***TASCA 5***

***ENCAPSULAMIENTO***

Nom: Alan Romeo Delzoto

Asignatura: Redes Locales

Data: 22/10/2024

**ÍNDICE**

[**1. Explica el procés d’encapsulament. Què són les capçaleres i les cues, i com s’afegeixen**](#_4qpo687tiqw) **3**

[**2. Explica el procés d’enviament d’un email entre un emissor i receptor sense cap node intermig. Realitza-ho fent servir el model OSI, explicant què realitza cada capa i com se encapsulen les dades per passar a la capa inferior o superior segons el moment.**](#_djrab9kr9sln) **3**

[**3. Explica el procés d’enviament de dades (protocol) que empra la capa de transport**](#_h12ne9hngxnq) **4**

# **Explica el procés d’encapsulament. Què són les capçaleres i les cues, i com s’afegeixen**

El encapsulamiento es el proceso mediante el cual los datos se preparan para ser transmitidos a través de una red. Este proceso implica añadir información adicional a los datos originales para garantizar que se puedan enviar, recibir e interpretar correctamente. Esta información adicional se añade en forma de cabeceras y colas a cada capa del modelo de referencia OSI (o del modelo TCP/IP), dependiendo del protocolo que se esté utilizando.

**Las cabeceras que tiene son 2 y son:**

**Cabeceras:** Son bloques de información que se añaden a primeros de los datos encapsulados en cada capa. Incluyen información necesaria para el proceso de transmisión, como direcciones, tipos de servicio, números de puertos, etc.

**Colas:** Se pueden añadir a algunas capas (especialmente a la capa de enlace de datos) al final de los datos encapsulados y sirven principalmente para la detección de errores u otras funcionalidades de control.

**Y las colas funcionan así:**

Cuando los datos llegan al destinatario, el proceso de encapsulamiento se hace a la inversa, se van eliminando las cabeceras y colas a medida que los datos pasan hacia arriba por las capas hasta que llegan a la aplicación final.

Este proceso asegura que los datos se pueden enviar, recibir e interpretar correctamente dentro de una red de comunicaciones.

# **Explica el procés d’enviament d’un email entre un emissor i receptor sense cap node intermig. Realitza-ho fent servir el model OSI, explicant què realitza cada capa i com se encapsulen les dades per passar a la capa inferior o superior segons el moment.**

**7. Capa de Aplicación:** El cliente de email (Outlook, Gmail) crea y envía el correo usando el protocolo SMTP.

**6. Capa de Presentación:** Si es necesario, los datos del email se traducen o se cifran.

**5. Capa de Sesión:** Se establece y mantiene la conexión entre el emisor y receptor.

**4. Capa de Transporte:** El mensaje se divide en segmentos que son ordenados y verificados para asegurar que lleguen correctamente usando TCP.

**3. Capa de Red:** Los segmentos se encapsulan en paquetes IP y se dirigen usando las direcciones IP del emisor y receptor.

**2. Capa de Enlace de Datos:** Los paquetes IP se encapsulan en tramas que incluyen direcciones MAC para la red local.

**1. Capa Física:** Las tramas se transmiten como señales eléctricas, de luz o radio a través del medio de transmisión.

Y luego el receptor para responder al emisor realiza el proceso inverso al que he descrito.

# **Explica el procés d’enviament de dades (protocol) que empra la capa de transport**

**Proceso de TCP:**

Establecimiento de conexión (Three-way handshake):

* Primero, se establece una conexión fiable entre el emisor y el receptor. El proceso se conoce como "handshake de tres pasos" o three-way handshake.
  1. **SYN:** El emisor envía un mensaje de sincronización (SYN) para iniciar la conexión.
  2. **SYN-ACK:** El receptor responde con un mensaje de sincronización y confirmación (SYN-ACK).
  3. **ACK:** El emisor envía una confirmación (ACK), estableciendo así la conexión.

**Segmentación:**

* Los datos grandes se dividen en fragmentos más pequeños llamados segmentos, que contienen información como el número de secuencia, el puerto de origen y destino, y un checksum para detectar errores.

**Control de flujo:**

* TCP utiliza mecanismos como sliding window (ventana deslizante) para controlar la cantidad de datos que se pueden enviar sin esperar la confirmación (ACK) del receptor, evitando saturar al receptor con demasiados datos de golpe.

**Retransmisión y corrección de errores:**

* Si un segmento no llega al receptor o llega con errores (detectados gracias al checksum), el receptor no envía el ACK correspondiente. TCP detecta la ausencia del ACK y retransmite el segmento perdido.

**Reordenamiento de segmentos:**

* Si los segmentos llegan fuera de orden, TCP se encarga de reordenarlos utilizando los números de secuencia antes de entregar los datos a la capa superior.

**Cierre de la conexión:**

* Cuando el emisor ha terminado de enviar datos, se cierra la conexión mediante un proceso de "handshake" de cuatro pasos (FIN, ACK, FIN, ACK), que garantiza que ambos extremos reconozcan el final de la transmisión.

**Proceso de UDP:**

**Envío sin conexión:**

* UDP no establece una conexión previa entre el emisor y el receptor, simplemente envía los mensajes llamados datagramas. No hay ningún handshake como en TCP.

**Segmentación mínima:**

* Los datos se dividen en datagramas, cada uno con su propio encabezado, que contiene información como el puerto de origen, el puerto de destino, la longitud del mensaje y un checksum opcional para detectar errores básicos.

**Sin control de flujo ni corrección de errores:**

* Si un datagrama se pierde o llega con errores, UDP no ofrece mecanismos para retransmitirlo ni corregirlo. Tampoco asegura que los datagramas lleguen en el orden correcto.

**Fin del proceso:**

* No hay cierre de conexión. El emisor simplemente deja de enviar datagramas cuando ha terminado.