arrives properly, it is able to transfer data much faster than TCP

However, this speed creates tradeoffs. If a UDP datagram is lost in transit, it will not be re-sent. As a result, applications that use UDP must be able to tolerate errors, loss, and duplication.

Technically, such packet loss is less a flaw in UDP than a consequence of how the Internet is built. Most network routers do not perform packet ordering and arrival confirmation by design, because doing so would require an unfeasible amount of additional memory. TCP is a way of filling this gap when an application requires it.

**Multicast**

Multicast networking is based on the simple concept that a single packet can be sent by a server and it will be received by many receivers. Multicast is different from broadcast because it’s more selective.

Where broadcast packets are received by all receivers in a particular network segment (or broadcast domain), multicast packets are received only by receivers that want them. Also, multicast receivers can be distributed throughout a larger network behind routers.

**Primary uses for multicast networking**

In IPv6 networks, there’s no such thing as broadcast. Multicast is used for everything that broadcast was previously used for, including a number of standard network infrastructure things like router discovery, address allocation, and neighbour discovery (which replaces ARP).

Multicast has a couple of fundamental characteristics that dictate how it’s used. Because a server only needs to send each packet once and will reach all of the recipients, it’s useful for situations where a large number of receivers need to receive the same data. Since the replication and distribution of these packets is done by the network rather than the head end server, it scales well to extremely large numbers of receivers.

But because the multicast is one-way, any responses would need to be implemented using a separate protocol. This also means that dropped packets must either be unimportant, or the recovery mechanisms for lost data must be built separately.

**Unicast**

Typical communications over an IP-based network are directed unicast communications. Unicast is basically a single, direct request sent from one host to another, and only the two hosts interact over the established route. For example, when you click a hyperlink in a Web browser, you are requesting HTTP data from the host defined in the link, which, in turn, delivers the data to your browser. This is useful in the Web-browsing environments we have grown accustomed to, where there is a demand for a personal, user-controlled experience.

Unicast is not useful for delivering streams of audio or video to large audiences, since a single stream of audio/video data is very costly for only one user. This is where multicast communications are effective. Multicast provides a single stream for multiple hosts. The hosts select the data by requesting the local routers to forward those packets of data from the host providing the multicast data to the subnet of the listening host. When the host decides to stop listening to the multicast traffic, IGMP is responsible for notifying the router that the host is no longer participating.

**RARP**

What happens if your own computer does not know its IP address, because it has no storage capacity, for example? In these cases, the Reverse Address Resolution Protocol (RARP) can help. The RARP is the counterpart to the ARP – the Address Resolution Protocol.

The Reverse ARP is now considered obsolete, and outdated. Newer protocols such as the Bootstrap Protocol (BOOTP) and the Dynamic Host Configuration Protocol (DHCP) have replaced the RARP. However, it is useful to be familiar with the older technology as well. For instance, you can still find some applications which work with RARP today. It also helps to be familiar with the older technology in order to better understand the technology which was built on it.

**What is the RARP?**

The RARP is a protocol which was published in 1984 and was included in the TCP/IP protocol stack. The RARP is on the Network Access Layer (i.e. the lowest layer of the TCP/IP protocol stack) and is thus a protocol used to send data between two points in a network. Each network participant has two unique addresses more or less: a logical address (the IP address) and a physical address (the MAC address). While the IP address is assigned by software, the MAC address is built into the hardware. You have already been assigned a Media Access Control address (MAC address) by the manufacturer of your network card.

It is possible to not know your own IP address. This may happen if, for example, the device could not save the IP address because there was insufficient memory available. In such cases, the Reverse ARP is used. This protocol can use the known MAC address to retrieve its IP address. Therefore, its function is the complete opposite of the ARP. The ARP uses the known IP address to determine the MAC address of the hardware.

**How does the RARP work?**

Who knows the IP address of a network participant if they do not know it themselves? A special RARP server does. This server, which responds to RARP requests, can also be a normal computer in the network. However, it must have stored all MAC addresses with their assigned IP addresses. If a network participant sends an RARP request to the network, only these special servers can respond to it.

Since the requesting participant does not know their IP address, the data packet (i.e. the request) must be sent on the lowest layers of the network as a broadcast. This means that the packet is sent to all participants at the same time. However, only the RARP server will respond. If there are several of these servers, the requesting participant will only use the response that is first received. The request-response format has a similar structure to that of the ARP.