Redes de Computadoras

Solución Examen – 15 de diciembre de 2023
(ref: sol20231215.odt)

Instrucciones

- **Indique su nombre completo y número de cédula en cada hoja.**
- **Numere todas las hojas e indique la cantidad total de hojas que entrega en la primera.**
- **Escriba las hojas de un solo lado y utilice una caligrafía claramente legible.**
- **Comience cada pregunta teórica y cada ejercicio en una hoja nueva.**
- **Solo se responderán dudas de letra. No se responderán dudas de ningún tipo los últimos 30 minutos del examen.**
- **El examen es individual y sin material. Apague su teléfono celular mientras esté en el salón del examen.**
- **Es obligatorio responder correctamente al menos 15 puntos en las preguntas teóricas y 20 de los problemas prácticos. Los puntos ganados en el curso se suman a los puntos de teórico.**
- **El puntaje mínimo de aprobación es de 60 puntos.**
- **Para todos los ejercicios, si es necesario, puede suponer que dispone de los tipos de datos básicos (p.ej. lista, cola, archivo, string, etc.) y sus funciones asociadas (ej: tail(lista), crear(archivo), concatenar(string, string).**
- **Justifique todas sus respuestas.**
- **Duración: 3 horas. Culminadas las 3 horas el alumno no podrá modificar las hojas a entregar de ninguna forma.**

Preguntas Teóricas

Pregunta 1 (8 puntos)

En el servicio de DNS se encuentran configurados, entre otros, los siguientes registros:

```
cert.uy                 IN NS           dns06.agesic.gub.uy
cert.uy                 IN NS           dns05.agesic.gub.uy
dns06.agesic.gub.uy.    IN      A               179.27.169.254
dns06.agesic.gub.uy.    IN      AAAA            2800:a8:a02:2:179:27:169:254
dns05.agesic.gub.uy.    IN      A               179.27.169.190
```

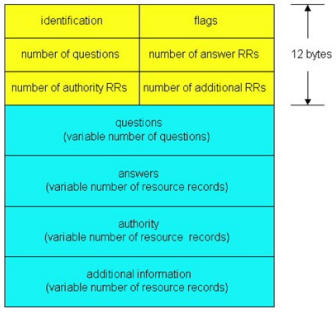
Desde una computadora se realiza una consulta recursiva a un servidor no autoritativo por el servidor de nombres del dominio `cert.uy`.

a) Muestre el formato y contenido del mensaje *query* enviado.
b) Muestre el formato y contenido del mensaje *reply* recibido.

Todos los detalles cuentan, sea exhaustivo en su respuesta.

Solución

El formato de los mensajes query y reply (o query response) es el mismo, el que se muestra a continuación.



a)
Lleva un Identificador (Transaction ID) para asociar el mensaje reply al query, generado por quien crea el mensaje query.
Flags
Existe una flag (1 bit) que indica el tipo de mensaje: si está en 0 es una query (este caso).
Otra flag (1 bit) indica si se pretende que sea una query recursiva.
Lleva un Identificador (Transaction ID) para asociar el mensaje reply al query.
Los valores de los campos “contadores” (“number of...”) son los siguientes:
Questions: 1
Answers: 0
Authority: 0
Additional: 0

En el campo Questions el contenido es
cert.uy: type NS, class IN
No existen campos Answers, Authority ni Additional.

b)
Lleva el Identificador copiado del mensaje query que lo motiva
Flags
Flag (1 bit) en 1 indicando que es un reply (query response)
Otra flag (1 bit) indica si se pretende que sea una query recursiva.
Otra flag (1 bit) indica si se dispone de servicio recursivo.
Los valores de los campos “contadores” (“number of...”) son los siguientes:
Questions: 1
Answers: 2
Authority: 0
Additional: 3
En el campo Questions el contenido es
cert.uy: type NS, class IN
En el campo Answers el contenido es
cert.uy: type NS, class IN, ns `dns06.agesic.gub.uy`
cert.uy: type NS, class IN, ns `dns05.agesic.gub.uy`
No existe campo Authority.
En el campo Additional el contenido es
`dns06.agesic.gub.uy`: type A, class IN, addr `179.27.169.254`
`dns06.agesic.gub.uy`: type AAAA, class IN, addr `2800:a8:a02:2:179:27:169:254`
`dns05.agesic.gub.uy`: type A, class IN, addr `179.27.169.190`

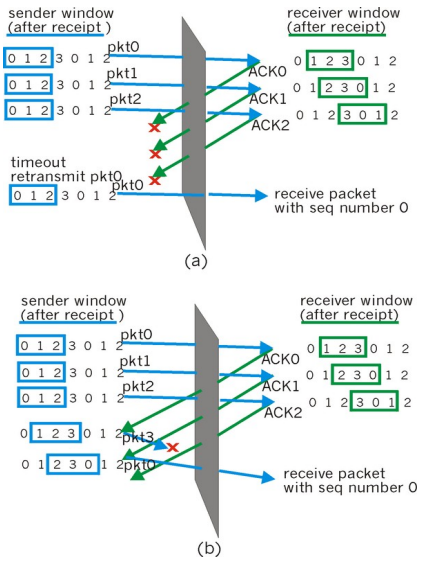
Pregunta 2 (8 puntos)

a) Explique las diferencias entre protocolos de transporte stop-and-wait y en pipeline.
b) De un ejemplo del problema de tener tamaños de ventana muy parecidos al rango de números de secuencia en protocolos de tipo Selective Repeat.

Solución:

a) En un protocolo stop-and-wait, el emisor espera por el reconocimiento de un paquete para empezar la transmisión de un nuevo paquete.
En el caso de los protocolos de pipeline, el emisor puede enviar varios paquetes sin esperar a los mensajes de reconocimiento. Para esto es necesario definir una ventana de transmisión que definirá cuantos paquetes se pueden transmitir “simultáneamente” sin haber recibido un reconocimiento.

b) Para el caso de un protocolo de pipeline, una opción es utilizar selective repeat para el reconocimiento de los paquetes. Esto implica que el receptor genera un paquete de reconocimiento por cada paquete recibido. Del lado del emisor, solo se reenvían paquetes que no fueron reconocidos luego de un cierto tiempo.
Imaginemos un escenario con una ventana de transmisión de 3 paquetes y número de secuencia del 0 al 3. Se ilustra a continuación un posible problema:



En la imagen (a) se puede observar que el receptor no detecta que el emisor está enviando un mensaje duplicado y considera que es un nuevo paquete.
Ver que no puede diferenciarlo del caso (b) donde es un nuevo paquete con secuencia 0.

Pregunta 3 (8 puntos)

Mencione y compare las principales características de los algoritmos de enrutamiento de estado de enlace (LS, *link-state*) con las de los algoritmos de vector distancia (DV, *distance-vector*).

Solución:

Se analizan los siguientes 4 tópicos: cuál es centralizado y cuál distribuido, qué "habla" cada nodo, con quiénes "habla" nodo, y qué parte de la red conoce cada nodo.
Un algoritmo de enrutamiento centralizado calcula la ruta de coste mínimo entre un origen y un destino utilizando un conocimiento global y completo acerca de la red.
En un algoritmo de enrutamiento descentralizado, el cálculo de la ruta de coste mínimo se realiza por parte de los routers de manera iterativa y distribuida. Ningún nodo tiene toda la información acerca del coste de todos los enlaces de la red.
Los algoritmos con información de estado global a menudo se denominan algoritmos de estado de enlaces (LS, Link-State), ya que el algoritmo tiene que ser consciente del coste de cada enlace de la red.
El algoritmo de enrutamiento descentralizado que estudiamos en el curso se denomina algoritmo de vector de distancias (DV, Distance-Vector), porque cada nodo mantiene un vector de estimaciones de los costes (distancias) a todos los demás nodos de la red.
En un algoritmo de estado de enlaces, la topología de la red y el coste de todos los enlaces son conocidos; es decir, están disponibles como entradas para el algoritmo LS. En la práctica, esto se consigue haciendo que cada nodo difunda paquetes del estado de los enlaces a todos los demás nodos de la red, conteniendo cada paquete de estado de enlaces las identidades y costes de sus enlaces conectados.
El resultado de difundir la información por parte de los nodos es que todos los nodos tienen una visión completa e idéntica de la red. Cada nodo puede entonces ejecutar el algoritmo LS y calcular el mismo conjunto de rutas de coste mínimo que cualquier otro nodo.
Mientras que el algoritmo LS es un algoritmo que emplea información global, el algoritmo por vector de distancias (DV) es iterativo, asíncrono y distribuido. Es iterativo porque este proceso continúa hasta que ya no se intercambia más información entre los vecinos. El algoritmo es asíncrono en el sentido de que no requiere que todos los nodos operen sincronizados entre sí.
Es distribuido en el sentido de que cada nodo recibe información de uno o más de sus vecinos directamente conectados, realiza un cálculo y luego distribuye los resultados de su cálculo de vuelta a sus vecinos.

Pregunta 4 (8 puntos)

a) Defina que es una ruta en el contexto de BGP.
b) Defina los dos atributos principales de una ruta BGP.
c) ¿Cuándo es necesario que un router ejecute el algoritmo de selección de ruta de BGP?
d) Mencione al menos cuatro pasos del algoritmo de selección de ruta de BGP.

Solución

a) Una ruta está compuesta por un prefijo de red acompañado de sus atributos. Este objeto se denomina *Network Layer Reachability Information (NLRI)*, y se anuncia en los mensajes UPDATE de BGP.

b) Los principales atributos de una ruta son AS-PATH y NEXT-HOP. El AS-PATH contiene la lista de los Números de Sistemas Autónomos (*ASN – Autonomous System Number*) por los que ha pasado el anuncio, mientras que el NEXT-HOP es la dirección IP del router que inicia la secuencia de ASNs (AS-PATH). En una sesión BGP externa (eBGP), el router cambia el atributo NEXP-HOP de una ruta BGP (a su propia dirección IP) cuando envía el anuncio de dicha ruta.

c) El algoritmo de selección de ruta se debe aplicar cunado existen varias rutas posibles para el mismo prefijo. Este algoritmo busca "desempatar" entre todas las opciones, eligiendo una sola de ellas.

d)
1) Se comparan los valores de Preferencia Local (*local-pref*), y gana el más grande. Este es un valor administrativo que se fija en base a la política del Sistema Autónomo. Por ejemplo, un AS que tiene dos proveedores, puede asignar mayor *local-pref* a uno de ellos por precio, confiabilidad del enlace u otras razones.

2) Si luego de aplicar la regla anterior quedan rutas empatadas (con igual local-pref), se compara el largo del atributo AS-PATH (*AS-PATH lenght*), y gana la más corta, es decir, que *AS-PATH lenght* se usa como una métrica de vector-distancia.
3) Para las rutas que siguen empatadas se aplica el "enrutamiento de papa caliente": gana la ruta con menor costo al router de borde.
4) Si en los pasos anteriores no se ha logrado desempatar, se utilizará el "Indentificador BGP" (*BGP ID*) como mecanismo de desempate. Se prefiere la ruta que proviene del router con *BGP ID* más bajo. Este valor es la dirección IP más alta del router, dándose preferencia a las direcciones de *loopback*. Además, se puede fijar administrativamente por configuración del router.

Pregunta 5 (8 puntos)

a) Defina los términos dominio de colisión y dominio de broadcast.
b) Explique cuál es la función fundamental de un conmutador de capa 2 (switch)
c) Los puertos de un switch, ¿tienen direcciones MAC? Justifique su respuesta.

Solución

a) En un contexto de tecnología Ethernet:
1) Un dominio de colisión es el conjunto de dispositivos que pueden sufrir colisiones al transmitir una trama. En la implementación original de Ethernet con todos los hosts de una red de área local conectados a un bus (o un hub), el dominio de colisión son todos los hostos conectados al medio. Sin embargo, en un medio conmutado (con switches), el dominio de colisión se restringe a cada host y puerto del switch que comparten un cable.
2) Por otro lado, el dominio de broadcast es el conjunto de hosts que forman parte de una red de área local (Local Area Network - LAN), cuya frontera es un router. A todos estos hosts le puede llegar una trama de broadcast (dirección destino FF-FF-FF-FF-FF-FF).

b) La función fundamental de un switch es recibir tramas por los puertos entrantes y reenviarlas por los puertos de salida, en base a una tabla de conmutación que se construye mediante la técnica del auto-aprendizaje.

c) No, los switches son trasnparentes, en el sentido que los hosts de una LAN desconocen su existencia, y envían tramas ethernet cuyos orígenes y destinos son los propios hosts de la LAN. Nunca es posible enviar una trama cuyo destino sea el switch, ya que como se mencionó, no tiene direcciones MAC.

Problema 1 (35 puntos)

La conectividad de la organización *Aiouti* está implementada mediante un router R1 con 4 interfaces de red: eth1..eth4.
El uso de las 4 interfaces es: eth1 para la red LAN1, eth2 para la red LAN2, eth3 para las redes LAN3 y LAN4, y eth4 para la red WAN.
R1 se interconecta con la interfaz eth4 de un router R2 administrado por el ISP (quien ofrece acceso a Internet), a través de un enlace punto a punto en la red WAN.
A la red LAN1 potencialmente se pueden conectar 5.000 dispositivos, pero se asegura que de forma simultánea nunca más de 400. Dichos dispositivos sólo deberán estar accesibles desde las estaciones de trabajo ubicadas en la red LAN3.
En la red LAN2 se deben conectar 8 servidores: DNS autoritativo, portal web y correo corporativo entre otros, los que deben estar accesibles desde Internet y desde todas las redes LAN de la organización.
Las redes LAN3 y LAN4 se implementan mediante un *switch* SW1 que soporta VLANs, y donde se conectan 200 estaciones de trabajo a LAN3 y 4 a LAN4; de las estaciones de trabajo de LAN3 y LAN4 sólo las de esta última deberán poder acceder a Internet y sí se permitirá la conectividad entre LAN3 y LAN4.
Las cantidades mencionadas no cambiarán en el futuro.
El dominio de la organización es `aiouti.uy`.

a) Elabore un diagrama de la topología de red de la organización, incluyendo a R2.
b) Realice un plan de numeración de toda la red de la organización optimizando el uso de las direcciones IP, considerando que se dispone del prefijo 190.20.30.0/23, posibilitando la conectividad requerida y justificando todas las decisiones de diseño tomadas. Para la red WAN el ISP asignó el prefijo 200.12.200.12/30.
c) Muestre las tablas de *forwarding* para cada uno de los siguientes equipos:
   i) R1
   ii) Dispositivo en LAN1
   iii) Servidor en LAN2

iv) Estación de trabajo en LAN3
v) Estación de trabajo en LAN4
vi) R2 – sólo lo que refiere a la organización *Aiouti*

d) Si el primer tráfico que se observa en la red es el ocasionado por ejecutar en la estación de trabajo 6 de LAN3 (su nombre es `et6lan3.aiouti.uy`) un comando `ping` dirigido al nombre de la estación de trabajo 1 de LAN4 (su nombre es `et1lan4.aiouti.uy`), indique la secuencia de tramas que se observan en el *switch* SW1 como consecuencia de ello. Suponga que el `ping` ejecutado es exitoso.

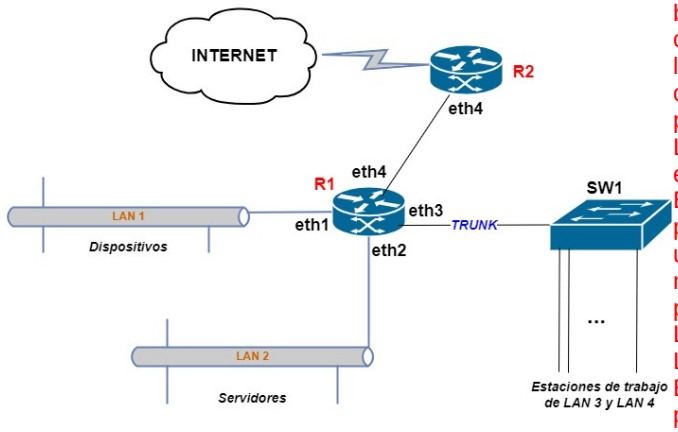
Para cada trama justifique se ocurrencia e indique claramente encabezados y carga útil de cada capa que corresponda.

Para las direcciones IP considere las asignaciones realizadas en la parte b).

Para las direcciones MAC asigne todas las que considere necesarias; para ello utilice el formato MAC*X*, donde *X* identifica de forma unívoca una interfaz de red.

Solución

a)



b) Se dispone del prefijo 190.20.30.0/23. Siempre que corresponda se comentan las decisiones de diseño. El resto de las decisiones de diseño se plasman a través de la configuración de las tablas de forwarding que se muestran en la parte c). La demanda de direcciones IP efectivas es:
LAN 1 - 400 (por la cantidad máxima de dispositivos conectados en cada momento)
En LAN 1 se debería desplegar un servidor DHCP. Un servidor propiamente o un servicio activado en R1 sirviendo a LAN 1. Es un requisito indispensable para la gestión de direcciones IP necesaria para la enorme cantidad de dispositivos que se podrán estar conectando y desconectando (5000)
LAN 2 – 8
LAN 3 – 200
En LAN 3 se debería desplegar un servidor DHCP. Un servidor propiamente o un servicio activado en R1 sirviendo a LAN 3. Es un requisito indispensable para la gestión de direcciones IP necesaria para la enorme cantidad de dispositivos que se podrán estar conectando y desconectando (200)
LAN 4 – 4
Claramente el prefijo disponible no es suficiente para contemplar la demanda de direcciones IP de las 4 redes LAN.
Afortunadamente la demanda de la red LAN 1 se puede satisfacer con direcciones IP privadas pues no hay necesidad de que sean accesibles desde Internet.
Por lo tanto, para LAN 1 asignamos un prefijo de direcciones IP privadas, por ejemplo: 172.31.0.0/23 (máscara 255.255.254.0).
Asignamos la dirección IP 172.31.0.1 a la interfaz eth1 de R1.
El resto de la demanda la cubrimos con el prefijo de direcciones IP públicas disponible: 190.20.30.0/23
Por lo tanto, completamos el plan de numeración de la siguiente forma:
LAN 2 - 192.20.31.0/28 (máscara 255.255.255.240) - Asignamos 192.20.31.1 a eth2 de R1
LAN 3 - 192.20.30.0/24 (máscara 255.255.255.0) - Asignamos 192.20.30.254 a eth3.1 de R1 (interfaz virtual en eth3)
LAN 4 - 192.20.31.16/29 (máscara 255.255.255.248) - Asignamos 192.20.31.17 a eth3.2 de R1 (interfaz virtual en eth3)
WAN - 200.12.200.13/30 del lado de R1 y 200.12.200.14/30 del lado de R2 (máscara 255.255.255.252)

c) Tablas de forwarding
i) R1

Prefijo	Next Hop	Interfaz
200.12.200.12/30	DC	eth4
190.20.30.0/24	DC	eth3.1
192.20.31.0/28	DC	eth2
192.20.31.16/29	DC	eth3.2
172.31.0.0/23	DC	eth1
0/0	200.12.200.14	eth4

ii) Dispositivo en LAN1

Prefijo	Next Hop	Interfaz
172.31.0.0/23	DC	eth0
192.20.30.0/24	172.31.0.1	eth0

Solo se incluye la ruta específica para LAN3, según las restricciones de la letra.
Se podría agregar una ruta para LAN2 (192.20.31.0/28) para poder acceder al DNS y otros servicios locales.

iii) Servidor en LAN2

Prefijo	Next Hop	Interfaz
192.20.31.0/28	DC	eth0
0/0	192.20.31.1	eth0

iv) Estación de trabajo en LAN3

Prefijo	Next Hop	Interfaz
192.20.30.0/24	DC	eth0
172.31.0.0/23	192.20.30.254	eth0
192.20.31.0/28	192.20.30.254	eth0
192.20.31.16/29	192.20.30.254	eth0

Solo se incluyen las rutas específicas (en particular, NO acceso a internet), según las restricciones de la letra.

v) Estación de trabajo en LAN4

Prefijo	Next Hop	Interfaz
192.20.31.16/29	DC	eth0
0/0	192.20.31.17	eth0

vi) R2 – sólo lo que refiere a la organización Aiouti

Prefijo	Next Hop	Interfaz
192.20.30.0/23	200.12.200.13	eth4
200.12.200.12/30	DC	eth4

d) En et6lan3.aiouti.uy se ejecuta ping et1lan4.aiouti.uy.

Las tramas que se observan en SW1 son:

ARP request para conocer la MAC asociada a la dirección IP 192.20.30.1 – en LAN3

Justificación: se debe conocer la dirección MAC del DG de et6lan3.aiouti.uy pues el destino, de acuerdo a la tabla de forwarding, se alcanza a través de él.

Encabezado capa 2

MAC origen: MAC_eth_et6lan3

MAC destino:FF:FF:FF:FF:FF:FF

Carga útil

MAC origen: MAC_eth_et6lan3

MAC destino: 00:00:00:00:00:00

IP origen: 192.20.30.6

IP destino: 192.20.30.254

ARP reply con la información solicitada – en LAN3

Justificación: retorna la información solicitada

Encabezado capa 2

MAC origen: MAC_eth3.1_R1

MAC destino:MAC_eth_et6lan3

Carga útil

MAC origen: MAC_eth3.1_R1

MAC destino: MAC_eth_et6lan3

IP origen: 192.20.30.254

IP destino: 192.20.30.6

Query de DNS para identificar la dirección IP correspondiente al nombre et1lan4.aiouti.uy. – en LAN3

Justificación: dado que el comando ping recibe como parámetro el nombre del destino, se debe traducir el nombre a una dirección IP, por lo tanto antes de enviar los mensajes ICMP tipo echo request se debe realizar una consulta al servidor DNS autoritativo del dominio aiouti.uy.

Encabezado capa 2

MAC origen: MAC_eth_et6lan3

MAC destino: MAC_eth3.1_R1

Encabezado capa 3

IP origen: 192.20.30.6

IP destino: 192.20.31.2 (asumo que esta IP es la asignada al servidor de DNS)

Encabezado capa 4

Mensaje UDP

Puerto origen: 54321 (algún puerto alto, no bien conocido, asignado por el SO)

Puerto destino: 53

Carga útil

Mensaje DNS del tipo Query, conteniendo 1 consulta: “¿Dirección IP asociada al nombre et8lan4.aiouti.uy?”. Por lo tanto, solicita el valor de un registro tipo A asociado al nombre “et8lan4.aiouti.uy”.

Reply de DNS con la información solicitada. – en LAN3

Justificación: retorna la información solicitada

Encabezado capa 2

MAC origen: MAC_eth3.1_R1

MAC destino: MAC_eth_et6lan3

Encabezado capa 3

IP origen: 192.20.31.2

IP destino: 192.20.30.6

Encabezado capa 4

Mensaje UDP

Puerto origen: 53

Puerto destino: 54321

Carga útil

Mensaje DNS del tipo Query Reply, conteniendo la respuesta a “¿Dirección IP asociada al nombre et8lan4.aiouti.uy?”. Información a partir de lo cual se deduce que el registro de DNS es el siguiente: et8lan4.aiouti.uy IN A 192.20.31.18

ICMP echo request de et6lan3.aiouti.uy a et1lan4.aiouti.uy., en LAN3. – en LAN3

Justificación: ahora sí se dispone de la información necesaria para enviar el mensaje ICMP de tipo echo request.

Encabezado capa 2

MAC origen: MAC_eth_et6lan3

MAC destino: MAC_eth3.1_R1

Encabezado capa 3

IP origen: 192.20.30.6

IP destino: 192.20.31.18 (la obtenida mediante la consulta DNS)

Carga útil

Mensaje ICMP

Tipo Echo (Request) (tipo 8)

ARP request para conocer la MAC asociada a la dirección IP 192.20.31.18 – en LAN4

Justificación: se debe conocer la dirección MAC del DG de et6lan3.aiouti.uy pues el destino, de acuerdo a la tabla de forwarding, se alcanza a través de él.

Encabezado capa 2

MAC origen: MAC_eth3.2_R1

MAC destino:FF:FF:FF:FF:FF:FF

Carga útil

MAC origen: MAC_eth3.2_R1

MAC destino: 00:00:00:00:00:00

IP origen: 192.20.31.17

IP destino: 192.20.31.18

ARP reply con la información solicitada – en LAN4

Justificación: retorna la información solicitada

Encabezado capa 2

MAC origen: MAC_et1lan4

MAC destino:MAC_eth3.2_R1

Carga útil

MAC origen: MAC_et1LAN4

MAC destino: MAC_eth3.2_R1

IP origen: 192.20.31.18

IP destino: 192.20.31.17

ICMP echo request de et6lan3.aiouti.uy a et1lan4.aiouti.uy. – en LAN4

Justificación: ahora sí se dispone de la info necesaria para enviar el mensaje ICMP de tipo echo request en LAN4

Encabezado capa 2

MAC origen: MAC_eth3.2_R1

MAC destino: MAC_et1lan4

Encabezado capa 3

IP origen: 192.20.30.6

IP destino: 192.20.31.18

Carga útil

Mensaje ICMP

Tipo Echo (Request) (tipo 8)

ICMP echo reply de et1lan4.aiouti.uy a et6lan3.aiouti.uy. – en LAN4

Justificación: se recibe el mensaje IMCP de tipo echo (reply) motivado por el “echo request” anterior (en LAN4).

Encabezado capa 2

MAC origen: MAC_et1lan4

MAC destino: MAC_eth3.2_R1

Encabezado capa 3

IP origen: 192.20.31.18

IP destino: 192.20.30.6

Carga útil

Mensaje ICMP

Tipo Echo Reply (tipo 0)

ICMP echo reply de et1lan4.aiouti.uy a et6lan3.aiouti.uy. – en LAN3

Justificación: se recibe el mensaje IMCP de tipo echo (reply), en LAN3.

Encabezado capa 2

MAC origen: MAC_eth3.1_R1

MAC destino: MAC_eth_et6lan3

Encabezado capa 3

IP origen: 192.20.31.18

IP destino: 192.20.30.6

Carga útil

Mensaje ICMP

Tipo Echo Reply (tipo 0)

Se repiten las parejas de mensajes “echo request - echo reply”, en LAN3 y LAN4, hasta completar la cantidad enviada por defecto según la implementación del comando ping del SO de et6lan3.aiouti.uy.

Justificación: se observa el mismo comportamiento que en la pareja inicial de mensajes ICMP.

Problema 2 (25 puntos)

Se desea implementar un servidor proxy HTTP1.1, con una funcionalidad de reemplazo de texto. Este servidor aceptará conexiones en el puerto 80. Los clientes solo realizarán acciones GET.

Cuando el objeto HTTP accedido solicitado sea del tipo "text/plain", el proxy reemplazará todas las ocurrencias de la cadena "qu" por "k". De este modo, si un documento de texto contiene la cadena "queue", esté será servido como "keue".

Se pide:
Implemente el servidor en un lenguaje de alto nivel usando las primitivas de Sockets de la cartilla del curso. Dispone de funciones auxiliares para estructuras de datos, manipulación de strings, etc.

```
MY_IP = "*"

function proxy_thread(cli)
   master = socket.tcp()
   srv = socket.tcp()  // socket hacia el servidor, a conectar cuando sepamos

   // conexión persistente
```

```
while true do

    // leemos el get
    request = ""
    repeat
          fragment, err = cli.read()
          request += fragment
    until request.sub(-4)=="\r\n\r\n" or err="closed"
    if err=="closed" then
          srv.close()
          return
    end

    // la primera vez conectamos al servidor
    if not srv then
          host = request.match("Host: (.*)\r\n")
          srv, err = srv.connect(host, 80)
          if err=="failure" then
              cli.close()
              return
          end
    end

    // enviamos el get
    repeat
          request, err = srv.send(request)
    until request == "" or err=="closed"
    if err=="closed" then
          cli.close()
          return
    end

    // leemos respuesta hasta tener el cabezal
    response = ""
    repeat
          fragment, err = srv.read()
          response += fragment
    until response.pos("\r\n\r\n")>-1 or err="closed"
    if err=="closed" then
          cli.close()
          return
    end

    header = response.sub(1, response.pos("\r\n\r\n"))
    body = response.sub(response.pos("\r\n\r\n")+4, respone.len())

    length = header.match("Content-Length: (.*)\r\n")
    mimetype = header.match("Content-Type: (.*)\r\n")

    // leemos el resto del body
    while body.len()<length or err=="closed"
          fragment, err = srv.read()
          body += fragment
    do
    if err=="closed" then
          cli.close()
          return
    end

    // reempalazxamos en los docs. de texto
    if mimetype="text/plain" then
          body.replace("qu", "k")
          // actualizamos cabezal por si cambiamos el body
          header.replace("Content-Length: %d+\r\n", "Content-Length: "+body.len()+"\r\n")
    end

    // transmitimos todo
    remain = header+body
    repeat
```

```
            remain, err = srv.send(remain)
        until remain == "" or err=="closed"
        if err=="closed" then
            cli.close()
            return
        end
    end
end

master = socket.tcp()
master.bind(MY_IP, 80)
server = master.listen()
while true do
    client, err = server.accept()
    thread.new(proxy_thread, client)
end
server.close() // inalcanzable, para cuando se de de baja el servidor
```