Redes de Computadoras
Letra y Soluciones – 25 de febrero de 2022

Pregunta 1 (8 puntos)
Considere el formato de los datagramas IPv4.
a) ¿Por qué es necesario contar con un campo de *Longitud de la cabecera* ***(header length)***?
b) ¿Qué función cumple y por qué es necesario el campo *Tiempo de vida* ***(ttl – time to live)***?

Solución 1

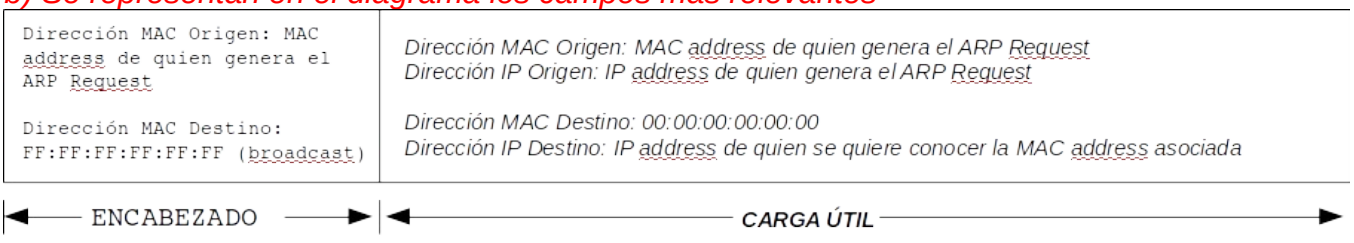
a) Puesto que un datagrama IPv4 puede contener un número variable de opciones (las cuales se incluyen en la cabecera del datagrama IPv4), estos 4 bits son necesarios para determinar dónde comienzan realmente la carga útil (por ejemplo, el segmento de la capa de transporte encapsulado en este datagrama) del datagrama IP.

b) El campo Tiempo de vida (TTL, Time-To-Live) se incluye con el fin de garantizar que los datagramas no estarán eternamente en circulación a través de la red (debido, por ejemplo, a un bucle de enrutamiento de larga duración). Este campo se decrementa en una unidad cada vez que un router procesa un datagrama. Si el campo TTL alcanza el valor 0, el datagrama tiene que ser descartado por el router.

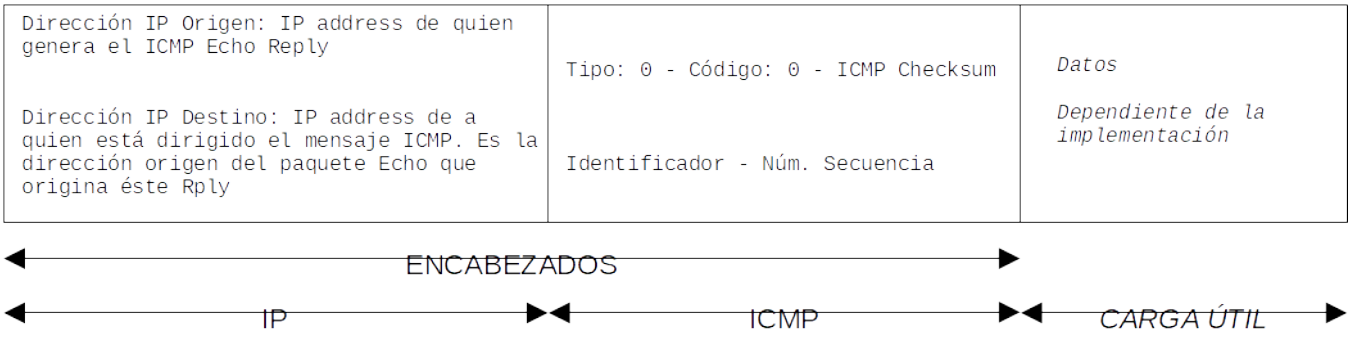
Pregunta 2 (10 puntos)
a) ¿En qué capa se transporta el protocolo ARP?
b) Represente en un diagrama los encabezados (de ARP y la capa que lo transporta) y la carga útil de un mensaje `ARP Request`.
c) ¿En qué capa se transporta el protocolo ICMP?
d) Represente en un diagrama los encabezados (de ICMP y la capa que lo transporta) y la carga útil de un mensaje `ICMP Echo Reply`.

Solución 2

a) Capa de Enlace de Datos
b) Se representan en el diagrama los campos más relevantes



c) Capa de Red
d) Se representan en el diagrama los campos más relevantes (no se incluye el encabezado de capa 2)



Pregunta 3 (12 puntos)

a) Describa los elementos que definen una conexión TCP y en base a esto explique las consideraciones que realiza el receptor para aceptar un segmento.

b) Analice la siguiente afirmación: ***"dada una conexión TCP establecida, si un tercero pretende lograr la caída de la conexión, es suficiente con el envío de un segmento Reset de TCP (un segmento con la bandera RST en 1) a uno de los pares de la conexión, sin otras consideraciones adicionales"***. Justifique detalladamente.

Solución 3

a) Una conexión TCP está definida por la 4-tupla IP origen, puerto origen, IP destino, puerto destino. Además de esto, en una conexión TCP se acuerda un número de secuencia para cada uno de los pares. Un segmento es aceptado por un receptor cuando pertenece a una conexión establecida, es decir la 4-tupla corresponde a una conexión establecida y además el número de secuencia es válido (es decir, está dentro de la ventana de recepción en cuestión)

b) La afirmación no es correcta. Para que el segmento en cuestión logre su efecto, lo primero es que sea aceptado por el destinatario. Si asumimos que el tercero se encuentra enviando el segmento desde una computadora que, en el caso más general, está conectada a cualquier red, a una distancia en saltos y/o ASs, a priori desconocida entonces deberá considerar lo siguiente:

- Identificación de la conexión. Se debe disponer de toda la información que identifica unívocamente la conexión TCP, a saber: IP_Origen, IP_Destino, Puerto_Origen y Puerto Destino.

- Número de Secuencia incluido en el segmento RST. El número de secuencia del segmento RST enviado debe ser válido para el receptor, o sea que debe estar dentro de la ventana de recepción en cuestión.

- Routing/forwarding al destino. En principio, nada asegura que el paquete IP que se origine desde el tercero, con IP origen falsificada efectivamente alcance al otro peer. Aquí observamos consideraciones a contemplar vinculadas al "forwarding/routing" en el camino entre el tercero y el destino.

De lograrse todo lo anterior, y dado que la especificación original de TCP indica que de ello ocurrir, ningún otro segmento debe ser procesado y se debe proceder al cierre inmediato de la conexión TCP, se habría logrado el objetivo del tercero.

Como complemento a la respuesta, pero aclarando que no es parte de lo exigido pues escapa a los objetivos del curso, resulta de orden comentar que una gran cantidad de los problemas

de seguridad que se sufren, se sustentan en decisiones de diseño tomadas en otros tiempos y para otras realidades. El buen uso de la flag RST es una gran herramienta al momento del diseño, desarrollo y del troubleshooting, pero la vulnerabilidad intrínseca puesta de manifiesto en este ejemplo, ha hecho que la comunidad haya trabajado en diversos paliativos que deberían complementar las consideraciones mencionadas a la hora de realizar un análisis o investigación más profunda y profesional. Para quienes estén interesados, la RFC 5961 es un buen punto de partido ahondar en ello.

Pregunta 4 (10 puntos)

A diferencia de IPv4, donde cada interfaz normalmente posee una única dirección de red, IPv6 está diseñado para tener tantas direcciones por interfaz como sea necesario.

a) Nombre las direcciones de red que debe implementar cada interfaz (sin ser la de loopback) de un host en IPv6

b) Nombre las direcciones de red que debe implementar cada interfaz (sin ser la de loopback) de un router en IPv6

c) Explique la utilidad del uso del EUI-64 y como es utilizado en IPv6

Solución 4

a) En IPv6 cada host debe implementar las siguientes:

- *link local*
- *all-nodes(multicast)*
- *solicited-node*

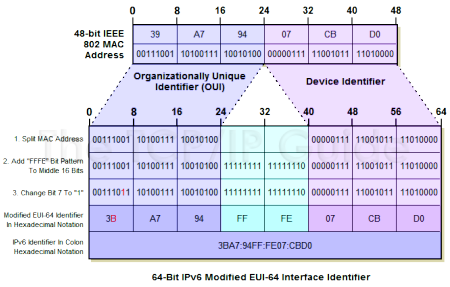
b)En IPv6 cada router debe implementar las siguientes:

- *todas las de un host*
- *anycast subnet-router*
- *multicast all routers*

c) Para el protocolo SLAAC la dirección link-local se genera de acuerdo a:

[fe80 (10 bits) + 0 (54 bits)] + interface ID (64 bits)

con el uso del EUI-64 se calcula el intefaceID, donde fundamentalmente se coloca ff:fe “al medio” de los 48 bits de la dirección mac



64-Bit IPv6 Modified EUI-64 Interface Identifier

Problema 1 (35 puntos)

Se desea implementar un sistema de domótica o “casa inteligente” donde se cuenta con una serie de sensores de monitoreo y diversos actuadores (aires acondicionados, puertas y ventanas automáticas, etc.) distribuidos y conectados a una red IP privada. Se cuenta además con un controlador centralizado que recibe la información de monitoreo, ejecuta un algoritmo de control (por ejemplo para mantener la temperatura de la casa en un cierto nivel) y envía acciones de control a los distintos dispositivos.

Los sensores de monitoreo envían la información al controlador mediante un mensaje UDP con el formato:

```
<id_sensor>:<timestamp>:<nombre_parametro>:<valor>
```

El controlador espera estos mensajes en una IP y puertos conocidos (10.0.0.1, 8888).

Por cada mensaje que recibe, el controlador almacena la información recibida así como la dirección IP del sensor que la envió utilizando la función `saveNewSensorData(id_sensor, ip_sensor, timestamp, nombre_parametro, valor)`.

Cada cierto tiempo, el controlador ejecuta su algoritmo de control y puede generar comandos a los actuadores para ajustar su configuración. Para esto se conecta a cada actuador utilizando el protocolo TCP por donde envía el mensaje de control (de largo fijo 8 bytes) con el formato:

```
<nombre_parametro>:<valor>
```

y espera la respuesta:

`OK`
si el comando se pudo ejecutar correctamente o

`ERROR`
si hubo algún tipo de error.

(El actuador cierra la conexión luego de enviar su respuesta)

Para realizar esta tarea el controlador utiliza la función `int sendCommand (IP_actuador, nombre_parametro, valor)` que retorna 1 si hubo error y 0 en caso contrario.

Se pide:

Implemente en un lenguaje de alto nivel usando la API de sockets del curso:

a) El procedimiento que ejecuta en el controlador para recibir la información de los sensores de monitoreo.
b) La función `sendCommand` que utiliza el controlador para enviar los comandos a los actuadores.
c) El procedimiento que ejecuta en cada actuador para recibir los comandos del controlador. Dentro de cada actuador el comando recibido se ejecuta con la función `bool executeCommand (nombre_parametro, valor)` que devuelve `false` si hubo error y `true` en caso contrario.

Solución 1

a)

```
def serverControlador()
  #creo socket y lo asocio con la interfaz y puertos dados
  skt = socket.udp()
  skt.bind('10.0.0.1', 8888)

  while true do
    #espero por mensaje de sensores
    datagram, ip_sensor, port_sensor = skt.receive()

    #obtengo los valores dentro del mensaje
    data = datagram.split(':')

    saveNewSensorData(data[0], ip_sensor, data[1], data[2], data[3])
```

```
  end
end

b)
int sendCommand(IP_actuador, parametro, valor)
  master = socket.tcp()
  client = master.connect(IP_actuador, 1234)

  comando = parametro + ":" + valor
  repeat
    comando, err = client.send(act)
  until comando=='' or err=='closed'

  if err=='closed':
    exit()
  end

  buff = ''
  repeat
    r, err = client.receive()
    if not err then
      buff = buff + r
    end
  until err=='closed'

  if buff == 'OK'
    return 0
  else
    return 1
  end
end

c)
def serverActuador()
  server = socket.tcp()
  server.bind(*, 1234)
  server = server.listen()
  while true:
    client, err = server.accept()
    new.thread(ejecutarComando, client)
  end
  server.close()
end

def ejecutarComando (client)
    msg = ''
    repeat
      r, err = client.receive()
      msg=msg + r
    until msg.size()==8

    nombre_parametro, valor = m.split(':')
    result = executeCommand (nombre_parametro, valor)
```

```
    if result:
      msg = 'OK'
    else
      msg = 'ERROR'
    end

    repeat
      msg, err = cliente.send(msg)
    until msg=='' or err=='closed'

    client.close()
end
```

Problema 2 (25 puntos)

Se considera una topología de interconexión de Sistemas Autónomos como se muestra en la Figura 1 donde:

- T1: proveedor `Tier 1`
- A, B, C: clientes de T1
- C1, C3: clientes de B
- C2, C4: clientes de C
- C5: cliente *dual-homed* de B y C
-
- L1….L9: relaciones de cliente-proveedor, según se describió anteriormente
-
- P2P1 y P2P2: relaciones entre pares, que solo permiten tráfico entre los pares y sus clientes

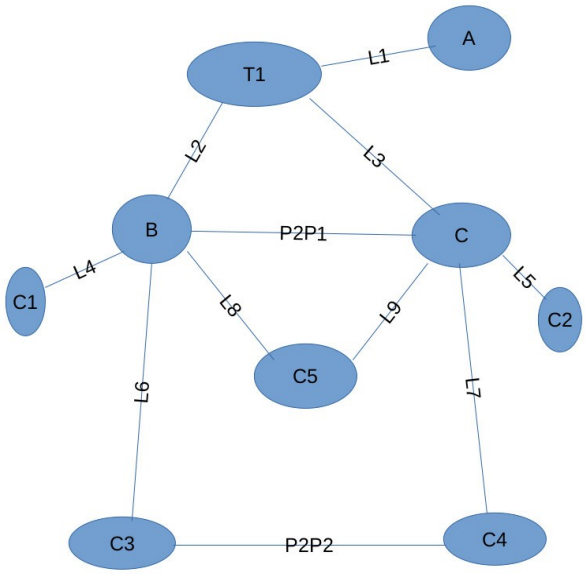


Fig 1: Topología de Sistemas Autónomos

Las políticas que se utilizan son las habituales de cliente-proveedor, con la excepción de las relaciones de pares descriptas anteriormente.

Además, el Sistema Autónomo C5 tiene la topología descrita en la Figura 2, y la configuración de BGP no utiliza el atributo `Local Preference`. Todos los routers ejecutan BGP (externo o interno, según corresponda).

Se pide:

a) Para los siguientes pares de ASes, escriba los `AS-PATH` válidos, y justifique su respuesta en base a las políticas definidas:

1) desde A a C5
2) desde C1 a C2
3) desde C3 a C4

b) Dados los costos del *IGP* que están marcados en la Figura 2, ¿qué camino seguirá el tráfico desde R1 (dentro del AS C5) a los prefijos del AS A? Justifique.

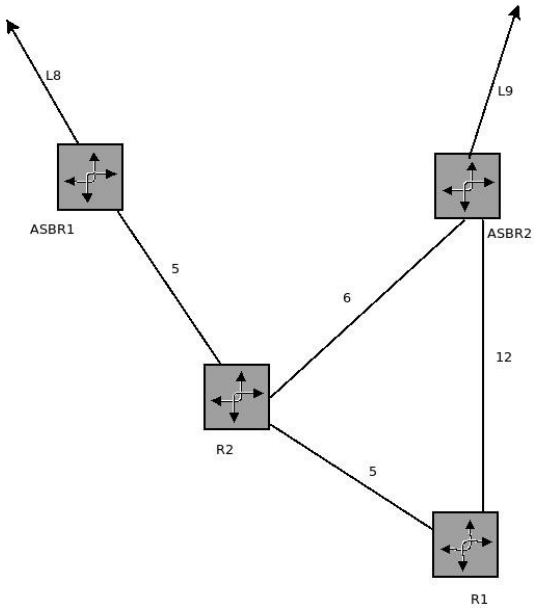


Fig 2: Topología interna de C5

c) Considere su respuesta anterior, ¿el camino del tráfico desde A hacia R1 es el mismo? Justifique.

Solución 2

Parte a)
1) Caminos posibles A->C5

A → T1 → B → C5
A → T1 → C → C5

Siempre se recorren caminos cliente – proveedor
NO son correctos los caminos que utilizan el enlace P2P1, porque la relación de pares no admite tráfico externo, ni tampoco los caminos que utilizan el enlace P2P2, por el mismo motivo, y porque además se recorrerían ASes "stub" para llegar a otro "stub", subiendo y bajando de proveedores, violando el principio de enrutamiento de que todo tráfico entrante a un "stub" debe ser destinado a esa red y de que todo tráfico saliente de un "stub" debe ser tráfico originado en esa red.

2) Caminos posibles C1->C2

C1 → B → C → C2
C1 → B → T1 → C → C2

No son posibles los caminos que usan a C3, C4 o C5, por la misma razón explicada en la parte 1)

3) Caminos posibles C3->C4

C3 → C4
C3 → B → C → C4
C3 → B → T1 → C → C4

No es posible el camino que usa a C5, por la misma razón explicada en la parte 1)

Parte b)

1) Los dos caminos posibles desde C5 hacia A son C5 → B → T1 → A y C5 → C → T1 → A, ambos de igual longitud de AS_PATH. Por lo tanto hay que usar un criterio de desempate, y dado que no está configurado Local Preference, se utiliza la distancia del IGP. EN definitiva, el camino del tráfico desde R1 hacia A es R1 → R2 → ASBR1 hasta el borde del AS C5, y luego sigue el camino C5 → B → T1 → A.

2) El camino "de vuelta" no tiene por qué ser el mismo, ya que se desconocen las políticas de A, y por lo tanto podría ser cualquiera de las dos posibilidades:

A → T1 → B → C5: el tráfico entra por ASBR1
A → T1 → C → C5: el tráfico entra por ASBR2