Redes
Control 2

José M. Piquer - 2 hrs
Con Apuntes - Hojas Separadas

7 de noviembre de 2010

Pregunta 1 (IP, conceptos)

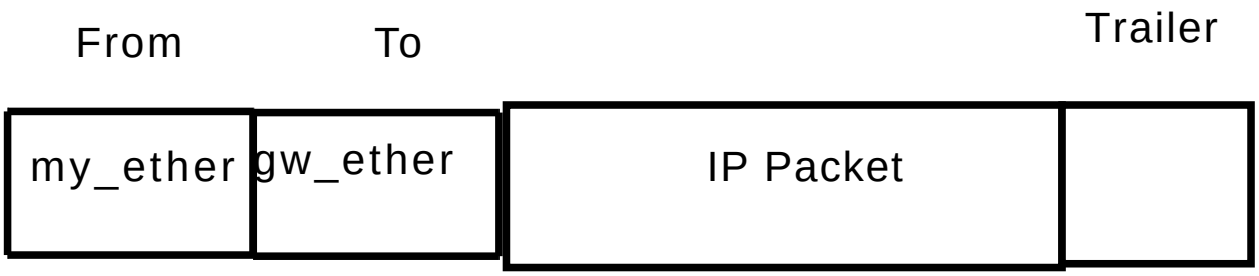
Parte I

En pseudo-código, el ruteo de un paquete IP en un computador termina llamando a `sendphys(dgram, Route.gateway);` quien debe enviar el paquete `dgram` a la dirección IP del gateway.

Explique qué hace esa función en el caso de una red ethernet (lo más detallado posible).

Dibuje el paquete ethernet generado, donde va el datagrama hacia el gateway y explique los diversos campos de direcciones y su contenido.

```
sendphys(dgram, gw) {
    gw_ether = ARPgetaddr(gw); /* bcast de ARP, espera respuesta */
    if(gw_ether == NULL) return NO_HOST; // No existe o no responde
    physpack = ether_encapsulate(dgram, my_ether, gw_ether);
    send_ether(physpack);
}
```



Parte II

La mayoría de las veces que veo un datagrama en Internet, la dirección IP de origen y la de destino no coinciden con las direcciones físicas de origen y destino del frame de red. Explique porqué y cuando ocurre que sí coinciden.

Normalmente, un paquete físico va entre un router y otro, por lo que sus direcciones de origen y destino corresponden a ellos y no a su origen y destino final. En el único momento en que estas direcciones coinciden es en la red de origen (donde la IP de origen corresponde con la dirección física de origen) y en la red de destino final (donde la IP de destino corresponde con la dirección física de destino).

Parte III

Explique por qué se dice que las sub-redes en IPv4 se pudieron implementar sin modificar los protocolos de ruteo de Internet, en cambio las super-redes necesitaron una modificación completa.

Las sub-redes se pueden implementar en la red interna, sin modificar nada fuera de ella. El prefijo de red, aunque se base en la clase IP solamente, igual funciona hasta que uno llega a la entrada a la red. Solamente dentro de ella requiero conocer la máscara de red que utilizaré. En cambio, con super-clases, necesito conocer la máscara de red en todo Internet, puesto que varias redes distintas ahora corresponden a una misma red. Si no entiendo el nuevo protocolo (de IP sin clases) tendremos que difundir todas las redes sin agruparlas, como si fueran redes disjuntas.

Pregunta 2 (ICMP, ruteo, fragmentacion)

Parte I

Un ingeniero argumenta que la fragmentación no aumenta la probabilidad de pérdida de los datagramas, puesto que la cantidad de bytes que se envían es la misma de antes y la probabilidad de pérdida se distribuye en forma uniforme. Discuta técnicamente esta afirmación.

La probabilidad de pérdida en Internet se distribuye por paquete, no por byte. Por eso, aunque la distribución fuera uniforme, al enviar 5 paquetes en vez de uno, tengo 5 veces más probabilidad de perder uno, lo que genera la pérdida total del paquete.

Parte II

Un ingeniero argumenta que IPv6 fue un error, porque el header quedó muy grande y se eliminó el checksum del header al pensarse que las redes del futuro eran de fibra óptica (mucho ancho de banda y pocos errores) y hoy en día son inalámbricas (poco ancho de banda y muchos errores). Discuta esa afirmación.

La eliminación del checksum del header no es ningún problema, porque aunque las redes fueran muy ruidosas, el destino final verifica el checksum de transporte (que ahora es obligatorio) y descarta los errores. La ganancia en eficiencia en los routers, al no tener que recalcular el checksum cada vez, justifica de todo

modos el costo. El tamaño del header sí es un problema y hay que intentar usar paquetes de datos grandes para que justifiquen el overhead de IPv6. Se espera que las redes inalámbricas vayan siendo de cada vez mayor capacidad, para que esto no sea un problema a futuro.

Parte III

En el curso hemos dicho que todo computador rutea, incluso el notebook conectado por una sola interfaz de red. Explique por qué y cómo funciona.

Todo computador debe conocer los prefijos IP de las redes a las que está físicamente conectado (incluyendo el loopback). Y debe tener al menos una ruta default apuntando al router de su red local. Por lo tanto, su tabla de rutas tiene al menos 3 entradas típicamente. Sin embargo, si hay más de un router en la red local, debo tener una entrada por cada red que se rutea por routers distintos al default. Esto permite que los paquetes vayan directamente a la red local a la que corresponden, o al router que les corresponde, sin pasar dos veces por la misma red.

Pregunta 3 (TCP)

Parte I

Explique lo que ocurre en un protocolo de Stop-and-Wait si el timeout del transmisor es muy grande o si es muy chico. ¿Cual es el valor óptimo del timeout si todos los parámetros de la red fueran conocidos y constantes en el tiempo?

Cuando el timeout es muy grande, al haber una pérdida, retraso mucho la retransmisión, haciendo que la transmisión sea tremendamente lenta. Cuando el timeout es muy pequeño, causamos retransmisiones innecesarias, obligando a enviar ACK innecesarios, saturando la red. El valor óptimo de un timeout es simplemente el Round-Trip-Time máximo + 1.

Parte II

En un protocolo Go-Back-N, vimos que el transmisor puede asumir que un ACK para la secuencia N implica un ACK para todas las secuencias previas a N. Frente a desorden de los paquetes, ¿esto también es válido? Justifique.

La idea de esto es que, al recibir un ACK para N, puedo suponer que el enviador de ese ACK ya envió todos los ACKs anteriores, aunque yo no los haya recibido. Por protocolo, en efecto, al tener una ventana de recepción de tamaño 1, jamás puedo enviar ACK para N si me falta algún paquete previo a ese. El protocolo indica siempre responder a un paquete de datos con el ACK del último paquete recibido correcto y en orden. En caso de desorden de paquetes, esto sigue siendo cierto: un paquete que llega antes que uno anterior será descartado (y se enviará el ACK para el último paquete recibido en orden), y un ACK que llega antes que uno anterior será aceptado bien, y sigue implicando que todos los ACKs anteriores fueron enviados. Cuando me lleguen ACKs antiguos, igual que siempre, los descartaré.

Parte III

En una red con muchas pérdidas, el algoritmo de Karn genera timeouts muy grandes. Proponga una mejora al protocolo de TCP que permita mantener timeouts precisos y que no requiera multiplicar los timeouts cada vez que haya retransmisiones.

Una propuesta sería agregar al header TCP un contador de repeticiones de este segmento. La primera vez que lo enviamos iría con un contador 0, y luego se va incrementando al ser re-transmitido. Los ACKs tendrían que venir con el número de secuencia y el contador respectivo. De esta forma, al re-transmitir un segmento, asocio una lista "hora, contadorçon todas las veces que lo he enviado. Al recibir un ACK, puedo obtener la hora de envío asociada al segmento y al contador, y calcular el RTT en forma precisa, lo que evita multiplicarlo por dos cada vez que hay retransmisiones.