Indice

Contenidos Página

INTRODUCCIÓN

Cuando se habla de aplicaciones de redes, sin duda alguna que el modelo Padre a considerar es el OSI de la ISO (Internacional Standard Organitation). Este modelo es una buena base para comprender los servicios de cada capa que necesita cumplir un protocolo para su implementación. Si bien, el modelo OSI consta de 7 capas como ya hemos estudiado en clases, en cada una de estas capas se implementan distintos servicios que se sostienen el uno al otro, es decir, que necesita implementar primero en forma óptima el de nivel inferior para que el siguiente (ascendentemente) pueda llevar a cabo sus servicios y es aquí precisamente donde nos

detendremos para describir y comprender como se llevan a cabo tareas tan fundamentales como:

- *La detección y corrección de errores.*
- *Los algorítmos de encutamiento.*
- *La compresión de datos.*
- *El encriptamiento de la información.*
- *El correo electrónico y,*
- *Los terminales virtuales.*

Si bien son temas y conceptos que tienen cierta complejidad, en el presente informe se explican de la forma más simple y precisa, aunque si se debe decir que el lector debe tener algun tipo de conocimento con respecto a los protocolos.

Sólo resta invitar a continuar con las siguientes páginas, las cuales son absolutamente de competencia para el mundo de la Informatica.

DETECCION Y CORRECCION DE ERRORES

Sin duda que a cuanto mayor es la trama que se transmite, mayor es la probabilidad de que contenga algún error.

Los errores de transmisión en las líneas telefónicas son provocados por varios fenómenos físicos. Un fenómeno que siempre está presente es le ruido térmico. Los electrones, en los hilos de cobre, están moviéndose a muy alta velocidad y en todas las direcciones, produciendo un amplio espectro de nivel de ruido de fondo. Esta es la relación de señal a ruido.

Otra fuente de errores de suma importancia es el hecho de que la amplitud, velocidad de propagación y fase de las señales, dependen todos de la frecuencia el sistema telefónico, en efecto, descompone en una serie de Fourier todas las señales, distorsiona cada componente de frecuencia en forma separada y después las recombina al final, es posible alquilar líneas acondicionadas e especial sobre las cuales los servicios portadores tratan de minimizar estos efectos, pero tales líneas son más costosas que las que se utilizan por lo regular u, además, nunca llegan a tener una ecualización perfecta.

Existen otras muchas fuentes de errores comunes: el cruce de señales entre líneas, por ejemplo, puede ocurrir entre dos hilos físicamente adyacentes. Cuando los eliminadores de eco se apagan se presentan ecos. Los enlaces por microondas están sometidos a desvanecimientos de propagación: pájaros, y cosas por el estilo. Para realizar una transmisión de voz, es deseable comprimir la amplitud de la señal a un rango muy estrecho, dado que la amplificadores no son lineales en rangos muy amplios. Esta comprensión denominada ***comprensión−expansión****, puede introducir algunos errores. Por último, resulta imposible producir una portadora de onda perfecta, su amplitud, frecuencia y fase siempre exhibirán ciertas fluctuaciones.*

Para detectar errores, se añade un código en función de los bits de la trama de forma que este código señale si se ha cambiado algún BIT en el camino. Este código debe ser conocido e interpretado tanto por el emisor como por el receptor.

Para entender fácilmente cómo puede funcionar un código detector de errores se va a definir el término distancia en un código binario.

Distancia entre dos combinaciones binarias es el número de bits que deber ser modificados en una de las combinaciones para obtener la otra.

Ejemplo:

11100110

11110011

Distancia 3

Distancia de un código binario es la menor de las distancias entre dos combinaciones binarias cualesquiera del mismo.

Para que un código pueda detectar errores, su distancia tiene que ser superior a la unidad, ya que se es 1, los errores de un BIT pueden llevarnos a otra combinación válida sin que podamos detectarlo.

Ejemplo:

- *000*
- *011*

2 110 Código de distancia 2

3 110

Cualquiera error de 1 BIT se detecta por producir un dato que no pertenece al código.

La distancia de un código está íntimamente ligada al número de errores capaz de detectar.

Los códigos correctores además de detectar el error lo corrigen, pero con el inconveniente de necesitar mayor número de bits que los anteriores.

La distancia mínima que debe tener un código para poder corregir errores de 1 BIT es 3. en general, para corregir en n bits es necesario que la distancia del código sea al menos de 2n + 1.

Para corregir estos errores los diseñadores de redes han desarrollado dos estrategias fundamentales para tratar los errores. Una de ella consiste en incluir una cantidad suficiente de información redundante, junto con cada bloque de datos enviado, para permitirle al receptor deducir cuál fue el carácter que se transmitió. La otra estrategia consiste en incluir suficiente redundancia para permitirle al receptor deducir que ocurrió un error, pero qué tipo de error, y que tiene que solicitar una retransmisión. La primera estrategia utiliza ***códigos correctores de errores.***

A continuación se mencionará tres formas para detectar y corregir errores:

**Bit de paridad*

**Código de redundancia cíclica*

**Código de Hamming*

BIT DE PARIDAD

Este consiste en un único BIT, que indica si el número de bits con valor 1 enviados es par o impar. Es el método más elemental de detección de errores.

Este es un detector de error cuya distancia es 2, por lo que permiten controlar errores de 1 BIT. Se obtiene

añadiendo a las combinaciones del código un BIT que se denomina de paridad.

El criterio que se utiliza es el número de bits que se encuentran a 1 dentro de cada combinación, incluido el de paridad.

Existen dos tipos de códigos de paridad:

**Paridad par: El número de bits a 1 es par (muy utilizado)*

Ejemplo; 1011101 1

**Paridad impar: El número de bits a 1 es impar.*

Ejemplo; 1011101 0

Estos códigos también pueden contabilizar el número de bits a 0, pero apenas se utilizan con este criterio.

Ejemplo: Código BCD natural con paridad par.

- *0000 0*
- *0001 1*
- *0010 1*
- *0011 0*
- *0100 1*
- *0101 0*
- *0110 0*
- *0111 1*
- *1000 0*

9 1001 0 Código de distancia 2

BIT de paridad par

En este código se detectarían errores de 1 o 3 bits que producirán una combinación no perteneciente al código.

Veamos otro ejemplo.

0000000000

0000011111

1111100000

1111111111

Este código tiene una distancia de 5, lo cual significa que puede corregir errores dobles. Si llegara a presentarse la palabra código 0000000111, e receptor sabrá que la original debería haber sido 0000011111. sin embargo, si un error triple cambia el patrón 0000000000 a 0000000111, el error no se podrá corregir apropiadamente.

Nos pondremos en el caso que se desea diseñar un código con m bits de mensaje y r bits de redundancia, que

permita corregir todos los errores individuales. Cada uno de los -2m mensajes legales tienen n palabras código ilegales, a una distancia de 1. Estos se forman al invertir, en forma sistemática, cada uno de los n bits en las palabras claves de n bits.

Este límite inferior puede alcanzarse mediante el empleo de un método desarrollo por Hamming. Los bits de la palabra código son numerados en forma consecutiva comenzando con el bit1, localizando en el extremo izquierdo. Los bits que son potencia de 2(1, 2, 4, 8, 16, etc.), son bits de redundancia; y el resto de ellos (3, 5, 6, 7, 9, etc.), se llenan.

Con los m bits de datos. Cada uno de los bits de redundancia fuerza la paridad de alguna colección de bits, incluyéndose a sí mismo, para que sea par (o impar. Se puede incluir el mismo BIT en varios cálculos de paridad. Para saber a qué bits de redundancia contribuye el BIT de datos de la posición d, se rescribe k como la suma de potencias de 2. Por ejemplo, 11 aquellos bits de redundancia que aparecen en su expansión (es decir, el BIT 11 se comprueba mediante el uso de os bits 1, 2, y 8.

Cuando llega la palabra código, el receptor inicia un contador a cero. Después, revisa cada uno de los bits de redundancia, k (k = 1, 2, 4, 8...), para ver si tiene la paridad correcta. Si no es así, el receptor suma el valor k al contador. Si el valor del contador es cero, después de haberse revisado todos los bits de redundancia (es decir, si todos fueron correctos), la palabra código se acepta como válida. Si porque es el único comprobado por los bits 1, 2 y 8.

CÓDIGO DE REDUNDANCIA CÍCLICA

CRC: (CYCLIC REDUNDANCY CHECK, también conocido como código polinomio)

Esta técnica es muy utilizada en redes de LAN, además es ampliamente utilizada debido a que es fácil de implementar en los circuitos integrados a muy gran escala (VLOSI) que forman el hardware. Un mensaje puede verse como un simple número binario, el cual puede ser división por una cantidad que consideremos constante, al efectuar la división (a módulo 2) se obtiene un cociente y un residuo, este último es transmitido después del mensaje y es comparado en la estación receptora con el residuo obtenido por la división de los datos recibidos y el mismo valor constante. Si son iguales los residuos se acepta e mensaje, de lo contrario se supone un error de transmisión. En el proceso de datos comercial es ampliamente usada la verificación por redundancia cíclica de 16 bits de longitud, aunque también es posible usar 32 bits lo cual puede ser más efectivo.

Esto se basa en códigos polinominales, ya que no tenemos caracteres, sino una secuencia de bits.

Estos códigos se basan en el tratamiento de series de bits como si fuera representaciones de polinomio, con coeficientes de valor 0 y 1 únicamente. Una trama de k bits se ve como una lista de coeficientes de un polinomio con k términos, cubriendo un rango desde Xk−2, y así sucesivamente. Por ejemplo, el código 110001 tiene 6 bits y, por consiguiente representa a un polinomio de seis términos, que contiene los siguientes coeficientes 1, 1, 0, 0, 0, 1, es decir: X5 +X46 + X0. De acuerdo con las reglas de la teoría del campo algebraico, la aritmética del polinomio se realiza en módulo 2. no hay términos de acarreo para la suma ni de préstamo para la resta; las dos operaciones son idénticas al OR EXCLUSIVO. Por ejemplo:

100110011 00110011 11110000 01010101

+11001010 +11001101 −10100110 −10101111

01010001 11111110 01010110 11111010

La división larga se realiza de la misma manera que en caso binario, con excepción de la resta que se efectúa

en módulo 2, como en el caso anterior. Se dice que divisor cabe en el dividendo, si tiene tantos bits como este último.

Cuando se emplea el método del código polinomio, el emisor y el receptor deberán estar de acuerdo respecto a un polinomio generador, G(x), en forma anticipada. Los bits de orden superior e inferior del generador deben ser 1. para calcular el código de redundancia de alguna trama con m bits, correspondiente al polinomio M(XC), la trama deberá sé más grande que el polinomio generador la idea básica consiste en incluir un código de redundancia al final dela trama, de tal manera que, el polinomio representado por la trama con el código de redundancia sea divisible por G(x), cuando el receptor recibe la trama de suma comprobada, intenta dividirla entre G(x). Cuando el recepto recibe la trama de suma comprobada, intenta dividirla entre G(x). Si existe un resto, habrá ocurrido un error de transmisión.

El algoritmo para calcular la redundancia es el siguiente:

- *Sea r el grado de G(x). Agregar r bits a cero al extremo de orden inferior de la trama, de tal manera que ahora contenga m + r bits, y corresponda al polinomio xr M(x).*
- *Dividir la serie de bits correspondientes a xr M8x) entre la serie de bits correspondientes a G(x(), empleando la división en módulo 2.*
- *Restar el resto (que siempre tiene r o menos bits) de la serie de bits correspondientes a xr M8x), empleando la resta en módulo 2. el resultado es la trama lista para transmitir.*

Se deberá quedar claro que T(x) es divisible (módulo 2) por G8x). En cualquier problema de división, si se le diminuye el resto al dividendo, lo que queda es divisible por divisor. Por ejemplo, considerando la base 110, se divide 210278 entre 10941, el resto es 2399. al restar 2399 dela cantidad 210278, el resultado que da es(207879) es divisible pro 10941.

Ahora veamos la potencialidad de este método. ¿Qué tipo de errores se detectaran'. Imagínense que ocurre un error en una transmisión, así que, en lugar de recibir el polinomio para T(x), llega T(x) + E(x). Cada BIT con valor de 1 en E(x), habrá ocurrido k errores de un solo BIT. Una sola ráfaga de errores se caracteriza por tener un 1 inicial, una mezcla de 0 y 1m y un 1 final, con el resto de los bits teniendo un valor de 0.

Una vez que se recibe la trama con la redundancia, el receptor la divide entre G(x); es decir, calcula [T(x) + E(x) / G(x). El término T(x)/G(x). Aquellos errores inadvertidos pero todos los demás errores quedarán detectados.

Si llega a ocurrir un error de un solo BIT, E(x) = x, donde el valor de i determina cuál es el BIT que contiene el error. Si G(x) contiene dos o más términos, nunca dividirá a E(x), así que todos lo errores de un solo BIT serán detectados.

Si llega a ocurrir dos errores aislados de un solo BIT, E(x) = Xi + Xj, donde i > j. En forma alternativa, esta expresión se puede escribir como E(x) =Xi(xi − J + 1). Si se supone que G(x) no es divisible por x,, una condición suficiente para que todos lo errores dobles sean detectados es que G(x) no divida a Xk + 1, para cualquier k hasta un valor máximo de i−j (es decir, hasta la máxima longitud de la trama). Se conocen algunos polinomios simples, de grado inferior, que brindan protección a tramas largas. Por ejemplo, x15 + x14 +1 m no dividirá a xk + 1, para ningún valor de k que esté por debajo de 32768. para ver que no hay polinomio, con un número impar de términos, que pueda ser divisible por x + 1.

Para que quede un poco más claro veamos el siguiente ejemplo.

BER: BIT Error Rate, probabilidad de que un BIT sea erróneo (Ph), este parámetro depende del medio, ruido, etc.

Ahora se mostrará el siguiente ejemplo: a partir de Pb podemos saber la probabilidad de que una trama sea errónea o no (Pf). Si tenemos una trama de L bits se pude deducir lo siguiente:

L

*Pf = 1 − (1Pb) =L*Pb si L*Pb <<1 (1−Pb) probabilidad de trama*

Si la longitud de trama L es muy grande entonces (1−Pb) tiende a cero y por lo tanto la Pf de trama tenderá a uno, y por lo tanto casi siempre se producirá un error.

En CRC se define la secuencia de bits a transmitir como un polinomio:

M(x)=Sk−1 k−1 + Sk−2 X + Sk−2 X k−2 + ... + S1x + S0

Donde la k representa el número de bits que contiene datos. Estos k representa el número de bits que contiene datos. Estos k bits de datos se pueden representar por un polinomio de grado d−1.

CRC = resto M (x) − xL M (x) + CRC = 0

G(X) G(X)

Ejemplo:

Datos = 4 bits = L

M(x)=x7 + x6 + x5 + x2 + x

G(x) = x4 + x3 + 1 escogido al azar.

*M(x)*x4 = x11 + x10 + x9 + x6 + x5*

X11 + x10 + x9 + x5 !x4 + 1

X11 + x10 + x7 x7 + x5 + x4 + x2 + x

X9 +x7 + x6 + x5

X9 + x8 + x5

X8 + x7 + x6

X8 + x7 + x4

X6 + x4

X6 + x5 + x2

X5 + x4 + x2

X5 + x4 + x

X2 + x

Con un G(x) bien escogido se pueden detectar:

−Errores de un solo bit

- *errores de un número impar de bits*
- *dos bits erróneos*
- *ráfagas de tamaño menor que k*
- *ráfagas de tamaño mayor que k*

Las ráfagas son el número de bits que hay entre bits erróneos: 1<u>1</u>00110<u>0</u>0 ráfaga = 6.

Un G(x) es el CRC−CCITT de 16 bits = x16 + +x15 + + x2 + 1

CÓDIGO DE HAMMING

Son códigos correctores de errores cuya distancia mínima es 3 y permiten detectar errores de 2 bits y corregir errores de 1 bit.

Se forman añadiendo al código a proteger una sie de bits para detectar varias paridades. El conjunto de bits que se añaden forman un número en binario puro que indica la posición del bit erróneo. Caso de no haber error,, el número será 0.

N bits	*p*

p

P detección de paridad

N binario de p bits

`0 no hay error

`1 error en 1−ésimo bit'

El número p de bits añadidos debe ser tal que:

2p > = n + p + 1

Vamos a transmitir números BCD natural con 7 bits:

b b b b b b b

6 5 4 3 2 1

Número en BCD natural

b b b son los tres bits de paridad.

2 1

Las combinaciones de paridad son:

C3 c2 c1

0 0 0 : si no hay error

0 1 1 : si es erróneo b1

1 0 2 : si es erróneo b2

1 1 3 : si es erróneo b3

0 0 4 : si es erróneo b4

0 1 5 : si es erróneo b5

1 1 0 6 : si es erróneo b6

1 1 7 : si es erróneo b7

El proceso para obtener los c3 c2 c1 es:

C1 vale 1 si hay error en b7 b5 b3 o b1,

C1 vale 1 si hay error en b7 b6 b3 o b2,

C1 vale 1 si hay error en b7 b6 b3 o b4,

Por tanto:

B1 es un bit de paridad par entre b7, b5, y b3,

B2 es un bit de paridad par entre b7, b6, y b3

B4 es un bit de paridad par entre b7, b6, y b3

El código Hamming resultante será

b7	*b6*	*b5*	*b4*	*b3*	*b2*	*b1*	
0	*0*	*0*	*0*	*0*	*0*	*0*	*0*
1	*0*	*0*	*0*	*0*	*1*	*1*	*1*
2	*0*	*0*	*1*	*1*	*0*	*0*	*1*
3	*0*	*0*	*1*	*1*	*1*	*1*	*0*
4	*0*	*1*	*0*	*1*	*0*	*0*	*0*
5	*0*	*1*	*0*	*1*	*1*	*1*	*1*
6	*0*	*1*	*1*	*0*	*0*	*0*	*1*

7	*0*	*1*	*1*	*0*	*1*	*1*	*0*
8	*1*	*0*	*0*	*1*	*0*	*0*	*1*
9	*1*	*0*	*0*	*1*	*1*	*1*	*0*

Por ejemplo, si queremos transmitir 0011001 (2) y recibimos 0001001, que evidentement4e no pertenece al código.

0011001

0001001

c1: b7 = 0 b5 = 0 b3 = 0 b1 = 1 (impar) c1 = 1

c2: b7 = 0 b6 = 0 b3 = 0 b1 = 0 (par) c2 = 0

c3: b7 = 0 b6 = 0 b5 = 0 b1 = 1 (impar) c3 = 1

1 0 1 (5) error en bit 5. se corrige cambiándolo.

Analicemos : Sean los mensajes secuencias binarias de n bits, esto hace que los mensajes o vectores de n bits formen un conjunto de 2n posibles. La distancia Hamming o distancia entre dos vectores es el número de posiciones de bits en los cuales dichos vectores toman valores diferentes. Por ejemplo, sean 2 vectores de 4 bits:

***c1** = 0 1 0 1 y **c2** = 1 1 0 0*

*La distancia entre **c1** y **c2** es 2.*

Supongamos que de entre los 2n posibles vectores elegimos un subgrupo que constituyen nuestro código. Pues la distancia del código será la mínima distancia Hamming entre todas las posibles combinaciones de codewords. Se demuestra que si la distancia de un código es d, entonces podemos detectar y corregir los errores que afecten a t bits de cada codeword, siendo: A cada n bits de datos se le añaden k bits de paridad de tal forma que el carácter transmitido tiene n+k bits de longitud. Los bits se numeran de izquierda a derecha (el 1º BIT es el más significativo).

Todo BIT cuyo número sea potencia de 2 es un BIT de paridad, los restantes serán bits de datos. Los bits de dato se acomodan en sus posiciones y los bits de paridad se calculan dé modo que tengan una paridad par sobre los bits cuyo número de BIT forme, por ejemplo: El BIT 1 (paridad) es determinado por los bits de datos: 3 (1+2=3), 5 (1+4=5), 7 (1+2+4=7), 9 (1+8=9), etc. De esta forma cada BIT está verificado por una combinación única de bits de paridad, de modo que analizando los errores de paridad se puede determinar que BIT es el que ha invertido su estado.

A continuación se dan algunos ejemplos que muestran cómo se pueden localizar los bits alterados:

Paridad incorrecta en los bits	***El error está en el BIT número***
4 *1 y 4* *1, 2 y 4* *1 y 8*	*4* *5* *7* *9*

En el caso que exista más de un error en el bloque de información se llegan a producir varias situaciones que pueden llevar a la "corrección" de un BIT no alterado (Ej.: si cambian los bits 1 y 2 llevan a la corrección del BIT sano 3), entre muchas otras situaciones. Una variante del código Hamming es adicionarle 1 BIT de paridad global. De esta forma es posible tener la seguridad de detección de 2 errores, manteniendo la capacidad de corrección si se produce sólo 1 error.

Desventajas del código Hamming

La cantidad de bits de paridad empleados en la transmisión de la información le restan eficiencia al proceso. Se define la eficiencia de transmisión con la siguiente fórmula:

$$\text{Eficiencia} = \frac{\text{Información / Tiempo unitario}}{\text{Capacidad / Tiempo unitario}}$$

Suponiendo que se desea transmitir bloques de 8 bits de información, se necesitan 4 bits de paridad para ello, con lo que se tiene un total de 12 bits. La eficiencia sería:

$$\text{Eficiencia (8+4)} = \frac{8}{12} = 0.6666\ (\times 100) = 66.66\ \%$$

La eficiencia de este tipo de transmisión resulta de 66.66% debida solamente al plan de codificación. Además, dependiendo del método de transmisión puede decaer todavía más.

<u>ALGORITMOS DE ENCAMINAMIENTO</u>

Los algotitmos de encaminamiento se pueden agrupar en dos clases principales:

no adaptativos y adaptativos.

Los algoritmos no adaptativos no basan sus decisiones de encaminamiento en mediciones o estimaciones del tráfico o topología actuales, más bien, la elección de la ruta utilizable para ir de la i a la j (para toda i y j), se determina anticipadamente, fuera de línea y se carga en los IMP cuando la red se arranca. A este procedimiento se le denomina a veces ***encaminamiento estático.***

Los algoritmos adaptativos, por otra parte, intentan cambiar sus decisiones de encaminamiento para reflejar los cambios de topología y de tráfico actual.

A continuación se estudiarán con mayor detalle estas tres clases de algoritmos.

Encaminamiento por el camino más corto

Comencemos nuestro estudio de los algoritmos de encaminamiento por medio de una técnica que se utiliza ampliamente de maneras diferentes, gracias a su simplicidad y facilidad de comprension. La idea consiste en construir un grafo de la subred, con cada nodo representando una IMP y cada arco, una línea de comunicación. Para escoger una ruta entre un par de IMP dadas, el algoritmo sólo determina el camino más corto que existe entre ellos.

El concepto de encaminamiento más corto merece una explicación. Una forma de medir la longitud del camino es a través de número de saltos. Empleando esta métrica, los caminos ABC y ABE, en la figura 5−10, tienen la misma longitud. Otra métrica es el camino geografico expresado en kilómetros, en cuyo caso la trayectoria ABC es claramente mucho más larga que ABE (suponiendo que la figura se ha dibujado a

escala).

Sin embargo, existe la posibilidad de utilizar muchos otros tipos de métrica.

Por ejemplo, cada arco podría etiquetarse con el retardo promedio de la espera en la cola y de la transmisión, para un paquete patrón de prueba, determinado en base pruebas de varias horas o días. Con este etiquetado del grafo el camino más corto resulta ser el más rápido en lugar del que representa el menor número de arcos o kilómetros.

Encaminamiento múltiple

Hasta ahora se ha supuesto tacitamente que existe un solo mejor camino entre cualquier par de nodos y que todo el tráfico entre ellos deberá utilizarlos. En muchas redes hay varios caminos entre pares de nodos, que son casi igualmente buenos. Con frecuencia, se debe obtener un mejor rendimiento al dividir el tráfico entre varios caminos, para reducir la carga en cada una de las líneas de comunicación. La técnica de utilizar encaminamiento multiple entre un solo par de nodos se conoce como encaminamiento de camino múltiple, o algunas veces como encaminamiento bifurcado.

El encaminamiento de camino múltiple se aplica tanto en subredes con datagramas, como en subredes con circuítos virtuales. Para el caso de subredes con datagramas, cuando un paquete llega a un IMP para su reexpedición, se hace una selección entre varias alternativas, para ese paquete en particular, en forma indepediente de las selecciones que se hicieron par a otros paquetes que se dirigieron al mismo destino en el pasado. Para subredes con circuítos virtuales, cada vez que se establece un circuíto virtual se selecciona una ruta, pero el encaminamiento para los diferentes circuítos virtuales (en beneficio de los diferentes usuarios) se lleva a cabo en forma independiente.

El encaminamiento de camino míltiple se puede realizar de la siguiente manera.

Cada IMP mantiene una tabla con una ristra reservada para cada uno de los posibles IMP destinatarios; cada ristra ofrece la primera mejor, la segunda mejor, la tercera mejor, etc. línea de salida para ese destino en particular junto con un paso o ponderacióon relativa. Antes de que se reexpida un paquete, un IMP genera un número aleatorio y después selecciona entre las diferentes alternativas que se presentan haciendo uso de los pesos como probabilidades. Los operadores calculan las tablas manualmente cargándolas en los IMP antes de que se arranque la red y que no se modifique después.

Con un tiempo considerese la subred de la figua 5−12 a en la figura 5−12−b se muestra la tabla de encaminamiento del IMP J Si J recibe un paquete cuyo destino es A este utiliza la risura etiquetada con A. En este caso se presentan tres opciones .

La línea hacia A es la primera elección y después le siguen las líneas hacia I y H respectívamente . Para tomar una decisión , J genera un número aleatorio entre 0.00 y 0.99 . Si el número es inferior a 0.63, se utilizará la línea A. Si el numero se encuentra entre 0.63 y 0.83, se utilizará I; y sino , utilizará H. Los tres pesos representan , por consiguiente ,las respectivas probabilidades para que se utilice A, I o H

Una de las ventajas del encaminamiento de camino múltiple sobre el encaminamiento por camino más corto, es la posibilidad de poder trasmitir diferentes clases de tráfico sobre diferentes caminos. Por ejemplo, una conexión entre un terminal y un ordenador remoto, que consta de paquetes cortos que deben de entregarse rápidamente, podría tomar una ruta a través de líneas terrestres, en tanto que la transferencia de un archivo grande, que necesita un gran ancho de banda, podría transmitirse a través de un enlace Vía satélite . No sólo ofrece este método el gran ancho de banda que necesita la transferencia de archivos .

Aunque el encaminamiento de trayectoria múltiple se utiliza extensamente con objeto de mejorar el

rendimiento, también se puede utilizar para mejorar la fiabilidad de la subred. En particular, si las tablas de encaminamiento contienen N rutas separadas entre cada par de IMP, la subred entonces podrá soportar la pérdida de {N−1} líneas ,sin que se rompa en dos partes .

Una manera simple de asegurar que todas las rutas alternativas se separen, consiste en calcular pimero el camino más corto entre la fuente y el destino. Después eliminar el grafo todos los nodos y arcos utlizados en la trayectoria más corta , para finanalmente calcular la trayectoria má corta, a través del nuevo grafo.

Con este algoritmo se asegura que los fallos originados en la línea o en el IMP del primer camino, no vayan a hacer que el segundo también falle. Además, al eliminar el segundo camino del grafo, se puede calcular un tercer camino que vine a ser completamente independiente de los dos primeros. Even {1975} diseñó un algoritmo más sofisticado para localizar caminos separados en un grafo.

Encaminamiento centralizado

Todos los algoritmos de encaminamiento que se estudiaron anteriormente necesitan tener información acerca de la topología y el tráfico de la red, para poder tomar las mejores decisiones. Si la topología es de característica estática y el tráfico cambia muy rara vez. La construcción de las tablas de encaminamiento es muy sencilla y se realiza una sola vez, fuera de línea, cargándolas en los IMP.

Sin embargo, si los IMP y las líneas se desactivan y después se reestablecen, o bien, si el tráfico varía violentamente durante todo el día, se necesitará algún mecanismo para adaptar las tablas a las circuntancias que imperan en ese momento. En esta sección se estudiarán las técnicas para la construcción de las tablas de encaminamiento en un lugar central. En las secciones posteriores se verá la manera como este trabajo puede realizarse en una forma total o parcialmente descentralizada.

*Cuando se utiliza un encaminamiento centralizado, en alguna parte de la red hay un RCC CENTRO DE CONTROL DEL ENCAMINAMIENTO. Periódicamente, cada IMP trasmite la **formación de su estado al RCC por ejemplo, una lista de sus vecinos activos, las longitudes actuales, las coloas de esperas, la cantidad de tráfico procesado por línea desde ultimo informe, etc.) El RCC. Recoje toda esta información y después, con base en el conocimiento total de la red completa, calcula las rutas óptimas de todos los IMP. A cada uno de los IMP restantes, utilizando por Ej. El algoritmo de camino más corto, que se estudio anteriormente A partir de esta información puede construir nuevas tablas de encaminamiento y distribuirlas a todos los IMP.*

El encaminamiento centralizado también tiene algunos serios, si no es que fatales, inconvenientes, por una razon, si la sub red se tiene que adaptar a un tráfico variable, el cálculo del encaminamiento tendrá que efectuarse constante frecuencia. Para una red grande, este cálculo tomará muchos segundos , incluso cuando se tenga un CPU razonablemente rápido. Si el proposito de correr el algoritmo consiste en adaptarlo a cambios en la topología de la red y no tanto a cambios en el tráfico, podría ser adecuado si se llegara a ejecutar cada minuto aproximadamente , dependiendo de cuán estable sea dicha topología .

Encaminamiento aislado

Todos los problemas que se sucitan con los algoritmos de encaminamientos centralizados sugieren que los algoritmos desentralizados podrían tener algo que ofrecer. En los algoritmos de encaminamientos decentralizados mas sencillos, los IMP llevan a cabo decisiones de encaminamiento, únicamente basadas en la informacion que ellos mismos hayan reunido. No intercambian información de rutas con otro IMP: Sin embargo, tratan de adaptarse a los cambios de topología y tráfico que se llegan a presentar. A estos se les conoce comúnmente como algoritmos de encaminamiento adaptables aislados.

Un algoritmo adaptable aislado sencillo es el desarrollo por baran (1964), conocido como el algoritmo de la patata caliente. En el momento en que le llega un paquete el IMP trata de deshacerse de él tan rápido como

le sea posible, al ponerlo en la cola de espera de salida más corta. En otras palabras, cuando llega un paquete, el IMP cuenta el número de paquetes que se encuentran en la cola de espera de cada una de las líneas de salida. Entonces instala el nuevo paquete al final de la cola de salida más corta, sin tomar en cuenta el lugar al que se dirige esta línea.

Inundación

La inundación es un caso de encaminamiento aislado, en el cual cada paquete que llega se transmite en todas las líneas de salida, exeptuando aquella por la que llega. Obviamente, con la inundación se genera un número considerable de paquetes duplicados; de hecho un número infinito, a menos que se tomen algunas medidas para amortiguar el proceso. Una de tales medidas consite en tener un contador de saltos, contenido en la cabecera de cada uno de los paquetes, el cual se decrementa con cada salto que lleve a cabo, y el paquete se desecha en el momento en que el contador alcance el valor cero. Idealmente, el contador de saltos deberá iniciarce con un valor correspondiente a la longitud del camino que existe entre el origen y el destino. Si el emisor no conoce la longitud del camino, puede iniciar el contador con el valor del peor caso , es decir , el valor del diámetro completo de la subred .

Una técnica alternativa para retener la inundación consiste en hacer que el IMP origen ponga un número de secuencia en cada paquete que recibe de su hostal , de esta forma cada IMP nesecitará una lista por IMP origen, indicando que números de secuencia originados en la fuente ya fueron vistos, para evitar la lista crezca sin límite, cada lista deberá aumentarce por medio de un contador, K, indicando que todos los números de secuencia hasta K ya fueron vistos. N el momento en que un paquete llega, resulta muy fácil verificar si éste es un duplicado; Si es el caso, se desechará.

En varias aplicaciones, la inundación no resulta ser muy práctica pero si tiene algunos usos importantes. Por ejemplo, en aplicaciones militares, en donde un gran número de IMP pueden desintegrar en pedazos en cualquier instante, la robustez de la inundación en una característica altamente deseable. En aplicaciones de bases de datos distribuidas, algunas veces se necesita actualizar todas las bases de datos en forma concurente en cuyo caso la inundación puede ser de gran utilidad. Una tercera forma de utilización de la inundación es como un sistema de medición contra el cuál otros algorimos de encaminamiento se pueden comparar. La inundación siempre escoge el camino más corto porque selecciona todos los posibles caminos en paralelo, por consiguiente ningun otro algoritmo puede producir un retardo más corto (si se llegara a ignorar la sobrecarga generada por el proceso mismo de inundación).

Encaminamiento distribuido

En esta clase de algoritmos de encaminamiento que en sus inicios se utilizaron en ARPANET, cada IMP intercambia periódicamente información de encaminamiento explicitos como cada uno de sus vecinos. Por lo que general, cada IMP mantiene una tabla de encaminamiento con una entrada por cada uno de los demás IMP de la subred. Esta entrada consta de 2 partes: La línea preferida de salida que se utilice para dicho destino y alguna estimación del tipo o distancia hacia el. La métrica utilizada podria ser el número de saltos, el tiempo estimado de retardo en milisegundos, la estimación del número total de paquetes encolados a lo largo de la ruta, el exceso del ancho de banda o bien algo similar.

Se supone que el IMP conoce la distancia a cada uno de sus vecinos si la métrica es la del salto , la distancia es sólo un salto. Si la métrica es la longitud de la cola espera, el IMP simplemente investiga cada una de las colas si la métrica es el retardo, el IMP puede medirlo en forma directa con paquetes especiales eco, que el receptor sólo marca con sello de tiempo, y lo envia devuelta tan rápido como puede.

Como un ejemplo, supongase que se utiliza el retardo como una métrica y que el IMP conoce el retardo para llegar a cada uno de sus vecinos. Cada IMP le envia a cada uno de sus vecinos una lista de sus retardos estimados para cada destino una vez cada T mseg. También recibe una lista parecida de cada uno de sus

vecinos. Imaginese ahora que cada una de estas tablas acaba de llegar, procedente de sus vecinos X, con X siendo la estimación hecha por X del tiempo que lleva llegar al IMP I . Si el IMP sabe que el retardo para X, es de m mseg., también sabe que puede alcanzar al IMP a través de X, en un tiempo de (Xi + m) a través de X. Al efectuar este cálculo para cada vecino un IMP podría determinar que estimación parece ser la mejor y utilizarla junto con su correspondiente línea en una nueva tabla de encaminamiento. Nótese que la antigua tabla de encaminamiento no se utiliza en el proceso de cálculo.

Este proceso de actualización se ilustra en la figura 5−15. En la parte a) se muestra una subred. En las primeras cuatro columnas de la parte b) se muestran los vectores de retardo que se recibieron, procedentes de los vecinos del IMP J. A dice ter 12 mseg. De retardo para llegar a B, 25 mseg de retardo para Cc, 40 mseg de retado para D, etc. Supóngase de que J ha medido o estimado su retardo para llegar hasta su vecino A, I, H y K en 8, 10, 12 y 6 mseg, respectivamente.

COMPRESION DE DATOS

La compresión de datos ha sido estudiada en muchos contextos durante muchos años, esto es fundamental, ya que con la compresión de datos, se abaratan los costos de la transmisión de estos, además de ahorro de espacio en la memoria, en discos y en cintas magnéticas, sin perder la exactitud de los datos.

A continuación se describen y explican las técnicas que se utilizan para llevar a cabo esta importante función de la capa de presentación del modelo OSI.

TECNICAS DE COMPRESION DE DATOS

Los datos que se transmiten por un canal pueden verse como una

secuencia de simbolos, se supone que estos se extrajeron de algun conjunto de simbolos (finito).

La compresion de datos puede obtenerse de tres tecnicas generales:

1. Codificación de un conjunto finito de símbolos igualmente probables
2. Codificación dependiente de la frecuencia
2.1 Codificación Huffman.
2.2 Codificación Aritmética.
3. Codificación dependiente del contexto

De todas formas hay que tener en cuenta que en la práctica estas técnicas no se suelen aplicar por separado, sino que suelen venir unidas a técnicas de corrección de errores.

1. Codificación de un conjunto finito de símbolos igualmente probables

Esta técnica es muy particular, ya que está condicionada por varios aspectos:

- *El conjunto de símbolos a transmitir debe ser finito.*
- *La probabilidad de aparición de los símbolos no influye (se consideran equiprobables).*
- *Ambos interlocutores deben conocer todos de antemano todos los símbolos que se pueden trasmitir (el diccionario de posibles símbolos).*

El desarrollo de la comunicación con esta técnica sería el siguiente:

- *Determinar cuál es exactamente el conjunto de datos que se pueden transmitir.*
- *Hacer llegar a los interlocutores de la comunación este conjunto de símbolos válidos mediante cualquier método que no sea la transmisión normal(este método consigue compresión gracias a no tener que enviar los datos completos, así que sí se transmiten normalmente no estaremos consiguiendo ninguna ventaja).*
- *Proveer al sistema de algún mecanismo que permita añadir nuevos símbolos al conjunto (esto sólo se utilizará de forma ocasional).*
- *Establecer un orden en los símbolos válidos y mantenerlos numerados.*

Una vez hecho esto, ya se puede establecer la comunicación, sólo que ahora ya, en lugar de transmitir los símbolos se transmite su número de orden.

La ventaja de esta técnica no viene exactamente de la compresión de datos (aunque su fundamento es el mismo: transmitir una misma información con un número de bits menor), sino de que es posible identificar cada elemento del conjunto de símbolos con una cantidad de información menor.

Evidentemente, el uso de técnicas como esta sólo se puede realizar en entornos cerrados y lo más estáticos posible. Su uso internamente en empresas o instituciones es su única posibilidad, ya que en el mundo real nunca se podría conseguir aplicar esta técnica.

Por último, señalar que el aumento en la velocidad de transmisión conseguido mediante este método depende de forma directa de cuál es la longitud media en bits de los símbolos a transmitir. Las ventajas del método son su sencillez y que aunque el conjunto de símbolos sea grande, la cantidad de bits trasmitida sigue siendo pequeña para cada uno de ellos (el crecimiento del conjunto tampoco va a tener un impacto destacado en la transmisión).

2. Codificación dependiente de la frecuencia (compresión estadistica)

Los codificadores estadísticos son la piedra angular de la compresión. La idea básica de su trabajo es la siguiente: el compresor predice la entrada y escribe menos bits en su salida si la estimación ha sido correcta. El descompresor debe ser capaz de realizar las mismas predicciones que el compresor para poder decodificar bien lo transmitido, ya que la transmisión es diferente dependiendo de la predicción.

Lo normal, es que la preocupación se centre en el codificador estadístico, más que en el funcionamiento global del proceso de compresión. El codificador es el algoritmo que codifica o decodifica un caracter con un número de bits proporcional a log2 (probabilidad predicha).

*Supongamos un alfabeto con cuatro caracteres **A**, **B**, **C** y **D**. De ellos sabemos que las probabilidades son respectivamente: 50%, 25%, 25% y 0%, por lo tanto lo ideal sería una codificación de la siguiente forma:*

*1 bit para la **A** (log2(0.5)=1)*
*2 bits para la **B** y la **C** (log2(0.25)=2)*

quedando así el resultado:

***A**: 0*
***B**: 10*
***C**: 11*

Por lo tanto esta codificación sugiere un esquema, en el cual los símbolos más comunes se les asignen códigos cortos, y a los ocasionales largos.

Dentro de la codificación dependiente de la frecuencia (compresión estadística) destacan dos técnicas:

- *Codificación de Huffman.*
- *Codificación aritmética.*

Aunque la máxima optimización de esta técnica de compresión estadística se alcanzaría si fuera posible transmitir un número de bits fraccionario por cada símbolo, se puede conseguir una buena aproximación a este óptimo mediante la codificación Huffman*.*

El óptimo teórico viene dado por la siguiente fórmula:

Sum** (**Prob** i)log**2**Prob** i*

Esta fórmula proporciona el contenido medio por símbolo (longitud media en bits por símbolo). El sumatorio va desde 1 hasta N (cantidad de símbolos), refiriéndose a la probabilidad de aparición de cada símbolo.

2.1 Codificación Huffman

La codificación de Huffman que obtiene los mejores rendimientos posibles utilizando códigos de longitud entera.

La codificación de Huffman se realizan pasos hacia atrás. Con Shannon−Fano se empezaba desde el conjunto de símbolos que se encontraban en la raíz del árbol y se iba dividendo en dos subconjuntos de forma recursiva. Con Huffman se comienza el proceso con todos los símbolos fuera del árbol, entonces los vamos cogiendo para formar las hojas, después se forman los padres de estas hojas, y así sucesivamente hasta llegar a la raíz.

El proceso más en detalle se realizaría así:

- *tomemos cada caracter y lo ponemos en un nodo. Este nodo tendrá un número asociado que son las ocurrencias del caracter. En este momento los nodos no tienen ni padres ni hijos (estos nodos nunca van a tener hijos).*
- *Considerando sólo los nodos y su número asociado, los ordenamos de forma que los números más bajos estén primero.*
- *Ahora se marcan los dos más pequeños y creamos un nuevo nodo como padre de ambos. Para el su número será la suma de las ocurrencias de los dos hijos.*
- *Se añade el nuevo nodo a la lista ordenada de nodos en la posición correspondiente.*
- *Se eliminan de la lista los nodos que han quedado como hijos en el paso anterior.*
- *Se repiten los tres pasos anteriores de forma recursiva hasta que en la lista sólo queda un nodo, que será el nodo raíz.*

Todo esto se puede ver con el siguiente ejemplo:
Lo primero es ordenar los nodos,

[c]: 1
[d]: 1
[b]: 2

[a]: 4

tomando los dos más pequeños: [c] y [d] hacemos un nuevo nodo [[c],[d]]: 2 y lo ponemos en la lista ordenada:

[b]: 2
[[c],[d]]: 2
[a]: 4

tomando los dos más pequeños: [b] y [[c],[d]] hacemos un nuevo nodo [[b],[[c],[d]]]: 4 y lo ponemos en la lista ordenada:

[a]: 4
[[b],[[c],[d]]]: 4

Evidentemente hay que unir los dos que restan: [[a],[[b],[[c],[d]]]]:8, con lo que ya tenemos el nodo raíz. Sólo resta dibujar el árbol:

a

/

4

/

raíz b

\ /

4 2

\ /

X c

\ /

2 1

\ /

X

**

1

**

d

*La **codificación de Huffman** presenta un problema y es que codifica los símbolos de forma independiente (necesidad de un número entero de bits para cada símbolo) con lo cual aunque se consigue buenos ratios de compresión, no se acerca tanto al óptimo previsto. Esto se consigue mejorar mediante la **codificación aritmética**, que codifica los símbolos de otra forma consiguiendo que ya no sea necesario emplear n bits para representar (n−1). x bits de información.*

2.2 Codificación Aritmética

Para utilizar este algoritmo necesitamos tener siempre presente la tabla de frecuencias de los símbolos representables. Cada uno de ellos se representa como una porción de la recta de números reales entre 0 y 1.

Cuando queremos transmitir una cadena de símbolos, en cada una de las selecciones, la recta se irá reduciendo tomando la parte correspondiente al intervalo que representa el elemento a transmitir. LLegados al final de la cadena, se tomará un número cualquiera dentro del último intervalo (el correspondiente al último elemento a transmitir) y precisamente ese número será el que se transmita.

El receptor, a partir de él, podrá obtener la cadena que se deseaba transmitir ejecutando el proceso de forma similar, identificando en cada paso el símbolo correspondiente según el intervalo seleccionado.

Todo esto quedará mucho más claro con un ejemplo. Suponemos un caso en el que sólo hay tres símbolos válidos: A, B y C.
*La cadena a transmitir es: **CAB***

Tabla de frecuencias	
Símbolo	*Frec.*
A	*0.5*
B	*0.25*
C	*0.25*

Comenzamos la transmisión:

A B C Símbolo a transmitir: C

|−−−−−−−−−−−−|−−−−−−|−−−−−−| Intervalo: 0.75−1

0 0.5 0.75 1

A B C Símbolo a transmitir: A

|−−−−−−−−−−−−|−−−−−−|−−−−−−| Intervalo: 0.75−0.875

0.75 0.875 0.9375 1

A B C Símbolo a transmitir: B

|−−−−−−−−−−−−|−−−−−−|−−−−−−| Intervalo: 0.8125−0.84375

0.75 0.8125 0.84375 0.875

Finalmente el valor a transmitir será cualquiera en el último intervalo

seleccionado: 0.8125−0.84375. Por ejemplo, podríamos transmitir el 0.83

En cada uno de los pasos se ha ido reduciendo el intervalo según el símbolo elegido, y para calcular el nuevo intervalo, se ha recalculado los límites proporcionalmente a los valores indicados en la tabla de frecuencias.

En el ejemplo, finalmente se ha elegido transmitir el 0.83. Para identificarlo en el receptor, el proceso a seguir sería similar. Empezando desde el primer intervalo 0−1 y comprobando donde estaría el valor 0.83 en cada caso. Según la fracción de intervalo en la que esté incluido, tomamos un símbolo u otro.

*En este caso el receptor comprobaría que el valor 0.83 corresponde a la letra **C** (está entre 0.75 y 1), recalcularía el nuevo intervalo y vería que 0.83 está dentro de la región correspondiente a la **A**. Finalmente, haría lo mismo con la **B**, obteniendo finalmente la palabra a transmitir: **CAB***

3. Codificación dependiente del texto

A diferencia de la compresión estadística, la predictiva no supone que la probabilidad de aparición de un símbolo sea independiente del símbolo que le precede. Además, es lógico pensar que en cualquier lenguaje haya unos símbolos que siguen a otros con mayor probabilidad (por ejemplo, en español la probabilidad de que tras una consonante siga una vocal es mayor a que a continuación venga otra consonante).

Este método se basa en utilizar información que indica que símbolos aparecen tras cada uno de los otros posibles y con que probabilidad, para con estos datos reducir el número de bits a transmitir.

El problema de este tipo de métodos es que pueden requerir unas estructuras de datos muy grandes si el conjunto de símbolos también lo es. Pensemos en el conjunto de números: serían necesarias 10 tablas con una longitud de 10 elementos. Para la tabla del 0 se almacenarían las probabilidades de que detrás venga otro 0, un 1, un 2, etc. Lo mismo lo tendríamos para la tabla del 1, del 2, ...

*En resumen para n símbolos se necesitan n*n entradas para almacenar todas las probabilidades que se pueden tener.*

Para utilizar este tipo de compresión es necesario tener agrupada la información según el tipo, es decir, tratarla en su "estado original". Si estamos tratando un fichero de texto, se deben leer las cadenas en bloques de 8 bits; las imágenes se deben leer bit a bit, el sonido se debe descomponer en sus bandas, etc.

Sobre el funcionamiento del algoritmo hay que decir que tiene que ocuparse de ir actualizando las tablas de probabilidades para cada símbolo leído. En el emisor tienen que existir estas tablas para poder ir decodificando el mensaje enviado.

Muy en relación con este método está la codificación de series largas. Su utilidad se manifiesta cuando el conjunto de símbolos a transmitir es pequeño, y por lo tanto, es muy probable que el mismo símbolo se repita

varias veces seguidas (formando una serie). Como es lógico en el caso de transmisiones binarias (con sólo dos estados) su aplicación es muy útil.

Otra forma de conseguir importantes reducciones sobre todo cuando el tiempo es crítico y hay similitud relativa entre los elementos a transmitir. Se trata de utilizar una técnica que en lugar de transmitir las unidades de información completas, transmita sólo las diferencias relativas entre una y la siguiente. De esta forma, sólo es necesario completa la primera unidad, mientras que para las siguientes sólo necesitamos tener establecida alguna forma de transmitir las diferencias, que será lo que se envíe por el canal.

Este tipo de técnicas se utilizan en algunas tarjetas gráficas. De hecho, Hewlett−Packard tiene un sistema de compresión de este tipo con un rendimiento bastante bueno, disponible para las tarjetas gráficas de sus workstations. Este sistema les permite, unido a un algoritmo de aproximación, obtener una calidad similar a true color trabajando sólo con una profundidad de color de 8 bits (es decir, las transferncias que realizan a través del bus son de 8 bits por pixel, pero la visualización en pantalla es equivalente a 24 bits por pixel).

INTRODUCCIÓN A LA CRIPTOGRAFÍA

Pero no se habla de una clave, sino que los sistemas criptográficos están formados por un par de claves. Una de ellas es la que se usa para codificar el mensaje y la otra para descodificarlo. De esta forma el emisor solo tiene que conocer una clave y el receptor la otra, y el mensaje se descodificará solo con la clave apropiada. Desde este momento podemos diferenciar o hacer una primera clasificación:

- *claves simétricas*
- *claves no simétricas*

Las claves simétricas son aquellas en las que la clave que cifra y la clave que descifra son la misma. Por ejemplo, la clave utilizada por Julio César es simétrica ya que para cifrar había que rotar cada letra tres posiciones detrás en el abecedario, por lo que para descifrar había que rotar cada letra tres posiciones delante en el abecedario. Otros ejemplos basados en estos tipos de claves son los siguientes:

- *claves matriciales,*

cada cierto número de caracteres del mensaje se forma una fila de la matriz hasta completar el mensaje. Un vez formada la matriz se envía uno nuevo pero concatenando en este caso las filas. Un ejemplo sería el siguiente:

− mensaje: La casa es roja y verde.

− clave: 4

− matriz formada:

L	*a*		*c*
a	*s*	*a*	
e	*s*		*r*
o	*j*	*a*	
r	*d*	*e*	

− mensaje que se envía: Laeoyrassj dc r e

Para recuperar el mensaje bastaría con reconstruir la matriz por columnas, poniendo en cada columna un

número de caracteres determinado, reconstruyendo el mensaje concatenando las filas. Para conocer la longitud de cada columna se utiliza el siguiente algoritmo:

− si el resto de la longitud del mensaje entre la clave es cero entonces el número de caracteres a poner en cada fila es el cociente de esta división.

− en caso contrario es el cociente más uno.

En nuestro ejemplo la longitud del mensaje recibido es 23 y la clave es 4, por lo que según lo anterior el número de caracteres por columna sería 5+1.

- ***claves matriciales desplazadas***

En este tipo de claves se conjunta la utilización de claves matriciales y de desplazamiento como la utilizada por Julio César. Para encriptar el mensaje se utilizaría primero uno de los dos métodos obteniendo un resultado al que se le aplicaría el otro método. Por este motivo la clave en este caso sería de un par de números, uno que indica la longitud de las filas de la matriz y el otro que indica el desplazamiento. Por ejemplo del caso anterior habíamos obtenido que el mensaje codificado era el siguiente.

Laeoyrassj dc r e,

por lo que al aplicarle un desplazamiento de 3 quedaría,

Ñdhrbudvvm gf u h

l clave sería pues el par (4,3).

También se podrían seguir anidando matrices y desplazamientos, es decir, crear un mensaje matricial desplazarlo, volver a crear otro mensaje matricial, etc. Pero esto conlleva la implementación de más código para descifrar el mensaje y el aumento de la clave simétrica.

Las claves no simétricas son aquellas en las que la clave con la que se cifra el mensaje, es distinta a la clave con la que se descifra el mensaje. Pero de este tipo de claves hablaremos más adelante.

AUTENTIFICACIÓN

¿QUÉ ES LA AUTENTIFICACIÓN?

En materia propiamente de autenticación lo que se pretende no es que se pueda leer el mensaje por personas ajenas, sino que se pueda asegurar la procedencia del mismo.

Todo esto no es nuevo ya que lo hemos estado utilizando desde muy antiguo:

- *Los reyes de la edad media sellaban sus cartas lacrándolas y poniéndoles el sello en el lacrado.*
- *En mensajería usual (como CORREOS, empresas como SEUR, WRW,etc.) ponemos nuestra firma en cada envío.*
- *En las transferencias bancarias es necesario la firma de la persona que tiene la cuenta bancaria*

Y por supuesto en materia de nuevas tecnologías también hay claros ejemplos:

- *En los cajeros automáticos necesitamos una tarjeta identificadora y una clave de acceso a nuestra cuenta.*

- *En los accesos a los edificios de una cierta importancia como La Casa Blanca, el edificio central de la ONU en Bruselas, etc. muchas veces es necesario la posesión de una tarjeta identificadora y el siguiente escaneo de las huellas digitales.*
- *Y más cercano a nosotros es el acceso a sistemas unix en los que cada cuenta de usuario necesita un login y una clave de entrada.*

Los objetivos que se pretenden con la autenticación en el envío de mensajes son los siguientes:

- *certeza de que el mensaje procede la persona que dice remitirlo.*
- *se asegura de que ninguna persona ajena ha podido modificar el mensaje en cuestión.*
- *seguridad por parte del remitente de que el receptor no podrá modificar el mensaje, y decir que es el remitente el que escribió el mensaje.*

FORMAS DE AUTENTICACIÓN

Por supuesto las distintas formas de autenticación deben ser únicas, deben ser capaces de distinguir a la persona de entre las demás. Las firmas escritas son las más usuales aunque hay falsificadores expertos en la copia de firmas. Los nombres y claves de acceso a sistemas informáticos pueden ser obtenidos casi fácilmente por otros usuarios gracias a la utilización de tecnologías como analizadores de red, funciones especiales en la implementación de las tarjetas de red, etc. Los accesos a edificios con escaneo de huellas digitales pueden producir la tentación de amputar los dedos a la persona que tiene acceso por parte de otras.

Con todo queda de manifiesto que es muy difícil asegurar la veracidad de la identidad de las personas. Para ello lo que se hace es complicar el sistema de autenticación, para hacer más difícil a los que intentan hacerse pasar por quienes no son el acceso a los lugares privados o personales. Por ejemplo los detectores de huellas digitales implementan demás detección de temperatura y pulso sanguíneo para evitar la amputación y robo de los dedos.

AUTENTICACIÓN INFORMÁTICA Y FIRMAS DIGITALES

En nuestro caso se trata de verificar la identidad de las personas que acceden a los sistemas informáticos o que envían mensajes electrónicamente. Para ello se utiliza la encriptación con todo su potencial.

Para los mensajes o correos se utilizan las firmas digitales, que no es más que añadir al mensaje un pequeño texto cifrado

Como ya hemos dicho hay sistemas criptográficos con claves simétricas y con claves asimétricas. Nos vamos a centrar en las claves asimétricas que son aquellas que para cifrar utilizan una clave totalmente distinta que para descifrar.

CIFRADO Y AUTENTICACIÓN

El algoritmo más conocido de clave pública (asimétrica) es el RSA. Básicamente se trata de la generación de un par de números (claves), de tal forma que un mensaje cifrado con el primer número del par sólo podrá ser descifrado por el segundo número del par. Pero otra característica de este algoritmo es que la segunda clave no pude derivarse de la primera. Gracias a esto la primera clave podrá ser comunicada a cualquier persona que desee enviar un mensaje a la persona que tiene la segunda clave, sólo esta persona será capaz de descifrar el mensaje. Por esto a la primera clave se le llama clave pública y a la segunda clave privada.

El mecanismo del RSA se puede apreciar en la siguiente figura:

Este algoritmo se basa en la elección de un par de números primos P y Q que deben ser positivos y grandes,

ya que cuanto más grandes sean más dígitos tendrá otro número llamado N, y más difícil será factorizar este N en un tiempo razonable. Algunos ejemplos son los siguientes:

- *N=100 dígitos se tardaría una semana*
- *N=150 dígitos se tardaría 1000 años*
- *N>200 dígitos se tardaría 1 millón de años*

Pero claro estos datos se quedan anticuados casi con el paso de los días. Es decir, no se puede asegurar que, debido al enorme desarrollo de las tecnologías informáticas, el computador más rápido actual, con el que se han sacado estos datos, no sea el más lento mañana.

Ahora bien, como el receptor, el que tiene la clave privada, sabe que el mensaje recibido es de quien dice ser, ya que la clave pública ha sido comunicada y por tanto un intruso la puede obtener con relativa facilidad. Es entonces cuando entra en juego la autenticación de los mensajes.

PROBLEMAS Y SOLUCIONES DE AUTENTICACIÓN

Otra característica importante del RSA se que un mensaje cifrado con clave pública se descifra con la clave privada, como ya sabemos, pero también se puede descifrar con la clave pública un mensaje cifrado con la clave privada.

Gracias esto podemos definir un sistema de claves de tal forma que el emisor tiene la clave pública del receptor y su propia clave privada, y el receptor tiene la clave pública del emisor y su propia clave privada. Por lo que para enviar un mensaje el emisor seguiría los siguientes pasos:

- *cifrar el mensaje con su clave privada.*

- *cifrar el mensaje con la clave pública del receptor.*

Y para recibir el mensaje el receptor seguirá los siguientes pasos:

- *descifrar el mensaje con su clave secreta.*
- *descifrar el mensaje resultante con la clave pública del emisor.*

Si el receptor ha sido capaz de realizar los dos pasos con éxito sabe con bastante seguridad que el emisor es quien dice ser y por tanto el mensaje se legítimo. Aunque todo este proceso requiere un elevado gasto de procesamiento, por lo que sólo será útil para volúmenes de datos relativamente pequeños. Todo este tiempo empleado en el procesamiento depende como siempre de las ordenadores y los circuitos integrados de ayuda a la realización de todos estos cálculos.

En nuestro caso solo tenemos que verificar las identidades de los emisores de los mensajes, así que este método es el idóneo.

De forma similar lo que se suele hacer es una especie de código redundancia al mensaje que es lo que se cifra y toma el nombre de firma digital. Una vez hecho esto se envía el mensaje seguido de su código de redundancia cifrado. El receptor descifrará sólo la firma digital y hará el código de redundancia al mensaje de nuevo comparando los resultados de descifrar la firma digital y realizar la redundancia al mensaje. De esta forma se evita que el emisor rechace después el mensaje, porque en poder del receptor todavía permanece la firma cifrada con la clave privada del emisor que es el único que posee dicha clave.

Pero todo este sistema falla en cuanto que el emisor del mensaje puede, en ciertas circunstancias, decir que no ha sido él el que envió el mensaje, estas circunstancias son las siguientes:

- *Si una vez enviado el mensaje el emisor hace pública su clave privada, el receptor nunca podrá demostrar que fue emisor quien envió el mensaje ya que la clave la podría haber utilizado cualquiera, incluso el receptor.*
- *Si el emisor cambia de clave después de enviar el mensaje, el receptor nunca podrá demostrar que fue el emisor quien envió el mensaje ya que la clave privada no es capaz de cifrar la firma del mensaje.*

Cuando se habla de comunicaciones importantes como en negociaciones de contratos entre empresas, acceso a cuentas bancarias o acceso a sistemas informáticos de importancia, esto supone un enorme problema.

Esto es, imaginemos que una empresa envía un mensaje de pedido a otra por este sistema de autenticación, pero poco tiempo después la empresa que envía el pedido encuentra a otra empresa más barata. Entonces si alega que han robado en sus oficinas centrales y lo denuncia a la policía, la empresa que había recibido jamás podrá demostrar ante un juez que la otra empresa había enviado el mensaje. La empresa para rechazar que había enviado el mensaje también podría haber, simplemente cambiado de clave. Es casi inimaginable el caos que podría llegar si mentes maliciosas utilizaran este sistema para estafar a las demás personas, como en la bolsa, banca , etc.

Por esto parece necesario la existencia de una autoridad que esté por encima de los emisores y receptores, y que controle el movimiento y cambio de las claves.

Esto sugiere un nuevo sistema de autenticación en el que hay una tercer participante en la comunicación que actuará como servidor de distribución de claves. Este servidor se encargará de que cada usuario tenga su clave privada y cuando un usuario quiera comunicarse con otro usuario gestionará el protocolo de comunicación. Este protocolo será el siguiente:

- *El usuario 1 pide al centro servidor que escoja una clave de sesión y que le envíe dos copias de la misma. La petición se realiza con su clave y el envío de la clave de sesión con las claves del usuario 1 y el usuario*
- *El usuario 1 envía al usuario 2 la segunda copia para que la descifre, diciéndole que la utilice como clave de sesión.*

Correo electrónico

Cuando ARPANET entró en operación, sus diseñadores esperaban que llegaría a dominar el tráfico de proceso a proceso; sin embargo, se equivocaron. Desde su inicio, ha dominado el volumen del correo electrónico entre personas, con respecto al volumen de comunicación entre procesos. Mientras las inclemencias del tiempo o la obscuridad de la noche, han sido capaces de evitar que los mensajeros de las oficinas de correos completaran con éxito sus tareas asignadas, la capacidad de ARPANET para entregar un mensaje de costa a costa en unos segundos inició una revolución en los procedimientos empleados por la gente para comunicarse.

La atracción del correo electrónico es su rapidez. Pero existen otras ventajas que no son tan conocidas como ésta. El teléfono, por ejemplo, también proporciona un acceso inmediato, pero algunos estudios han demostrado que aproximadamente el 75% de las llamadas de negocios fallan en el intento de alcanzar su propósito. El correo electrónico tiene la misma velocidad que el sistema telefónico, pero a diferencia de éste no necesita que los interlocutores se encuentren disponibles en el mismo instante. Además también deja una copia escrita del mensaje que puede archivarse o enviarse. Además, el mensaje puede transmitirse a varias personas a la vez.

Aunque el correo electrónico puede verse efectivamente como un caso especial del proceso de transferencia de archivos, tiene varias características particulares, que no son comunes a todas las transferencias de archivos. Por una razón, casi siempre son personas, y no máquinas, los extremos emisores y receptores. Este

hecho ha dado como resultado que los sistemas de correo electrónico se construyan como dos partes distintas, pero estrechamente relacionadas: una de ellas proporcionada para la interfase humana (como sería la composición, edición y lectura de correo), y otra para el transporte del correo (administración de listas de correo y verificación de entrega).

Otra de las diferencias entre el correo electrónico y la transferencia de archivos de propósito general, es que los mensajes de correo son documentos muy estructurados. En la mayoría de los sistemas, los mensajes están compuestos por un gran número de campos además del contenido. En estos campos se incluye el nombre y dirección del emisor, el nombre y dirección del receptor, la fecha y hora de envío, una lista de personas que deberán recibir copia del envío, la fecha de expiración, el nivel de importancia , el margen de seguridad y algunos otros aspectos.

Varias compañías telefónicas estuvieron muy interesadas en brindar el correo electrónico como un servicio normal a suscriptores particulares y compañías. Para evitar un caos mundial, en 1984 la CCITT definió una serie de protocolos a los que llamó MHS(Sistemas de tratamiento de mensajes) en sus recomendaciones de la serie X400. La I.S.O trató de incorporarlos en la capa de aplicación del modelo OSI con el nombre de MOTIS (sistemas de intercambio de textos orientados a mensaje), aunque esta incorporación no es directa debido a la falta de estructura de la X.400. Sin embargo en 1988 el C.C.I.T.T modificó el X.400, con el objeto de hacerla compatible con MOTIS.

El correo electronico, también conocido como e−mail, ha estado presente en las ultimas dos decadas. Los primeros sistemas de correo electronico consistieron de protocolos de transferencias de archivos, con la convención que la primera liniea de mensaje tuviera la dirección del receptor. Con el paso del tiempo las limitaciones de este planteamiento fueron aun más notorias. Algunas de estas son:

1.− Resultaba engorroso transmitir un mensaje a muchas personas.

2.− Los mensajes no tenian una estructura interna lo que hacia dificil su

procesamiento, por el Computador.

3.− El extremo emisor nunca podia saber si el mensaje efectivamente llegó o

no.

4.− Si alguien estaba ausente y deseará que otra persona administrará su

correo, no era viable.

5.−El usuario debia crear el archivo,y luego llamar al sistema emisor para

enviarlo.

6.− Era imposible transmitir mensajes que contubieran una mescla de

distintos formatos(texto, dibujo, facsimiles)

A medida que se ganó experiencia con estos sistemas, se hicieron nuevas propuestas para llegar a tener sistemas de correo electrónico más ambiciosos (Bauerfeld y colaboradores, 1984; Horak, 1984; Rose y colaboradores, 1985). En 1984 el CCIT diseño su recomendación X.400, que después se tomó como base para la creación del MOTIS de la OSI. En 1988, el CCITT modificó el X.400 para alinearlo con el MOTIS. A diferencia de FTMA, por ejemplo, que se percibe extensamente como algo difícil de manejar, y que quizás

nunca reemplazará a los sistemas de archivos nativos, el MOTIS/X.400 se está convirtiendo rápidamente en la forma dominante para todos los sistemas de correo electrónico. Por esta razón, centraremos nuestra discusión sobre el correo electrónico en esta recomendación.

ARQUITECTURA Y SERVICIOS DEL MOTIS Y X.400

En esta sección proporcionaremos un resumen sobre lo que los sistemas de correo electrónico, especialmente MOTIS y X.400, pueden hacer y la manera como están organizados. Por razones de brevedad, nos referiremos a ellos de acuerdo con el nombre especificado por la OSI, MOTIS (ISO 10021), pero todo lo de esta sección también es aplicable al X.400. El MOTIS se ocupa de todos los aspectos del sistema de correo electrónico, comenzando desde el instante en el que el extremo de origen decide escribir un mensaje, y terminando en el instante en que el extremo receptor lo tira al bote de la basura. Enseguida describiremos brevemente seis de los aspectos básicos de cualquier sistema de correo electrónico.

***COMPOSICIÓN** se refiere al proceso de crear mensajes y respuestas. Aunque cualquier editor de texto puede utilizarse como generar el cuerpo del mensaje, el mismo sistema puede proporcionar asistencia con el direccionamiento y la gran cantidad de cabeceras de cada uno de los mensajes.*

***TRANSFERENCIA** se refiere al movimiento de mensajes desde el origen hasta su recepción. En gran medida, este proceso requiere una interfase con la ACSE o la capa de presentación, para establecer la conexión necesarias, dar salida al mensaje y liberar la conexión. El sistema de correo deberá realizar esto en forma automática, sin molestar al usuario.*

***NOTIFICAR** tiene que ver con la capacidad de decir al origen lo que sucedió con el mensaje. ¿Fue entregado? ¿Fue rechazado? ¿Se perdió? Existen varias aplicaciones en las que la confirmación de la entrega es muy importante y puede incluso tener implicaciones legales, (Pues bien, Su Señoría, mi sistema de correo no es muy fiable, por lo que me imagino que la citación electrónica se perdió en algún lugar).*

*La **CONVERSIÓN** puede necesitarse para hacer el mensaje se exhiba en forma adecuada en el terminal o impresora del receptor. Esta característica es particularmente importante porque el mundo está lleno de dispositivos receptores y transmisores incompatibles.*

*El **FORMATEO** se relaciona con la forma de exhibición del mensaje sobre el terminal del receptor*

*La **DISPOSICIÓN** es el paso final y está relacionado con lo que hace el receptor con el mensaje después de recibirlo. Entre las posibilidades se incluye la de desecharlo inmediatamente, o leerlo en primer lugar y después desecharlo, leerlo y después guardarlo, etcétera. También debería haber una manera de recuperar los mensajes que se guarden para releerlos, reexpedirlos o procesarlos de cualquier otra manera.*

Además de estos servicios básicos, la mayoría de los sistemas de correos proporcionan una gran variedad de características avanzadas. Ahora se comentarán brevemente algunas de ellas. Cuando la gente cambia de lugar, o se encuentra lejos correo, por lo tanto, el sistema deberá ser capaz de hacer esto en forma automática.

Por otra parte, si alguien decidiera en un momento dada tomas unas vacaciones en Tahíti para olvidarse de todo durante algunas semanas, podría no desear que le reexpidan su correo. En lugar de esto, si desease que el sistema de correo enviara una respuesta estereotipada al originador de cada mensaje que llegara indicando que se encuentra fuera y cuándo regresará. El sistema de correo, sin embargo, deberá mantener un seguimiento de las personas a quienes se les envió una respuesta, para no repetirla por segunda vez, aun cuando llegara a presentarse un segundo mensaje procedente del mismo originador.

*La mayoría de los sistemas de correo le permiten a los usuarios crear **BUZONES** para guardar el correo que*

les llega. Se necesitan comandos para crear y destruir buzones, para revisar el contenido de los buzones, para insertar y eliminar mensajes de los buzones, etcétera.

Con frecuencia, los ejecutivos de compañías necesitan enviar un mensaje a cada uno de sus subordinados, clientes o proveedores. Esto da lugar a la idea de una ***LISTA DE DISTRIBUCIÓN****, que viene a ser una lista de direcciones de correo electrónico. Cuando un mensaje se envía a la lista de distribución, a todos los usuarios que se encuentran en ella se les entrega una copia idéntica.*

El correo certificado es otra idea importante, para permitirle al extremo de origen saber que su mensaje efectivamente llegó a su destino. Alternativamente, puede ser deseable tener una notificación automática del correo, indicando que no es posible entregarlo. En cualquier caso, el originador, debería tener algún tipo de control sobre la manera de notificar lo que haya sucedido.

Otras características avanzadas es los destinatarios con copias, el correo de alta prioridad, el correo secreto (puesto en clave), los receptores alternativos, en caso de no encontrarse disponible el destinatario original y la habilidad para que las secretarias puedan manejar el correo de su jefe.

Una idea clave en todo los sistemas de correo electrónico modernos es la distinción entre el sobre y su contenido. El sobre encierra el mensaje, el sobre contiene los parámetros necesarios para transportar e interpretar los mensajes.

Por lo general, los sistemas de correo distinguen tres tipos de mensajes:

- *Mensajes de usuarios contienen información que se transmite desde un usuario hasta otro. Estos son los mensajes más importantes transportado por el sistema de correo, y representan la razón de su existencia. Pueden ser arbitrariamente extensos y pueden contener cualquier cosa que el originador desee incluir.*
- *Notificaciones son mensajes especiales de pruebas constituidos por sobres vacíos. El propósito de enviar una sonda es el averiguar si se puede alcanzar al destinatario. Además, el emisor también puede determinar la ruta seguida por el mensaje, así como el tiempo que tomó cada uno de los saltos, a partir de los sellos indicativos del tiempo y localidad colocados por todas las máquinas intermedias a lo largo de la trayectoria.*

Los ordenadores personales tienen una capacidad de almacenamiento limitada y, además, están conectados al agente de transferencia de mensajes durante una pequeña fracción del día. Esto da lugar a la pregunta de qué hacer con los mensajes grandes, o con los mensajes que llegan, cuando el agente de usuario no está conectado al ordenador. La solución consiste en hacer que el agente de transferencia de mensajes mantenga buzones electrónicos, para cada usuario, dentro de un área llamada el almacén de mensajes. Los mensajes de entrada podrán ser colocados en estos buzones hasta que el usuario se conecte para leerlos, eliminarlos, moverlos a otros buzones, cargarlos selectivamente, etcétera.

Si el agente de usuario y el agente de transferencia de mensajes operaran en máquinas separadas, la interacción entre ellos, P3, se convierte en objeto de normalización. OSI ha definido normas del servicio y del protocolo para esta interacción P3. De manera similar, la interacción con el almacén de mensajes, P7, también ha sido normalizada, al igual que P1, el protocolo situado entre los agentes de transferencia de mensajes.

A la colección de todos los agentes de transferencia de mensajes se le llama sistema de transferencia de mensajes. A este sistema se le subdivide en dominios administrativos. Cada dominio es operado por una autoridad distinta. Algunos dominios son públicos y operados por un operados común o las PTT y otros dominios son privados, totalmente localizados dentro de una sola compañía.

El protocolo (virtual) entre los agentes de usuario, P2, también ha sido normalizado. En efecto, P2, queda

definido por la cabecera y el cuerpo de la figura 9−10, en tanto que P1 está relacionado con el sobre. A P2 se le conoce como IPM (mensajería interpersonal) y se diseñó para manejar mensajes entre dos personas. A medida que pasa el tiempo, no hay duda de que se definirán otros protocolos para hacer pedidos de productos, para expedir facturas y para muchas otras actividades.

El agente usuario

El sistema de correo electrónico está dividido en dos partes básicas, como ya hemos visto; los agentes de usuarios y los agentes de transferencia de mensajes. En esta sección estudiaremos a los agentes de usuarios y, en la siguiente, estudiaremos a los agentes de transferencia de mensajes. Maneja el diálogo con el usuario, en el terminal; se comunica con el sistema de transferencia de mensajes, con la aceptación y entrega de mensajes y; se relaciona con el almacén de mensajes.

Examinemos primero a la interfase de usuario. Típicamente, al agente de usuario se le invoca, llamando a un programa que acepta una gran variedad de comandos relacionados con la composición, el envío y la recepción de mensajes y con el manejo de buzones. Algunos programas de correo electrónico tienen presentaciones de menú estilizado o manejados por iconos, que se utilizan con un ratón, en tanto que otros poseen una lista de comandos de teclado de 1 carácter, para proporcionar las distintas funciones. Típicamente, el programa de correo explorará al buzón del usuario, para localizar el correo de entrada, antes de visualizar cualquier información en la pantalla. Después, anunciará la cantidad de mensajes que se encuentran en el buzón y esperará un comando.

Cada línea de la imagen en pantalla consta de varios campos extraídos del sobre, o de la cabecera del mensaje correspondiente. En un sistema de correo sencillo, los campos visualizados en la pantalla están programados de manera fija dentro del propio programa de correo. En un sistema más sofisticado, el usuario puede especificar qué campos quiere que se visualicen en la pantalla. Proporcionando un ***perfil de usuario,*** *que es un archivo que describe el formato de la pantalla. El primer campo corresponde al número del mensaje. El segundo campo trabaja con banderas para indicar si el mensaje no es nuevo, sino se leyó previamente y se guardó en el buzón, si fue contestado, o si fue retransmitido a otra persona. El tercer campo indica la longitud del mensaje y el cuarto indica quien envió el mensaje. Dado que este campo es simplemente extraído a partir del mensaje, podrá contener nombres, apellidos completos, iniciales, nombres de registro, o cualquier cosa que se le ocurra poner al emisor del mensaje. Finalmente, el campo de* ***asunto*** *proporciona una breve descripción del contenido del mensaje. La gente que omite incluir el campo Asunto, descubre pronto que la respuesta a su correspondencia no está obteniendo la mayor prioridad.*

*El siguiente grupo de tres comandos se ocupa del envío de mensajes, más bien que de su recepción. El comando "****s****" envía un mensaje, llamando a un editor apropiado que le permita al usuario hacer la composición del mensaje. Analizadores ortográficos, gramaticales y de dicción pueden utilizarse para detectar si la sintaxis del mensaje es la correcta. Desafortunadamente, la generación actual de programas de correo electrónico no tiene analizadores para detectar si el emisor sabe de qué está hablando. Cuando se termina el mensaje, queda preparado para ser transmitido al agente de transferencia de mensajes.*

*El comando "****f****" retransmite un mensaje del buzón, exigiendo una dirección para poderlo enviar. El comando "****a****" extrae la dirección fuente, del mensaje que debe ser contestado, y llama al editor para permitirle al usuario hacer la composición de su respuesta.*

*El siguiente grupo de comandos sirve para manejar los buzones. Los usuarios tienen típicamente un buzón para cada persona con la que tienen correspondencia, además del buzón de entrada de correspondencia, que acabamos de ver. El comando "****d****" elimina un mensaje del buzón, pero el comando "****u****" desactiva la eliminación (El mensaje no se elimina realmente hasta que el programa de correo acaba). El comando "****m****" mueve el mensaje a otro buzón, lo cual es la manera usual de conservar la correspondencia importante, después de haberla leído. El comando "****k****" conserva el mensaje indicado dentro del buzón, aun después de*

*que se haya leído: Cuando se lee un mensaje y, explícitamente no es conservado, se toma una acción por omisión (default) cuando el programa de correo acaba. Los comandos "**n**", "**b**" y "**g**" sirven para moverse dentro del buzón en curso. Es muy común que un usuario lea el mensaje 1, lo conteste, lo mueva, lo elimine y, después teclee una "**n**", para obtener el siguiente mensaje. La ventaja de este comando es que el usuario no tiene la necesidad de seguir la pista de dónde se encuentra este momento. Es posible ir hacia atrás, utilizando el comando "**b**", o apuntar a un mensaje dado, utilizando "**g**".*

*Finalmente, el comando "**e**" deja el programa de correo y realiza todos los cambios que se necesiten hacer, como eliminar algunos mensajes y marcar a otros con la bandera k. Este comando escribe sobre el contenido del buzón, actualizándolo.*

Pasemos ahora de la interfase del usuario al protocolo utilizando entre dos agentes de usuario En gran medida, este protocolo está definido por los campos de cabecera, que se incluyen en cada mensaje.

El campo originador contiene el nombre de la persona que realmente envió el mensaje. En campo de usuarios que autorizan indica la persona que autorizó el envío del mensaje. En la mayoría de los casos, estos dos campos serán idénticos, pero podrían ser diferentes si una secretaria envía un mensaje por cuenta del jefe.

Los siguientes tres campos determinan quienes son las personas que pueden recibir copias del mensaje. Los receptores principales representan a la gente a la cual está realmente destinado. A los receptores con copia se les envían copias del mensaje, frecuentemente con propósitos administrativos. Los receptores con copia ocultos, también obtienen copias pero sus nombres no se incluyen en las copias que se envían a los demás. Por lo tanto, los receptores principales llegan a saber quien recibió copias, pero no quien recibió las copias ocultas.

El siguiente grupo de campo se refiere a la respuesta. La respuesta no tiene que ir a parar necesariamente a la persona que envió el mensaje, y podrá existir un plazo de tiempo en el que es requerida.

Después viene un grupo que contiene identificadores de mensajes: de este mensaje, del mensaje al que se contesta, de mensajes previos reemplazados por el actual, y otros tipos.

El campo de asunto proporciona un resumen del mensaje.

El campo de importancia puede especificar la urgencia del mensaje o cualquier otro tipo de prioridad.

El campo de sensibilidad está relacionado con el carácter secreto que puede tener el contenido y con el deseo del emisor de revelarlo o no a terceras personas.

Finalmente el campo de fecha de expiración indicará durante cuanto tiempo es válido el mensaje.

Aunque el cuerpo del mensaje no esté tan fuertemente estructurado como la cabecera, sin embargo, si tiene una estructura. El cuerpo consiste de una parte principal y, opcionalmente, de uno o dos agregado. En el futuro, mucha gente recibirá correo electrónico en el hogar, a través de videotex, utilizando sus televisores como pantalla de visualización. Por lo general los televisores solamente tienen una resolución suficiente para cuarenta caracteres de texto por línea, por lo que aparecerá un problema cuando el emisor considere que una línea tiene ochenta caracteres.

Una posible solución consiste en codificar el cuerpo del texto como unn documento de formato simple normalizado. En este documento, el texto se descompone en párrafos, con líneas en blanco entre cada párrafo. El agente del usuario del extremo receptor visualiza entonces la cantidad de palabras por líneas que puedan caber, conservando solamente los limites del parrafo y no los limite de la línea.

Una solución más sofisticada consiste en hacer que todos los mensajes contengan comandos normalizados de composición de texto. Con estola gente de usuario podrá entonces visualizar en pantalla un texto sofisticado, incluyendo fórmulas matemáticas, tablas dibujos lineales de manera adecuada para que los utilicen los terminales receptores olas impresoras.

El agente de transferencia de mensajes

Este sistema se ocupa de transferir el mensaje desde el origen hasta el receptor. Es fundamentalmente distinto del proceso de establecer una asociación el la capa de aplicación, como la realiza FTAM. Con este último no se puede leer un archivo remoto si el servidor de archivo esta actualmente desactivado. Con el sistema de correo se le puede enviar un mensaje a alguien incluso cuando esta este de vacaciones o su agente de usuario no este actualmente registrado. El correo electrónico esta, por lo tanto, modelado de acuerdo a un sistema de almacenamiento y reenvío que opera de salto en salto y no de extremo a extremo. Cuando llega un mensaje al ATM, el proceso se desarrolla de la siguiente forma: si el mensaje procede de un agente de usuario, se coteja la validez de la sintaxis y, si esta aparece inválida se devuelve el mensaje con una explicación. Si la sintaxis es válida se adhiere al mensaje un identificador de mensaje y un sello de tiempo, y luego se le da el mismo tratamiento que aun mensaje que viene de otro ATM.

El siguiente paso consiste en averiguar si el agente de usuario receptor o el buzón receptor son locales. Si así fuera, se podrá entregar el mensaje o ponerlo en cola de espera para su entrega, o bien almacenarlo en el buzón. Cuando se requiera, se puede generar una notificación para confirmar la entrega y enviarlo de regreso. Si el receptor no es local, el mensaje se retransmite a otro ATM

En la mayoría de los sistemas se inserta en el sobre una relación de los ATM que han estado manejando el mensaje. Esto no solo facilita seguir la pista de los problemas que hallan ocurrido, sino que también permite detectar, si ha habido lazos. Si un ATM recibe un mensaje en cuya relación se encuentre el mismo, sabrá que el mensaje esta dando vueltas en un lazo y tendrá que aplicar medidas especiales para romperlo.

Realizar una entrega a un agente de usuario local no es siempre algo trivial, dado que el emisor y el receptor pueden tener distintos equipos. Algunos de los tipos de mensajes son los siguientes:

1.− Texto ASCII

2.− Facsímil analógico.

3.− Facsímil digital.

4.− Voz digitalizada.

5.− VIDEOTEX.

6.− Telex.

7.− Externo (algún otro sistema).

Si el receptor esta incapacitado para aceptar directamente el tipo de mensaje el ATM puede intentar hacer una conversión antes de entregarlo. No todas las conversiones se pueden realizar, como convertir la voz digitalizada al ASCII. Si la conversión no puede efectuarse, no puede entregarse el mensaje.

Pasarán muchos años antes que la gente posea el equipo requerido para recibir el correo electrónico. Mientras tanto las conversiones a otros sistemas serán muy necesarias.

Aun cuando la ISO no haya normalizado todos los detalles de la operación de almacenamiento y reenvío entre 2 ATM, sin embargo, ha adoptado la estructura general del CCITT para lo que se ha llamado el servicio de operaciones remotas. Esto no es nada distinto de nuestra y vieja conocida, la llamada de procedimiento remoto, sólo que, ahora, sepultada dentro del sistema de correo. Cuatro operaciones han sido normalizadas:

1.− Invocar una operación remota en otro ordenador.

2.− Devolver el resultado de una operación invocada remotamente, a la persona que llama.

3.− Devolver un mensaje de error a la persona que invoca.

4.− Devolver la llamada a una operación remota, por ser invalida.

El servicio de operaciones remotas podría ser utilizado para introducir un mensaje en la cola de espera de entrada de otro ATM.

Uno de los aspectos clave para el diseño de los ATM es el del direccionamiento. Por ejemplo, el sistema NAME@DOMAIN es el utilizado por ARPANET, la interconexión de redes ARPA, USANET y otras redes. Los dominios pueden tener subdomínios THOMPSON@CS.YALE.EDU se refiere al Thompson del departamento de ciencias de los ordenadores en la universidad de Yale, y THOMPSON@CHEM.YALE.EDU se refiere al Thompson del departamento de química.

TERMINALES VITUALES

Los terminales se distribuyen en tres clases principales:

- *Los de modo deslizamiento.*
- *Los de modo página.*
- *Los de modo formulario.*

Cada categoría tiene sus propios problemas que conducen a distintas soluciones.

Terminales del modo deslizamiento (TMD)

Estos carecen de procesadores internos y de cualquier clase de editor. Cuando se aprieta una tecla en el terminal 1, el carácter respectivo se envía a través de la línea al terminal 2 el que muestra el carácter enviado en la pantalla y 0 a medida que van apareciendo nuevas líneas, las que ya fueron mostradas en pantalla se deslizan una línea más arriba. La mayoría de los terminales de salida impresa, como los CRT, pertenecen a este modo.

A pesar de su aparente sencillez, los terminales MD son diferentes en varias características tales como la longitud de la líneas, el juego de caracteres que usa, la presencia o ausencia de eco automático y de sobreimpresión, así como también el retorno del carro, el salto de línea, la tabulación vertical y horizontal, el retroceso y el salto de página.

Debido a que este tipo de terminales carecen de cualquier tipo de procesamiento y no tienen posibilidad de comunicarse con la red, a través de algunos de los protocolos normalizados de red que existen; los usuarios adquieren normalmente una "CAJA NEGRA", conocida como PAD (ensamblador−desensamblador de paquetes), que se intercala entre el terminal y la red. Este PAD se comunica con el terminal mediante el RS−232, y con la red mediante un protocolo normalizado.

El PAD

El CCITT ha definido las interfases normalizadas de un PAD en sus recomendaciones X.3, X.28 y X.29. La X.3 define los parámetros del PAD; la X.28 define la interfase entre el terminal y el PAD; y la X.29 define la interfase entre el PAD y el ordenador.

El PAD no es exactamente un terminal, pero tampoco encaja bien en ninguna otra parte de la jerarquía de la OSI.

Al primer contacto entre el terminal y el PAD, el usuario o el ordenador fijan ciertos parámetros que describen la conversación entre el terminal y el PAD. Estos parámetros están numerados, así como sus valores opcionales. Un comando típico hacia el PAD es el siguiente: ***SET 1:0,*** *que significa, fija el parámetro 1 en un valor 0. También existen otros parámetros para establecer y suprimir conexiones, para leer los valores de los parámetros, para restablecer la línea y para forzar una interrupción.*

En el siguiente cuadro se muestra una lista completa de parámetros.

1	*¿Podrá el operador del terminal salir del modo de transferencia de datos para inspeccionar o cambiar los parámetros del PAD?*	***0*** *= NO (salida prohibida)* ***1*** *= SI (salida permitida)*
2	*¿Deberá el PAD devolver caracteres (eco) al terminal?*	***0*** *= NO* ***1*** *= SI*
3	*¿Cuáles caracteres podrán hacer que el PAD envíe un paquete parcialmente lleno?*	***0*** *=Envía sólo paquetes llenos.* ***1****= Retorno de carro.* ***126*** *= Todos los caracteres de control + DEL.*
4	*¿Cada cuanto tiempo deberá el PAD enviar un paquete parcialmente lleno si el terminal permanece inactivo?*	***0*** *=No existe una temporización* ***1−225****=Tiempo en tics de 50 ms*
5	*¿Puede el PAD (temporalmente) prohibirle al terminal que envíe caracteres?*	***0****=NO(modo simple)* ***1****=SI(control de flujo)*
6	*¿Tiene el PAD permitido enviar señales de servicio al terminal?*	***0****=Suprime las señales* ***1****=SI(entrega las señales)*
7	*¿Qué deberá hacer el PAD al recibir una señal de suspención desde el terminal?*	***0****=Nada* ***1****=Interrumpir* ***2****=Reiniciar* ***4****=Envier un paquete de control al hostal* ***8****=Salir al modo de comando* ***16****=Desechar salida*
8		***0****=NO (entregar)*

	¿Deberá el PAD desechar información de salida del ordenador, destinada al terminal?	***1**=SI (desechar)*
9	*¿Cuántos caracteres de relleno deberá insertar el PAD después de enviarle un retorno de carro al terminal?*	***0**=Ninguno* ***1−7**=Número de caracteres de relleno*
10	*¿Deberá el PAD ajustar automáticamente la salida para impedir el desbordamiento de línea?*	***0**=NO* ***1−225**=SI(longitud de la línea)*
11	*Velocidad del terminal (bps)*	***0**=110 **8**=200* ***1**=134.5 **9**=100* ***2**=300 **10**=50*
12	*¿Puede el terminal prohibirle (temporalmente) al PAD que le envíe información de salida?*	***0**=NO* ***1**=SI*
13	*¿Deberá el PAD insertar un salto de línea, después de un retorno de carro?*	***0**=NO* ***1−7**=Varias condiciones*
14	*¿Deberá el PAD añadir relleno después de los saltos de línea?*	***0**=NO* ***1−7**=SI (cuántos)*
15	*Permite el PAD la edición*	***0**=NO* ***1**=SI*
16	*Selecciona la tecla para borrar un caracter*	***0−127**= carácter*
17	*Selecciona una tecla para borrar una línea*	***0−127**= carácter*
18	*Selecciona la tecla para visualizar la línea actual*	***0−127**= caracter*

Terminales del modo página (TMP)

Son los típicos CRT que pueden visualizar 25 renglones de 80 caracteres cada uno. Todos estos terminales tienen los mismos problemas que los TMD, además de otros como son: la longitud de página. El direccionamiento del cursor, y la presencia o ausencia de parpadeo, de video inverso, de color y de múltiples intensidades

Ningún terminal tiene las mismas secuencias de escape para poder utilizar todas las características sofisticadas. Con este contexto se demuestra que los editores de pantalla son bastante difíciles. Una solución ampliamente utilizada consiste en definir un terminal virtual, que posea los comandos que tiene la mayoría de los terminales de modo página.

En el momento que un editor inicia su operación, averigua cuál es el tipo de terminal, y después leer la entrada del terminal, de una base de datos llamada TERMCAP(capacidad del terminal) que proporciona las secuencias de escape necesarias para cada comando virtual. Siempre que el software se restrinja a producir comandos del terminal virtual, podrá correr sobre cualquier terminal que posea una entrada en TERMCAP.

Los siguientes son algunos de los códigos comúnmente utilizados por TERMCAP.

bs	*booleano*	*El terminal puede retroceder un espacio con CTRL+H*

hc	*Booleano*	*El terminal imprime copia de papel*
nc	*Booleano*	*El terminal es un CRT pero no puede deslizar la pantalla.*
os	*Booleano*	*El terminal superponer (múltiples caracteres en la misma posición)*
ul	*Booleano*	*El terminal puede subrayar*
pt	*booleano*	*El terminal tiene tabuladores en hardware*

co	*Entero*	*Número de columnas en una línea.*
li	*Entero*	*Número de líneas en una pantalla o en una página.*
dB	*Entero*	*Número de milisegundos de retardo necesarios para un espacio hacia atrás.*
dC	*Entero*	*Número de milisegundos de retardo necesarios para un retorno de carro.*
dN	*Entero*	*Número de milisegundos de retardo necesarios para un salto de línea.*
dT	*Entero*	*Número de milisegundos de retardo necesarios para un tabulador.*

cd	*Cad. De caracteres*	*Borra la pantalla desde el cursor hasta el final.*
ce	*Cad. De caracteres*	*Borra la pantalla desde el cursor hasta el fin de la línea*
cl	*Cad. De caracteres*	*Borra toda la pantalla*
cm	*Cad. De caracteres*	*Movimiento del cursor.*
ct	*Cad. De caracteres*	*Borra todos los topes de tabulador.*
dc	*Cad. De caracteres*	*Borra el carácter bajo el cursor.*
dl	*Cad. De caracteres*	*Borra la línea que tiene el cursor.*
ff	*Cad. De caracteres*	*Pasa a la parte superior de la siguiente página (terminales de salida impresa)*
ic	*Cad. De caracteres*	*Inserta un carácter sobre el cursor.*
is	*Cad. De caracteres*	*Cadena utilizada para iniciar terminal.*
mb	*Cad. De caracteres*	*Entra al modo de parpadeo.*
md	*Cad. De caracteres*	*Entra al modo destacado (extra luminoso).*
us	*Cad. De caracteres*	*Entra al modo de subrayado.*
mh	*Cad. De caracteres*	*Entra al modo difuso (medio luminoso)*
so	*Cad. De caracteres*	*Entra al modo de video inverso.*
me	*Cad. De caracteres*	*Entra al modo normal (sin parpadeo, ni destacado, ni difuso, ni imagen invertida, etc.)*
pc	*Cad. De caracteres*	*Carácter utilizado para rellenar después de un tabulador, salto de página, etc.*
sf	*Cad. De caracteres*	*Desliza la pantalla hacia delante.*
sr	*Cad. De caracteres*	*Desliza la pantalla hacia atrás.*

kl	*Cad. De caracteres*	*Secuencia de escape enviada por la tecla de flecha izquierda.*
kr	*Cad. De caracteres*	*Secuencia de escape enviada por la tecla de flecha derecha.*
ku	*Cad. De caracteres*	*Secuencia de escape enviada por la tecla de flecha arriba.*
kd	*Cad. De caracteres*	*Secuencia de escape enviada por la tecla de flecha abajo.*
kh	*Cad. De caracteres*	*Secuencia de escape enviada por la tecla de inicio de pantalla*
kb	*Cad. De caracteres*	*Secuencia de escape enviada por la tecla de espacio hacia atrás.*

Terminales del modo formulario

Este tipo de terminales es utilizado con aplicaciones donde el ordenador está capacitado para presentar un formulario en la pantalla del terminal, con algunos campos de información que sólo se pueden leer y otros para ser llenados por el usuario mediante el teclado. El microprocesador de este puede suministrar una edición local y algunas otras características.

Después de llenar el formulario el microprocesador es capaz de correr un programa que verifique rápidamente la sintaxis. Si este formulario es sintácticamente correcto, se podrá cargar la opción modificada hacia el ordenador por medio de la red.

Aquí la TERMCAP no es lo más adecuado para manejar este tipo de terminales, pero si existe uno alternativo que se basa en la utilización de estructuras de datos compartidas. En este modelo, el software del terminal virtual conserva internamente una representación abstracta de la imagen de la pantalla que puede ser leída y actualizada tanto por el usuario como por el programa de aplicación que opera en alguna parte de la red. El software del terminal virtual esta encargado de garantizar que sea actualizada la imagen de la pantalla, cada vez que se modifica su representación abstracta.

Este modelo tiene dos variantes principales: la primera es la "síncrona" en donde existe una sola estructura de datos abstracta que representa la imagen de la pantalla, de la cual se mantienen 2 copias idénticas; una de ellas mediante el software del terminal virtual que corre sobre el microprocesador del terminal, y la otra mediante el software del terminal que corre cerca del programa de aplicación, en un hostal remoto. El usuario que está en terminal puede modificar mediante el teclado, la copia de la estructura de datos abstracta que se encuentra en el microprocesador. Esta modificación se puede ver en la pantalla. Pero para impedir que dos extremos traten de modificar la estructura de datos simultáneamente, se utiliza un testigo, para controlar el acceso de actualización. Tanto el ordenador como el terminal pueden tener el testigo, pero sólo aquel que lo posea podrá actualizar la estructura de datos.

El trabajo del testigo es muy similar al empleado en la capa de sesión.

Por la respuesta desfavorable de los usuarios con el modelo síncrono, nació el modelo "asíncrono" que está constituido por 2 monólogos independientes, en vez de un solo diálogo, así cada extremo de la conexión posee una estructura de datos de entrada y una segunda para la salida. Aquí la estructura de datos puede ser escrita por un solo lector y un solo escritor. Su costo es el almacenamiento adicional y su complejidad.

Terminales virtuales con una estructura de datos compartida

Modelo Síncrono

Pantalla Copia de la estructura de datos

compartida

Hostal Remoto

6 5

2

3 4

1

Red

Teclado Microprocesador Programa de

dentro del terminal aplicación

Modelo Asíncrono

Pantalla Copia de la estructura de datos

De salida

Hostal Remoto

6 5

4

3

1

2

Teclado Estructura de datos Estructura de datos

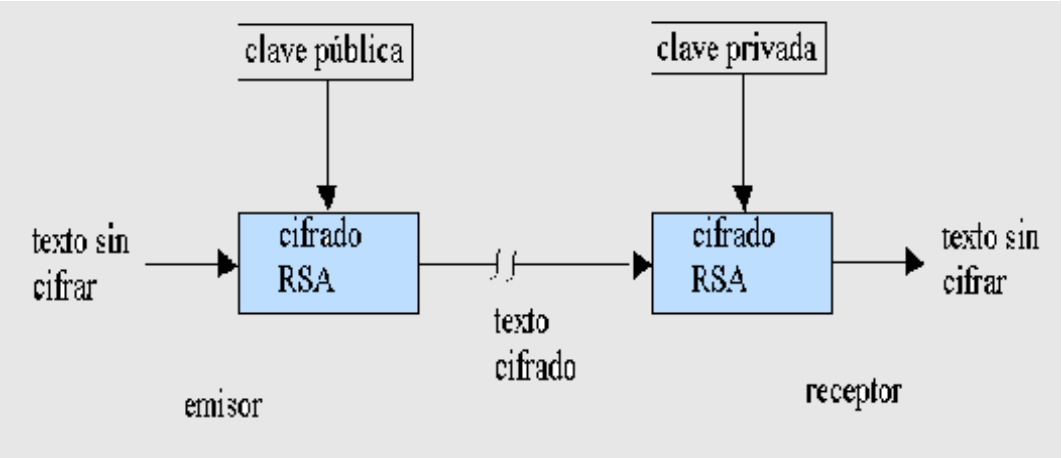
de entrada de entrada

CONCLUSIÓN

Hemos visto en lo extenso del presente trabajo la variedad de servicios para un sin numero de aplicaciones en la implementación de una red Modelo OSI. Pero no tan solo de ofrecer servicios, sino también para administrarlos eficientemente.

Si bien, los diferentes temas que se han descrito proporcionan una acabada explicación de los métodos de control, que se necesitan para normalizar y optimizar los servicios, del modelo OSI, es de suma importancia considerar que cada uno de los puntos expuestos tiene una relevancia, tanto técnica como usuaria, para comprender a cabalidad que un servicio de red necesita de las siguientes implementaciones:

- *Detectar y Controlar los posibles errores.*
- *Establecer los caminos que la información recorrerá(emisor/receptor)*
- *Disminuir los costos, aumentar la velocidad y la capacidad en la transmisión.*
- *Establecer los parámetros de seguridad de la información para que esta no permita infiltraciones, hurtos o daño en la transmisión de datos, de parte de los llamados Hackers y/o Crackers.*
- *Identificar y controlar que los mensajes enviados por correo electrónico manifiesten los patrones mínimos de formato y verificación en el envío y recepción de los mismos.*
- *Coordinar mediante el establecimiento de un terminal virtual el trabajo entre dos terminales diferentes.*


clave pública
clave privada
texto sin cifrar
cifrado RSA
cifrado RSA
texto sin cifrar
texto cifrado
emisor
receptor