Redes de Computadoras, Introducción a las Redes de Computador{ae}s y Comunicación de Datos

Solución – 17 de diciembre 2015

(ref: solredes20151217.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.**
- **Sólo 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 (10 puntos)

Sea el protocolo FTP.

a) Describa brevemente el protocolo, su utilidad y funcionamiento mencionando ¿cómo se utilizan el canal de control y el de datos por parte del protocolo?
b) Mencione al menos dos diferencias con HTTP, particularmente, cuál es un protocolo con señalización *in-band* y cuál *out-of-band*.

Solución:

a) El protocolo de transferencia de archivos (File Transfer Protocol) sirve para la transferencia de archivos entre hosts sobre una red IP utilizando el protocolo de transporte TCP. Sigue el paradigma cliente-servidor y utiliza conexiones de control y datos independientes entre el cliente y el servidor. El funcionamiento del protocolo se basa en un establecimiento de conexión y autenticación, seguido de una ejecución interactiva de comandos locales o remotos y posterior desconexión. Se pueden enviar y recibir archivos en una misma sesión FTP
Cuando el cliente se conecta al servidor, establece una conexión TCP al puerto 21 del servidor, que es utilizada para autenticación, envío de mensajes y recepción de códigos de salida. Cuando el servidor transfiere información al cliente, desde la salida de un comando ls (o dir) hasta la transferencia específica de un archivo, utiliza una segunda conexión, denominada de datos. El comportamiento original del protocolo involucraba que la conexión de datos era iniciada por el servidor y atendida por el cliente. Normalmente se usa el puerto 20 de TCP para la transferencia de datos.

b)

- HTTP utiliza el mismo canal para control y datos, por lo que es del tipo in-band. FTP utiliza canales separados para datos y control, por lo que es del tipo out-of-band.
- El protocolo HTTP es stateless (sin estado), mientras que FTP es statefull (o con estado)
- FTP ofrece servicios de adaptación de datos (ASCII, EBCDIC o Binario) mientras que HTTP solamente realiza transferencias binarias.
- Para las transferencias de datos, FTP puede funcionar en modos activo y pasivo.

Pregunta 2 (9 puntos)

a) Describa el mecanismo de *self-learning* de los *switches* (conmutadores de capa 2).
b) Cuando llega una trama a un *switch* cuya dirección de destino de capa 2 es la dirección de *broadcast*, la misma se reenvía por todos los puertos. ¿Es posible que una trama cuya dirección de destino no sea la de *broadcast* también se propague por todos los puertos del *switch*?

Solución:

a) Para su funcionamiento básico, los switches no requieren configuración, pues, son capaces de aprender lo requerido para operar. Cuando un mensaje llega a una boca de un switch, el switch puede inspecionarlo y en base a la dirección origen del mismo, es capaz de aprender que a dicha boca está conectado un host con esa dirección MAC específica. A medida que el tiempo pasa y diferentes hosts envían tramas, el switch aprende a que bocas están conectados cuales hosts. Con esa información, el switch es capaz de reenviar tramas solamente a las bocas donde están conectados los hosts. Los mapeos de MACs a puertos tienen tiempos de vida acotados y deben ser constantemente refrescados a efectos de permitir que los hosts puedan migrar entre bocas del switch. Siempre que se desconoce una asociación y se debe hacer llegar una trama a un host particular, la trama se copia a todas las bocas.
b) Si, es posible. Cuando se desconoce el mapeo, según se vio en a), una trama se copia a todas las bocas. En casos de protocolos como IPv6, cuando se transmite al grupo de multicast all-hosts (ff02::1), la trama se copia en todas las bocas asociadas al segmento.

Pregunta 3 (7 puntos)

Nombre el protocolo utilizado en IPv6 para asociar dinámicamente direcciones de capa de enlace a direcciones IPv6. Describa los mensajes utilizados.

Solución:

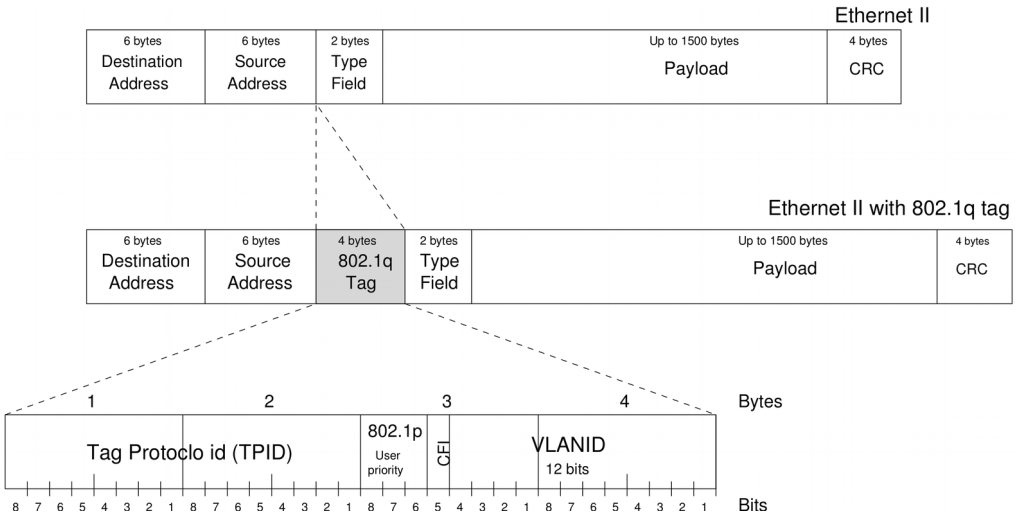
El protocolo que utiliza IPv6 para control en general es ICMPv6. El subprotocolo que se encarga del conocimiento, administración y gestión de vecinos en el enlace es el Neighboor Discovery Protocol, ND. Los mensajes que se utilizan para descubrir y anunciar las direcciones MAC de un vecino asociadas a cierta dirección IPv6 son Neighbor Solicitation y Neighbor Advertisement. Cuando un host desea conocer la direccion MAC asociada a un vecino HOST-A envía un mensaje ND, con destino IPv6 la dirección de multicast asociada a todos los hosts (ff02::1) y con dirección de destino a nivel mac, la asociada a dicho grupo de multicast (que no es broadcast). En el payload de ICMPv6, el mensaje ND contiene la dirección objetivo de la consulta. La eventual respuesta (si existe y es alcanzable en ese momento el destino) será enviado en un mensaje de tipo unicast con las direcciones origen y destino esperadas a nivel de cabezales MAC e IPv6. En el payload ICMPv6 viene la respuesta a la consulta donde se indica la dirección MAC solicitada.

Pregunta 4 (7 puntos)

Sea el protocolo de etiquetado VLAN 802.1Q. Describa las modificaciones que introduce a nivel de los cabezales de capa 2 y explique cómo se combina este tráfico en los *switches* con el tráfico *ethernet* original que no utiliza ese etiquetado.

Solución:

El protocolo 802.1Q involucra una redefinición del cabezal Ethernet, que introduce cuatro bytes adicionales luego de las direcciones MAC src y dst:



de los 32bits adicionales, se reservan 12 para identificar VLANs.

A nivel de los switches se distinguen puertos a los que se pueden conectar dispositivos que manipulan los cabezales extendidos y son capaces de intercambiar tráfico que incluye TAGS de los puertos en los que se conectan equipos que desconocen del uso de tags. Estos puertos, los que reciben tráfico sin tags se conocen como UNTAGGED PORTS y realizan la tarea de etiquetar tráfico al ingreso y quitar el tag al momento de forwardear el paquete hacia el host. Los puertos por los que se transmiten tramas extendidas se conocen tomo TRUNK PORTS. Cabe mencionar que la distinción es administrativa, y no hay diferencias físicas en los puertos.

Pregunta 4 IRC (7 puntos)

Describa los mensajes RTS y CTS presentes en los protocolos IEEE 802.11. Explique que problema pretenden resolver, y cómo se utilizan para lograr dicho objetivo.

Solución:

El objetivo es evitar el problema de la terminal oculta, por el mecanismo de reserva el acceso al canal. El transmisor emite un mensaje RTS, y el receptor responde con un CTS en cuanto esté listo para recibir. Cuando el transmisor recibe el CTS inicia la transmisión de datos. Otros nodos que reciban el RTS o el CTS retrasan sus transmisiones.

Pregunta 5 (7 puntos)

Suponga que el host A envía al host B un segmento encapsulado en un datagrama IP. Cuando el host B recibe el datagrama, ¿cómo sabe la capa de red de ese host que debería pasar el segmento a TCP en lugar de a UDP o cualquier otro protocolo?

Solución:

La capa de red le entrega el payload del datagrama a el protocolo indicado por el campo de ocho bits "*protocol*" del encabezado. Los valores de ese campo están estandarizados y cada uno de ellos indica un protocolo de capa superior determinado. Si en payload del datagrama es un segmento TCP, el campo *protocol* tiene el valor 6.

Problemas Prácticos

Problema 1 (30 puntos)

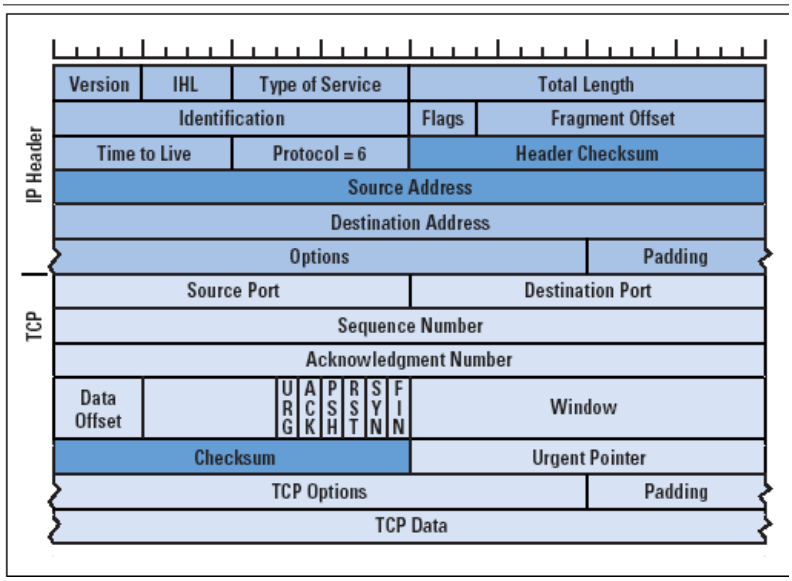
El equipo ***A***, con IP 15.15.15.150 establece una conexión TCP desde su puerto local 5555, al puerto 80 del equipo ***B*** con IP 6.6.6.60.

a) Suponiendo que ***A*** y ***B*** eligen como número inicial de secuencia *1200* y *4800* respectivamente, indique los cabezales intercambiados para el establecimiento de la conexión (*three-way handshake*).
b) Con la conexión de la parte a) ya establecida, ***B*** envía a ***A*** un segmento con la siguiente información en los cabezales TCP, *Sequence Number* = 5000 *Acknowledgement Number* = 2000 que conteniendo 1100 bytes de datos y suponiendo además que ***A*** no tiene bytes para enviar a ***B***.

i. ¿que información contienen los cabezales del segmento de *acknowledgment* enviado por ***A*** a ***B***, para notificar su recepción?
ii. Si ***B*** tiene a continuación 1100 bytes más para enviar a ***A***, ¿qué información contienen los campos de los encabezado del siguiente segmento para su envío?
iii. Suponiendo que se pierde el segmento de *acknowledgment* propuesto en la parte b.i, y el enviado en b.ii llega correctamente, ¿qué número de secuencia esperará ***A*** después de la recepción del segmento enviado en b.ii?

c) Después de enviados y aceptados todos los datos indicados en la parte b), **A** desea cerrar la conexión establecida en la parte a), indique los intercambios de datagramas entre los dos equipos.

Para sus respuestas, use el formato de los cabezales IP y TCP presentados en la figura debajo, pero especificando únicamente los siguientes campos: *Source Address, Destination Address, Source Port, Destination Port, Sequence Number, Acknowledgement Number y flags (ACK, SYN, FIN).*



Solución:

a) Se intercambian entre el cliente y servidor en la conexión 3 segmentos, donde la secuencia es comenzada por el equipo que la establece la conexión (cliente) hacia un equipo que lo espera.

Los intercambios son los siguientes:

Source Address	Destination Address	Source port	Dest. Port	Seq. Number	Ack Number	ACK	SYN	FIN	observaciones
15.15.15.150	6.6.6.60	5555	80	1200	0	0	1	0	Envío SYN del cliente
6.6.6.60	15.15.15.150	80	5555	4800	1201	1	1	0	Devolución SYN ACK, por parte del servidor
15.15.15.150	6.6.6.60	5555	80	1201	4801	1	0	0	Envío ACK, del cliente

Debe tenerse en cuenta que el número de aceptación es incrementado en uno para los ACK realizados.

b)

b.i)
Se incluye en la solución el segmento enviado con los 1100 bytes con dirección B → A , para aclarar la solución pedida.

Source Address	Destination Address	Source port	Dest. Port	Seq. Number	Ack Number	ACK	SYN	FIN	observaciones
6.6.6.60	15.15.15.150	80	5555	5000	2000	0	0	0	Envío indicado en la letra, con 1100 bytes
15.15.15.150	6.6.6.60	5555	80	2000	6100	1	0	0	ACK enviado para el nro de secuencia 6100. Pedido de bi

b.ii)

Source Address	Destination Address	Source port	Dest. Port	Seq. Number	Ack Number	ACK	SYN	FIN	observaciones
6.6.6.60	15.15.15.150	80	5555	6100	2000	0	0	0	Solución b.ii. Contiene 1100 bytes de datos.

b.iii) Como los segmentos conteniendo datos no se perdieron, (se ha perdido un ACK de A→B), a queda esperando el número de secuencia 7200 = 6100 + 1100.

El equipo A enviará un ACK con valor 7200.

c) El cierre de la conexión es iniciado por A y requiere el intercambio de los siguientes segmentos:

Source Address	Destination Address	Source port	Dest. Port	Seq. Number	Ack Number	ACK	SYN	FIN	observaciones
15.15.15.150	6.6.6.60	5555	80	2000	7200	0	0	1	Inicio de desconexión
6.6.6.60	15.15.15.150	80	5555	7200	2001	1	0	0	Aceptación del FIN
6.6.6.60	15.15.15.150	80	5555	7200	2001	0	0	1	Envío de FIN B->A
15.15.15.150	6.6.6.60	5555	80	2001	7201	1	0	0	Aceptación del FIN

En resumen el *equipo A* envía un FIN y el *equipo B* devuelve un ACK del FIN recibido incrementando el número de secuencia esperado en el ACK. En éste momento el *equipo B* podría seguir enviando datos mientras que el *equipo A* ya no puede, dado que ha enviado el FIN. Como no tiene más información para enviar, Inmediatamente el *equipo B* envía un segmento con FIN, el que el *equipo A* responde con un ACK, quedando finalizado el proceso. Si la aceptación del último FIN, hecha por *equipo A* se perdiera, se espera un tiempo y se da por finalizada la conexión.

Problema 2 (30 puntos)

La empresa Uatsap Inc. está trabajando en el desarrollo del sistema de chat SiempreLlega, el cual posee la novedosa funcionalidad de que si a un miembro no conectado le envían un mensaje, el sistema lo hará llegar a través de un correo electrónico. Implemente este servicio, considerando las siguientes especificaciones: El protocolo que debe implementar trabajará sobre TCP, y cuenta con los siguientes comandos:

CONNECT <usuario_cliente> <contraseña_cliente>: Indica al servidor que un cliente desea iniciar sesión con el nombre de usuario <usuario_cliente> y contraseña <contraseña_cliente>, y es lo primero que hace el usuario al establecer la conexión TCP con el servidor.

CONNECT_RESULT <resultado>: Mensaje de respuesta del servidor a un CONNECT del cliente. Indica el resultado del intento de conexión. Los posibles valores de <resultado> pueden ser: ok, bad_login.

MESSAGE <receptor> <texto>: Envía el mensaje <texto> a <receptor>. Si <receptor> no se encuentra conectado al servicio, y le pidió al servidor recibir los mensajes por correo electrónico en tal caso.
GETUSERSLIST: Utilizado por los clientes para obtener la lista de todos los usuarios del sistema.

USERSLIST <lista_usuarios>: Respuesta del servidor a GETUSERSLIST. Devuelve todos los nicknames de usuarios del sistema, separados por espacios.

Asuma que cuenta con las siguientes operaciones:

get_user_email(usuario: string): string. Devuelve la dirección de correo electrónico de un determinado usuario del sistema.

check_credentials(usuario: string, contraseña: string): boolean. Chequea si los datos de login suministrados corresponden a un usuario registrado.

get_users_list(): string. Devuelve una lista de todos los usuarios registrados en el sistema, separados por espacio.

send_email(destinatario, mensaje: string).

Asuma también que cuenta con primitivas para el manejo de sockets (creación, envío y recepción de mensajes), hilos y funciones de parseo de los comandos descritos anteriormente. El servidor escuchará peticiones de conexión en el puerto PUERTO_SERVIDOR, y tiene dirección IP_SERVIDOR.

Solución:

a)

```
Map<String, ClientSocket> clientes_activos;

void main () {
	server_socket = new Socket();
	server_socket.bind(IP_SERVIDOR, PUERTO_SERVIDOR);

	while (true) {
		client_socket = server_socket.accept();
		newThread(atender_cliente, client_socket);
	}
}

void atender_cliente(client_socket) {
  string nick_cliente;
	while (true) {
		line = client_socket.readLine()
		if line == null {
			clientes_activos.remove(client_socket)
			return
		}
		if isConnect(line) {
			nick_cliente = getUser(line);
			if check_credentials(nick_cliente, getPassword(line)) {
				clientes_activos.add(getUser(line), client_socket)
				client_socket.send("CONNECT_RESULT ok")
			} else {
				client_socket.send("CONNECT_RESULT bad_login")

			}
		} else if isGetUserList(line) {
			users = get_users_list();
			cliente_socket.send("USERSLIST " + get_users_list())
		} else if isMessage(line) {
			destinatario = getDestinatario(line)
			texto = getTexto(line)
			if (clientes_activos[destinatario] != null) {
				clientes_activos[destinatario].send(texto)
			} else {
				send_email(get_user_email(destinatario), texto)
			}
		}
	}
}
```