Trabajo de Grado

SEGURIDAD Y EVALUACIÓN DE PERFORMANCE EN REDES LOCALES

Director:

ARMANDO EDUARDO DE GIUSTI

Integrantes:

BARAC, ADRIANA

CORALLINI, GUSTAVO

MARANO, FERNANDA

SEGURIDAD Y EVALUACIÓN DE PERFORMANCE EN REDES LOCALES

Indice General

OBJETIVOS GENERALES

El notorio desarrollo de los sistemas distribuidos y en particular las redes locales resalta la importancia profesional y académica que reviste el estudio de diferentes aspectos de estas arquitecturas.

Nuestro objetivo fue desarrollar una herramienta que mejore la gestión de supervisión de un sistema de red bajo el sistema operativo NetWare, dada su gran difusión en el mercado.

Para esto se estudiaron, en primer lugar, los aspectos de seguridad en redes, referidos tanto a protección y supervisión de operaciones a nivel sistema operativo de la red, como a una protección en la comunicación entre usuarios distribuidos. En segundo lugar se estudiaron las características de la red que son útiles en la evaluación de performance.

Como resultado se desarrollo un sistema para NetWare v 2.11 y posteriores que permite controlar la seguridad de la red a nivel de conexiones y detección de intrusos, como también controlar la utilización de los recursos.

I INTRODUCCION A LOS SISTEMAS DISTRIBUIDOS

Contenido Páginas

INTRODUCCION A LOS SISTEMAS DISTRIBUIDOS

1 DEFINICION

Las siguientes definiciones expresan en forma concreta a que se denomina sistemas distribuidos:

Definición de Liebowitz y Carson

"Un Sistema Distribuido es un sistema de computación en el cual las funciones computacionales son alocadas entre varios elementos físicos de computación. Estos pueden estar geográficamente próximos o separados unos de otros."

Tanenbaum

"Una red de computadoras significa una colección de computadoras autónomas e interconectadas. Dos computadoras están interconectadas cuando son capaces de intercambiar información, que puede ocurrir a través de un medio físico de comunicación. Se compone por ejemplo, de alambre de cobre, micro onda, fibra óptica o satélites. El requisito de autonomía excluye los sistemas en los cuales existe una clara relación maestro esclavo entre las computadoras. "

Enslow

1- "Una multiplicidad de recursos computacionales de uso general incluyendo recursos físicos y lógicos, que pueden ser dinámicamente encargados de la ejecución de tareas específicas. La homogeneidad de recursos físicos no es esencial ";

2- "Una distribución geográfica de esos componentes físicos y lógicos, los cuales interactúan a través de una subred de comunicación de datos";

3- " Un sistema operacional de alto nivel que verifique e integre el control de los componentes distribuidos. Cada procesador individual puede tener su propio sistema operacional";

4- " La transparencia del sistema que permite el requerimiento de servicios apenas por el nombre, sin identificación de quien será el ejecutor";

5- "Autonomía cooperativa, caracterizada por la operación e interacción de los recursos físicos y lógicos. Implica la no

existencia de jerarquías de control dentro del sistema".

Tomamos como definición de sistemas distribuidos a la extraída de Tanenbaum

2 CARACTERISTICAS DE LOS SISTEMAS DISTRIBUIDOS

Un sistema de computación distribuido presenta una serie de ventajas:

- » *Alto Desempeño.*
- » *Disponibilidad.*
- » *Compartir Recursos.*
- » *Alta Extensibilidad.*

Desempeño:

Es definido en términos de ejecución y tiempo de respuesta. La determinación del índice de desempeño está directamente relacionado con la aplicación. El mejor desempeño esperado en los sistemas distribuidos se basa en el hecho de que la relación costo/desempeño de las pequeñas computadoras es inferior al de las grandes.

Disponibilidad:

Es la probabilidad de que, en cualquier instante, un sistema este en funcionamiento.

Compartir Recursos:

Disponibilidad para cualquier usuario, sin importar su ubicación geográfica, compartir recursos como programas, datos, elementos de procesamientos y otros dispositivos físicos.

Extensibilidad:

También llamado de "crecimiento incremental", es la posibilidad de construcción de sistemas que pueden ser fácilmente adaptados a nuevos ambientes. La extensibilidad se refiere a la facilidad de sustitución de una función lógica o de un elemento de hardware.

2.1 Estructuras de Interconexión

Los diversos elementos de un sistema distribuido por estar geográficamente separados, necesitan de un medio físico para transportar información que intercambian entre si.
La información es transmitida en forma de señales eléctricas. El medio mas simple de transmisión es el par de hilos. Este incluye el par trenzado y el cable coaxil. La velocidad de transición de la información se mide en bits por segundos.

Otro tipo de medio utilizado para la transmisión es la fibra óptica. Presentan características de alta velocidad de transmisión e inmunidad a los ruidos.
Otro medio de transmisión es la propagación de ondas electromagnéticas, por radio o láser. Las ondas son difundidas en el espacio, y pueden ser captadas libremente por cualquier elemento del sistema distribuido.

Se puede identificar, en un sistema distribuido, dos subsistemas: el primero constituido por dos *elementos de computación* que son máquinas destinadas a la ejecución de programas del usuario; el segundo es una *subred de comunicación* o sistemas de interconexión, que conecta dos terminales y transporta información entre ellas.

El sistema de interconexión se compone de medios físicos de interconexión y de nodos. Los medios físicos son llamados, líneas de transmisión, circuitos y canales. Las terminales son conectadas a los nodos que a su vez son enlazados a través de un medio físico de transmisión.
Sistemas de interconexión:

- *Subred Punto a Punto.*
- *Subred de Difusión.*

Subred Punto a Punto:

Los nodos son enlazados por medio de hilos o fibras ópticas. Si dos nodos no están directamente conectados por el mismo cable, la transmisión debe ser hecha indirectamente a través de otros nodos intermediarios (transmisión nodo a nodo).

SubrRed de Difusión:

El canal de comunicación es compartido por todos los nodos. Uno de ellos envía un mensaje y todos los otros lo reciben. Debe haber un mecanismo de forma de indicar a cual o cuales nodos va dirigido el mensaje.

2.2 La Necesidad de Uso de Protocolos

Para promover un intercambio ordenado de información entre los elementos, se torna necesario establecer un conjunto de reglas y convenciones denominadas *protocolos*. Un protocolo es una abstracción lógica del proceso físico de comunicación.
Razones fundamentales:

1ª) Para establecer un patrón de datos.
2ª) Para establecer las convenciones necesarias.
3ª) Para establecer caminos de comunicación.

3 ARQUITECTURA DE LOS SISTEMAS DISTRIBUIDOS

Los sistemas distribuidos son de naturaleza compleja, tanto para la parte referente a las aplicaciones, como para la parte referente a la comunicación entre los componentes distribuidos y separados físicamente. Contribuyen para esta complejidad cuestiones como el cambio de información a distancia y el control de sincronización .

Los sistemas distribuidos usan la idea aplicada por la disciplina llamada "Ingeniería de Software" (busca de soluciones que sean estructuradas en módulos, los cuales deben presentar la mayor independencia posible entre ellos).
Sean dos o mas computadoras interconectadas, físicamente, y cada una de ellas organizada como una máquina compuesta de una serie de capas o niveles de abstracción. Estas van, desde el nivel de "hardware", pasando por los niveles de lenguaje de máquina, de sistema operacional, de lenguaje de montaje, de lenguajes de alto-nivel, hasta el nivel del lenguaje de computación.
Cada nivel de la jerarquía constituye una *capa de servicio*. En cada capa existe una *interface* con la capa superior y otra con la capa inferior. El objetivo de una capa N es proveer ciertos *servicios* para la capa N+1 y superiores. La capa N es inplementada basándose en los servicios ofrecidos por la capa N-1.

Una capa se constituye de componentes activos llamados *entidades*. Para ejecutar sus servicios, las entidades de una capa mantienen una conversación. Existen un conjunto de reglas y convenciones entre entidades de una capa que son colectivamente conocidas como *protocolo* de capa N.

Los servicios de la capa N ofrecidos a la capa N+1 y superiores son provistos a través de una *interface*. Esta define

cuales son los servicios ofrecidos y de que manera los mismos son alcanzados.

La transferencia de datos entre entidades de una misma capa, no se da directamente. Las informaciones son pasada a la capa inmediatamente inferior, hasta alcanzar la capa mas baja, donde se sitúa el medio físico de comunicación. Ahí ocurre la transmisión real de información entre sus entidades.

Al conjunto de capas, protocolos e interfases se les da el nombre de *arquitectura de los sistemas de interconexión*.

3.1 La Propuesta de Arquitectura de la ISO

Una de las principales ventajas de una estructura organizadas en capas es permitir que elementos de computación heterogéneos
se puedan comunicar.

La iniciativa más importante en este sentido partió de la ISO ("International Organization for Standarization"), un organismo internacional que congrega representantes institucionales de diversos países interesados en aspectos de patronización en general. La ISO elaboró un modelo básico de referencia de la arquitectura de sistema de interconexión referido como modelo OSI ("Open Systems Interconnection"). La finalidad del modelo de referencia OSI/ISO es proveer una base común para la coordinación en el deserrollo de patrones con el propósito de interconectar sistemas.

Arquitectura en Capas

El modelo propone una división en siete capas (ver Fig.1):

1- Capa de Aplicación:

Provee servicios para los procesos de los usuarios finales. En esta capa los usuarios tienen la libertad de definir sus propios protocolos. Los procesos de la aplicación son los consumidores finales y generadores primarios de la información.

2- Capa de Presentación:

Tiene como objetivo la ejecución de funciones frecuentemente requeridas y que por eso pueden ser resueltas en forma general. Un problema típico es la conversión de caracteres. La capa de presentación se ocupa de los aspectos de sintaxis y semántica de la información que se transmite.

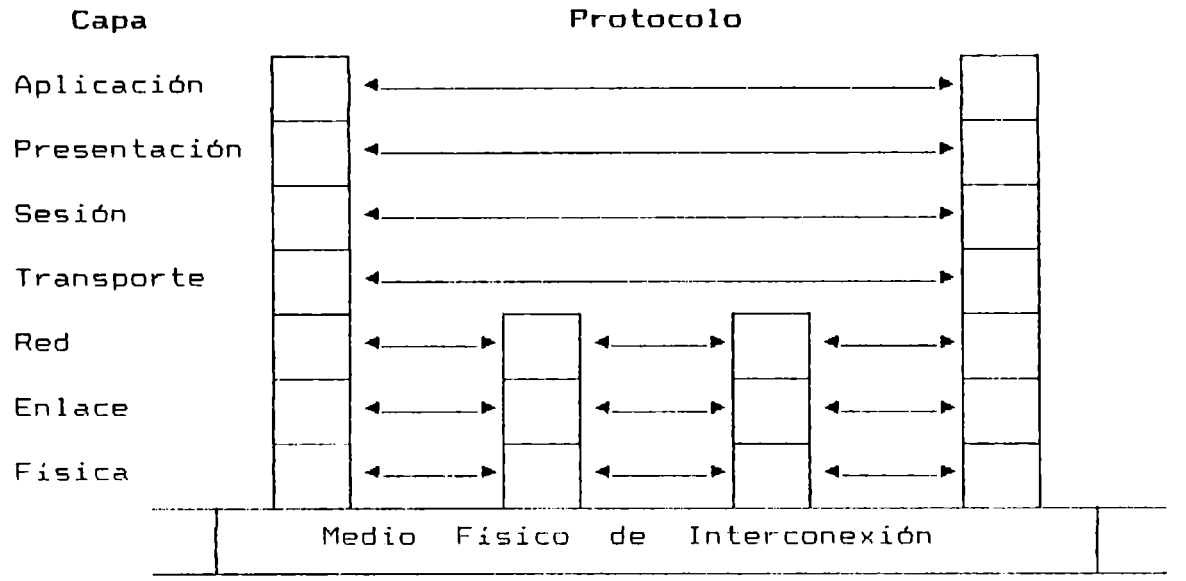


Fig.1: Esquema de una organización en capas.

3- Capa de Sesión:

Tiene como objetivo ofrecer los medios necesarios para organizar y sincronizar la comunicación entre dos aplicaciones, o partes de una aplicación.

4- Capa de Transporte:

Esta capa provee para las superiores un servicio independiente del medio de comunicación (capa inferior), de forma de tornar transparente la existencia del subsistema de transmisión, su estructura y características tecnológicas. Debe garantizar las transferencia correcta, ordenada y de manera eficiente de la información generada en un elemento de computación hasta su destino final. La función principal consiste en aceptar los datos de la capa de sesión, dividirlos, siempre que sea necesario, en unidades mas pequeñas, pasarlas a la capa de red y asegurar que todos ellos lleguen correctamente al otro extremo.

5- Capa de Red:

Esta capa esta afectada a la transferencia de información entre los diversos componentes del subsistema de transmisión . La unidad de información tratada es conocida como *paquete*. La comunicación entre las entidades de esta capa es llevada a cabo

de manera punto-a-punto.

Entre las funciones esta el control de flujo para evitar congestionamientos, la detección y corrección de los errores de los paquetes y la multiplexación de conexiones de red para optimizar los recursos de comunicación. El servicio típico de esta capa es el "circuito virtual".

El patrón escogido por la ISO es X.25 de la CCITT.

6- Capa de Enlace:

El objetivo de esta capa es proveer, para las entidades comunicante de la capa de red, una línea de comunicación que se encuentre libre de errores, a partir de la línea física existente. Los datos son segmentados y enviados en unidades conocidas como *tramas*. Es su función delimitar las tramas y transmitirlos en secuencia. Cuando una trama es recibida, el receptor envía otra de reconocimiento.

7- Capa Física:

Es responsable de la transferencia de bits a través del medio físico de transmisión. La unidad tratada en esta capa es el "bit". Se relaciona con las características eléctricas y mecánicas de la transmisión. Es también función de esta capa el establecimiento de conexiones físicas, el secuenciamiento de los bits, y notificación de condiciones de error para la capa de enlace.

II SEGURIDAD EN REDES COMPUTACION

Contenido Páginas

CRIPTOGRAFIA COMPUTACIONAL

SEGURIDAD EN REDES DE COMPUTACION

CRIPTOGRAFIA COMPUTACIONAL

En la actualidad las redes de comunicación constituyen un elemento vital de las organizaciones para la realización de los negocios u actividades. Esto a llevado a una gran expansión de las redes de computadores donde muchas personas pueden tener acceso a aéreas de datos, abriendo una gran puerta a posibles actividades ilícitas como extraer o modificar información vital de una empresas, personas o instituciones. La criptografía tiene por objetivo garantizar:

- » Confidencialidad de aquella información.
- » Integridad, o sea garantizar a los usuarios que la información es correcta, original, que no haya sido alterada accidental o intencionalmente.
- » Autenticidad de usuarios.

1 ASPECTOS BASICOS DE SISTEMAS DE CRIPTOGRAFIADO

La criptografía se basa en el encriptado que convierte el mensaje original (texto claro) en otro (texto oculto) el cual tiene sentido para el receptor autorizado por medio del desencriptado.

Un criptosistema es una transformación invertible con un parámetro simple k:

$E_k : k \in K$ Donde K es el espacio de claves de longitud finita.

Dado: M; espacio de los mensajes.
C; espacio de los textos encriptados.

El sistema tendrá las siguientes propiedades:

» Algoritmo de encriptado

$$E_k : M \rightarrow C$$

Para cada clave $k \in K$, es una transformación invertible del espacio de los mensajes al de los criptogramas:

$$E_k(m) = c \quad \text{donde:} \quad m \in M \quad y \quad c \in C$$

» Algoritmo de desencriptado; es el algoritmo inverso:

$$E^{-1}_k = D_k \quad \text{tal que} \quad D_k : C \rightarrow M$$
$$\text{donde} \quad D_k(c) = D_k[E_k(m)] = m$$

» Las claves deben definir univocamente al mensaje encriptado.

$$E_{k1}(m) \neq E_{k2}(m) \quad si \quad K_1 \neq K_2$$

Como indica la Fig.1 un Sistema de Criptográfico es un proceso de codificación y decodificación de modo tal que el 1ro no permita que un intruso pueda interpretar la información y además pueda ser decodificada, de forma tal de obtener el mensaje original; el 2do proceso se encarga de la decodificación del mensaje y en algunos casos verificar que este le pertenezca.

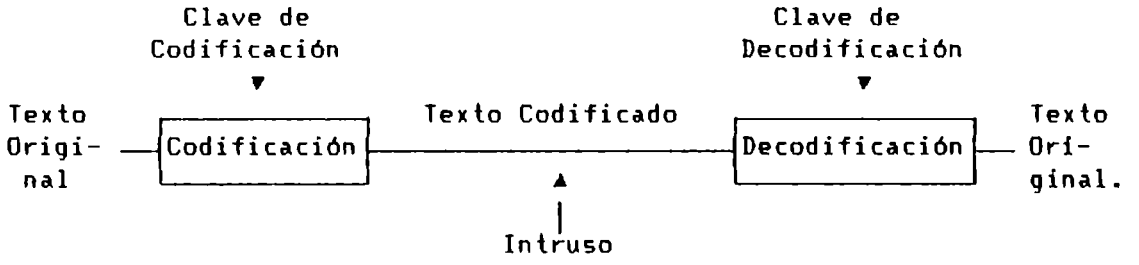


Fig.1: Sistema de encriptado

Como podemos ver en la Fig.1 intervienen claves al codificar y al decodificar, a su vez, estas pueden ser iguales o no.

El criptoanálisis es la ciencia que trata de recuperar el mensaje original a partir del texto encriptado, sin conocer la claves usada en el algoritmo. Aplica las teorías matemáticas, probabilísticas, estadísticas y algebraicas. Utiliza también información relacionada con el lenguaje natural del mensaje y su contexto.

Por lo tanto un sistema ideal sería aquel que tuviera una distribución plana de todos los símbolos posibles en el texto encriptado. Es decir, las características redundantes del

lenguaje deben estar ocultas

2 ENCRIPTADO CLASICO

Criptografiado por Sustitución Monoalfabética por Tabla:

Consiste en cambiar una letra por otra de acuerdo a una tabla.

Criptografiado por Sustitución Monoalfabética por Desplazamiento Fijo:

El emperador romano Julio Cesar tilizo sustitución monoalfabética para comunicarse con sus generales. Esta consiste en sustituir cada letra del texto por otra que está 3 posiciones más adelante y formando un círculo.

```
A -> D
B -> E
C -> F
```

```
Encriptado    : E: i -> i + j
Desencriptado: D: i -> i - j
```

Criptografiado por Sustitución Monoalfabética por Desplazamiento Aleatorio:

Consiste en una secuencia aleatoria d1 d2....dn de enteros que va de 0 a 25 y cada uno representa un desplazamientos, donde el 1ra. carácter se corresponde con el desplazamiento d1; el 2d. con d2; etc.

Criptografiado por Funciones Unidireccionales:

Las funciones unidireccionales tienen las siguientes propiedades:

- Bajo costo en tiempo, espacio y dinero del cálculo de f(x) dado un valor de x. Por lo tanto es fácil de implementar en el emisor.
- Es caro en tiempo, espacio y dinero el cálculo de x dado un valor de f(x). Por lo tanto, difícil de implementar en el receptor.

Es útil en ciertas aplicaciones que realizan funciones inseguras, como por ejemplo el almacenamiento de las "PASSWORD", donde sólo interesa comparar la clave de acceso puesta por el usuario con la que está almacenada y encriptada.

3 ALGORITMO DE ENCRIPTADO SIMETRICO

Cuando la clave para desencriptar es igual a la encriptar, ella debe ser mantenida en secreto y conocida por el receptor y el emisor, lo mismo ocurre cuando la clave de desencriptar es una función computacionalmente viable de la clave de encriptar. Decimos entonces que el sistema es simétrico o de clave secreta.

El texto encriptado es producto de una composición de t funciones $F_1,....,F_t$, donde cada F_i puede ser una transposición o una sustitución.

Shannon propuso utilizar distintos tipos de funciones para crear transformaciones mixtas, por ejemplo aplicando una transposición seguida de una secuencia alternativa de sustituciones y operaciones lineales simples. Para descifrar el texto encriptado se utiliza el mismo algoritmo, invirtiendo el orden de las operaciones y aplicando el inverso de cada sustitución y permutación.

La seguridad de estos algoritmos radica en que la clave solo debe ser reconocida en el emisor y el receptor.

3.1 Sustitución de HILL (Para alfabetos de 26 letras)

Dividiremos el texto en bloque de 3 letras. Para luego reemplazar cada letra por un correspondiente valor numérico de 0 a 25 (quedando una base de representación = 26):

a=0; b=1; c=2; ..... z=25

Ejemplo:

Texto: Sao Paulo es ....

Texto agrupado: Sao Pau loe s.. ...

Texto sustituido: $t_1(18,0,14)$ $t_2(15,0,20)$ $t_3(11,14,4)$ $t_4(18,..,..)$

Para el encriptado:

Seleccionamos una clave que será una matriz T, un nuestro caso una de 3x3 de modo tal que:

$$mdc(\det T, 26) = 1$$

Por ejem.:
$$T = \begin{vmatrix} 2 & 17 & 19 \\ 3 & 4 & 7 \\ 0 & 1 & 2 \end{vmatrix} \Rightarrow \det T = 43 \Rightarrow mdc(43,26) = 1$$

Ahora sustituiremos cada terna del texto original t_i por una

encriptada e_i donde:

$$e_i = (T.t_i) \bmod 26$$

Para el ejem.:

$$\begin{vmatrix}2 & 17 & 19\\ 3 & 4 & 7\\ 0 & 1 & 2\end{vmatrix} . \begin{vmatrix}18\\ 0\\ 14\end{vmatrix} \bmod 26 = \begin{vmatrix}16\\ 22\\ 2\end{vmatrix}$$ ◄ Terna encriptada

Para el desencriptado:

Tendremos que hallar una matriz T^{-1} tal que:

$$(T^{-1}.T) \bmod 26 = \begin{vmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{vmatrix}$$

En nuestro ejem.:

$$T^{-1} = \begin{vmatrix}3 & 7 & 25\\ 8 & 12 & 25\\ 9 & 20 & 1\end{vmatrix} \Rightarrow \begin{vmatrix}3 & 7 & 25\\ 8 & 12 & 25\\ 9 & 20 & 1\end{vmatrix} . \begin{vmatrix}16\\ 22\\ 2\end{vmatrix} \bmod 26 = \begin{vmatrix}18\\ 0\\ 14\end{vmatrix}$$ ◄ Terna Original

El algoritmo de HILL es fácil de quebrar para un criptoanálista con un texto conocido. O sea, determinar T y T^{-1} a partir de n-bloques $(t_1,t_2,..,t_n)$de texto descifrado, bastará resolver:

$$T(t_1,t_2,..,t_n) = (e_1,e_2,..,e_n)$$

3.2 Encriptado de VIGENERE (Para alfabetos de 26 letras)

El encriptado de Vigenere toma como clave a un vector $k=(k_0,k_1,....,k_{n-1})$ en base 26. Dado el texto representado por la letras t0, t1,..,tj,.. será sustituido por el correspondiente encriptado e1,e2,.., de modo que:

$$e_i = (t_i+k_i \bmod n) \bmod 26$$

Entonces para la 1ra. letra (k0) es la sustitución de "Cesar", para la 2da. letra (k1) es la sustitución de "Desplazamiento Aleatorio".

Por lo tanto el encriptado de Vigenere es extremadamente frágil para tamaños de clave pequeñas, así también cuando se hace un criptoanálisis de un texto conocido.

3.3 DES -Data Encryption Standard-

Tuvo su origen en el sistema LUCIFER de IBM que fue mejorado en forma conjunta con la NASA para ser aprobado en 1977 por el gobierno estadounidense para sus datos no clasificados.

DES codifica bloques de 64 bit, por lo tanto va tomando el texto de a bloque de 8 Byte; usando una clave de 56 bit (7 Byte). Esta es una manera segura de codificación ya que se pueden construir $256 = 7 \times 10^{16}$ claves distintas. A 1 µs por prueba de una clave y teniendo en cuenta que un año tiene 3×10^{13} µs entonces un proceso sin paralelismos tardaría 2.000 años en determinar la clave.

DES está basado en transformaciones compuestas:

» Sustituciones No Lineales para bloques pequeños.
» Permutaciones de bits, Transformaciones Lineales para cualquier tamaño de bloque.

El DES encripta bloques de 64 bit, separando el texto de bloques de 8 Byte cada uno, usando una clave de 56 bits (7 Byte). En realidad de 64 bits, donde 8 del los cuales son de paridad.

Al bloque de entrada T (Ver Fig.2) se le aplica una permutación inicial IP obteniendo $T_0 = IP(T)$. (Ver Fig.:2) Después de pasar a través de 16 iteraciones de la función f que combina Sustituciones y Transposiciones, se aplica la permutación inversa IP^{-1} para obtener el resultado final.

Sea T_i el resultado de la i-ésima iteración, L_i y R_i la mitad izquierda y derecha en que se divide T_i ($T_i = L_i R_i$), donde:

$$L_i = t_1 \ldots t_{32}$$
$$R_i = t_{33} \ldots t_{64}$$

Entonces:

$$L_i = R_{i-1}$$
$$R_i = L_{i-1} + f(R_{i-1}, K_i)$$

Donde: "+" es un Or-Exclusivo.
K_i es una clave de 48 bits.

Después de la última iteración, la mitad izquierda y derecha no son intercambiadas, el bloque R_{16} L_{16} concatenados es la entrada a la permutación final IP^{-1}. Esto es necesario para que el algoritmo pueda ser usado para encriptar y desencriptar

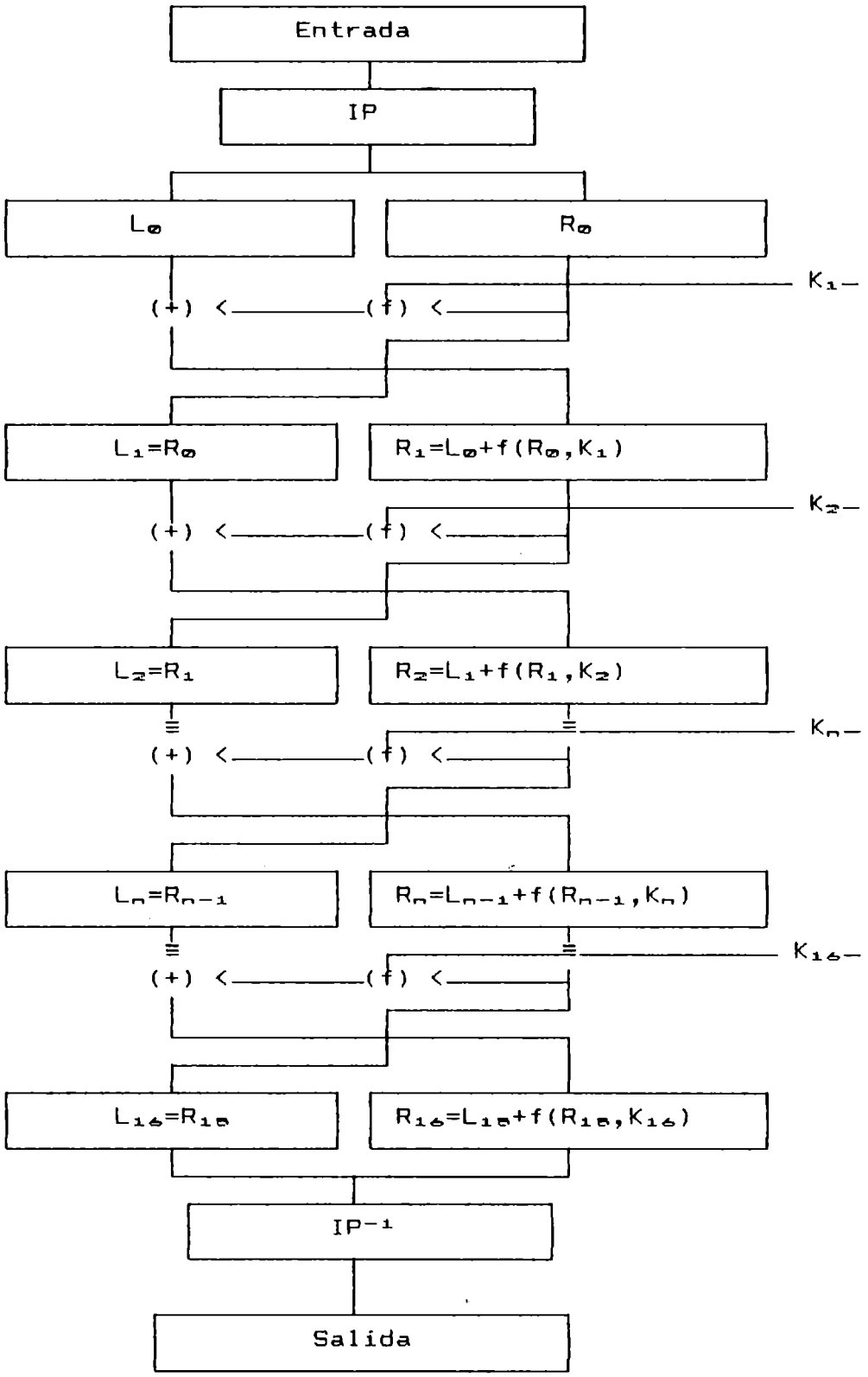


Fig.2: Diagrama en bloques del algoritmo DES.

La Función f y las Cajas S:

R_{i-1} de 32 bits es expandido a un bloque $E(R_{i-1})$ de 48 bits. Para tal fin algunos bits de R_{i-1} son seleccionados más de una vez. Luego se aplica un OR exclusivo entre $E(R_{i-1})$ y K_i. El resultado es dividido en 8 bloques e 6 bits: $B_1\ B_2...B_8$, donde

$$E(R_{i-1}) + K_i = B_1\ B_2...B_8$$

Cada bloque B_j de 6 bits es usado como entrada a la función de Selección S_j ("S-Box), a cual retorna un bloque de 4 bits $S_j(B_j)$, ver Fig.:3. Los bloques son concatenados, obteniéndose un bloque de 32 bits sobre el cual se aplica una transposición (Permutación P). Luego el bloque retornado por la función $f(R_{i-1},K_i)$ es:

$$P(S_1(B_1)....S_8(B_8))$$

Cada caja S_j transforma un bloque de 6 bits $B_j=b_1b_2b_3b_4b_5b_6$ en un bloque de 4 bits de la siguiente forma: el entero correspondiente a b_1b_6 selecciona la fila en la tabla, mientras que el entero correspondiente a $b_2b_3b_4b_5$ seleccionan las columnas. El valor de $S_j(B_j)$ es la representación de 4 bits del entero en esa fila y columna.

Las cajas S implementan las sustituciones no lineales. Su función es hacer que ningún bit de salida sea una función lineal de los bits de entrada y de la clave.

Cálculo de la Clave:

Cada iteración i usa una clave diferente de 48 bits k_i, derivada de la clave inicial K. K es un bloque de 64 bits, con 8 bits de paridad en las posiciones 8,16,...,64 (Ver Fig.:4). Los 8 bits de paridad son eliminados por medio de la permutación PC-1, obteniéndose una clave de 56 bits. El resultado PC-1(K) es dividido en dos mitades C y D.

$$C_i = LS_i(C_{i-1})$$
$$D_i = LS_i(D_{i-1})$$

Donde LS_i es un desplazamiento circular a izquierda. Luego la clave está dada por:

$$K_i = PC\text{-}2(C_iD_i)$$

Para desencriptar se usa el mismo algoritmo, excepto que k_{16} es usada en la primera iteración y K_1 en la 16. Esto es

debido a que la permutación final IP^{-1} es la inversa de la inicial.

$$R_{i-1} = L_i$$
$$L_{i-1} = R_i + f(L_i, K_i)$$

De lo anterior se puede inferir que las claves no son independientes ya que surgen por transposiciones de la clave principal.

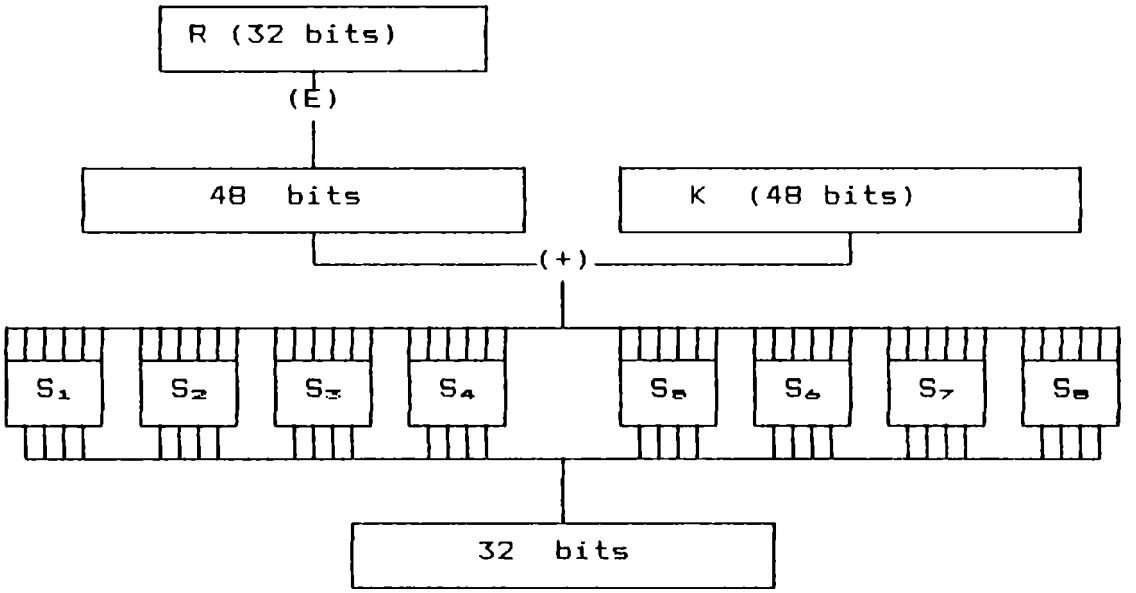


Fig.3: Cálculo de f(R,K)

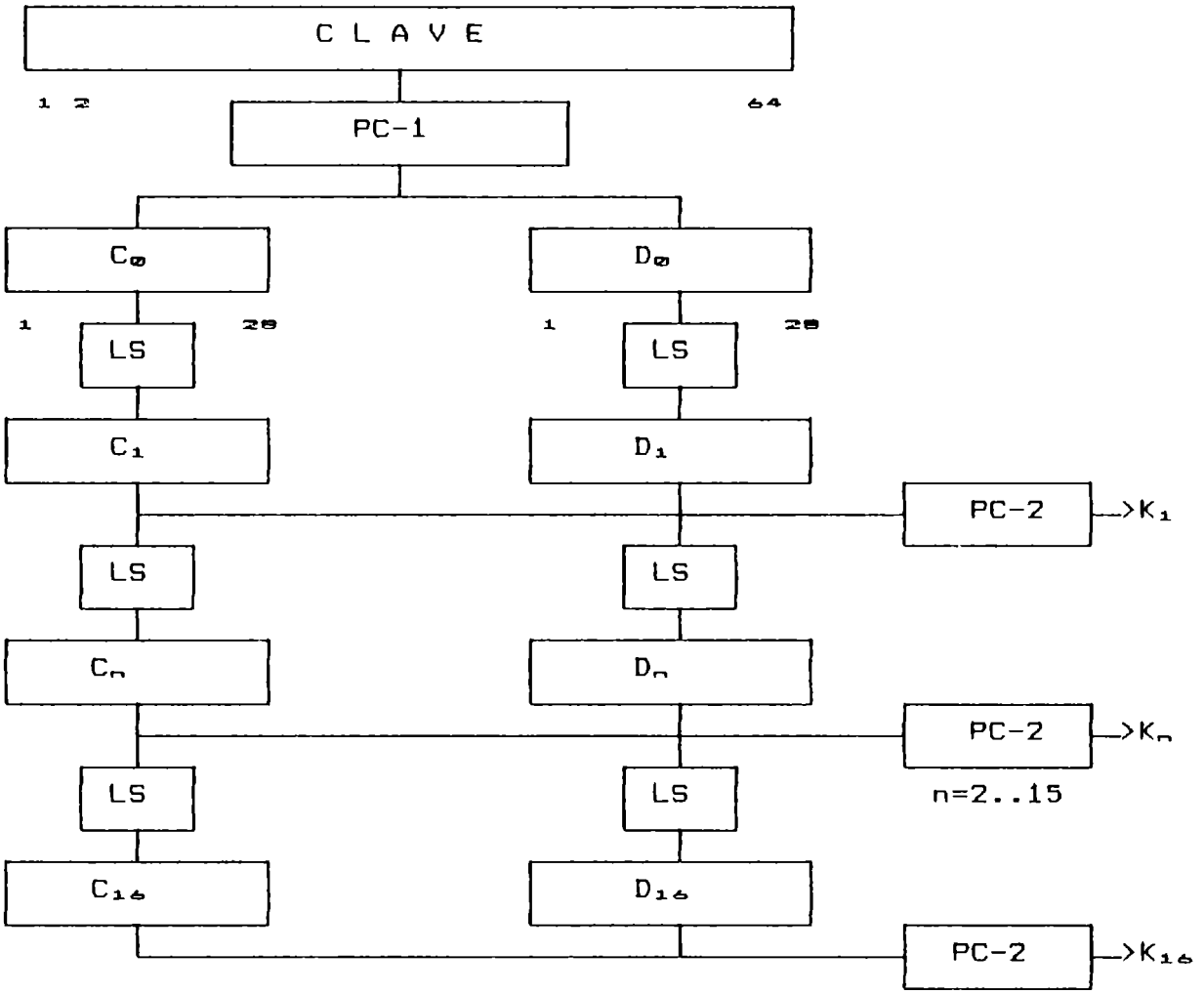


Fig.4: Cálculo de la Clave K

VISIÓN MACROSCÓPICA

La Fig.:5 representa una visión macroscópica la cual consiste en:

1) IP -Sustitución Inicial- : es una sustitución fija que va de 64 a 64 bit.
2) T -Transformación- : es una transformación que depende de una clave de 48 bit.
3) N : permuta dos mitades de 32 bit cada una.
4) K1,K2,..,K16 : claves de 48 bit que dependen de la clave original.

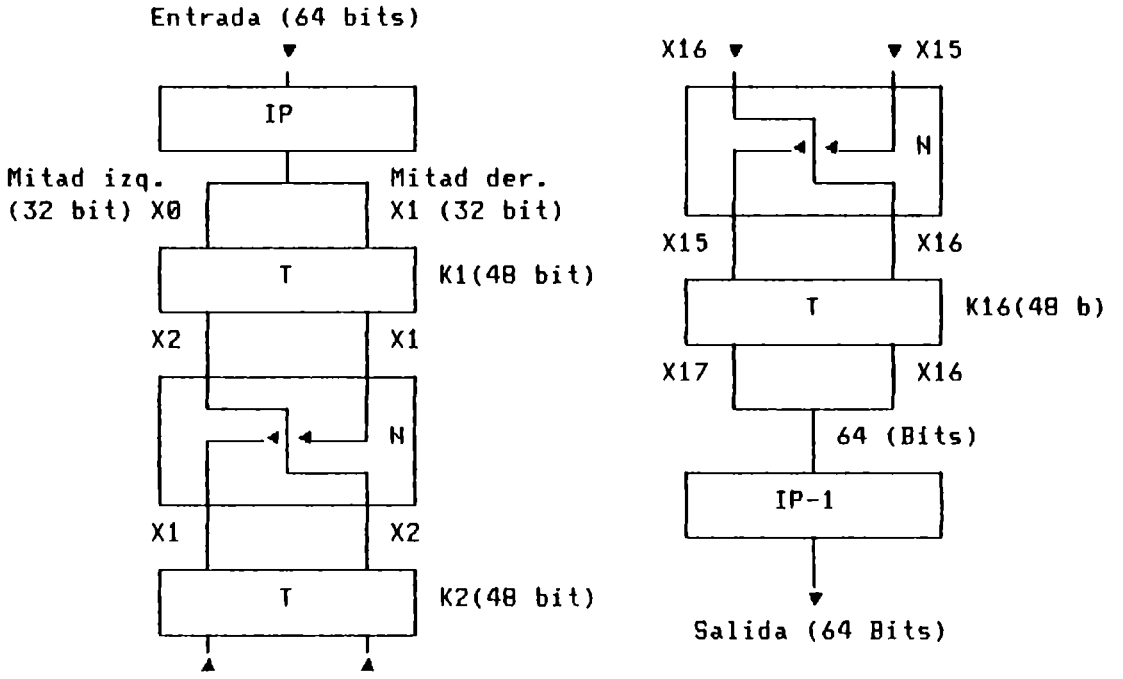


Fig.5: Visión macroscópica de DES

1) La sustitución consiste simplemente en un redireccionamiento a cada bit del bloque que va a encriptar.

Por ejem.: Bit de Entrada	Corresponde a ->	Bit de Salida
0		1
1		3
2		2
3		6
4		0
5		4
6		7
7		5

Al final se realiza el paso inverso IP-1.

2) La transformación; como indica la Fig. 6 consta de 4 pasos:

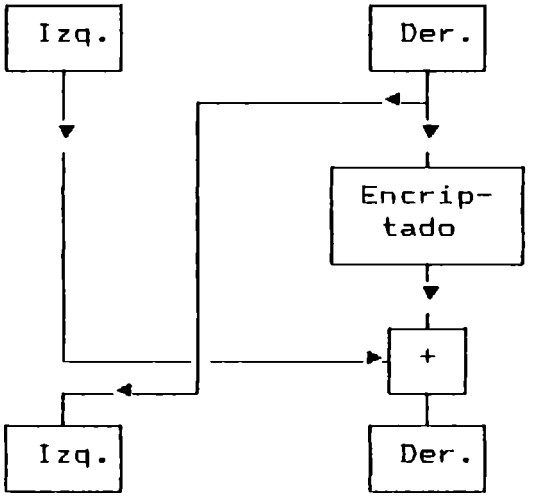


Fig.6: Transformación.

A- Los 32 bits de la Der., pasan a ser los 32 bits de la Izq.
B- Simultáneamente los 32 bits de la Der. mencionados se expande a 48 bits, se encriptan con una subclave de 48 bits (sale de la clave ppal. de 56 bits) por medio de un XOR; para luego contraerse a 32 bits. Finalmente se realiza una sustitución fija de 32 bits.
C- Los 32 bits encriptados en el paso anterior se combinan con los 32 bits de la Izq. por medio de un XOR.
D- El resultado del paso anterior pasa a ser el grupo de 32 bits de la Der. a la salida de la etapa.

3) Luego se permutan las dos mitades de 32 bits del bloque de 64 bits; donde la mitad Izq. pasa a ser la de la Der. y viceversa.
4) Estos dos últimos pasos se repiten en un total de 16 veces.

Generación de Subclaves:
La generación de 16 subclaves K1,..,K16 a partir de la original consta de:

» Como se ha expresado la clave ppal. consta de 64 bits, 56 de los cuales corresponden a la clave propiamente dicha y 8 bits son de paridad, 1 por cada bloque de 8 bits. Por lo tanto se eliminan los bits 7, 15, 23, 31, 39, 47, 55 y 63.

» El resto de los bits son cargados en dos registros C0 y D0 de 28 bits cada uno, de acuerdo a una sustitución fija.

Ejem.:
C0:56 48 40 32 24 16 8 0 57 49 41 33 25 17 9 1 58 50 42 34 26 18 10 2 9 51 43 35

D0:62 54 46 38 30 22 14 6 61 53 45 37 29 21 13 5 60 52 44 38 28 20 12 4 27 19 11 3

» Ahora para cada obtención de una subclave se realiza una rotación (ri) circular a izquierda de los registros C0 y D0 bajo un cierto patrón que indica el desplazamiento obteniendo los registros C1 y D1 de los cuales se seleccionan 48 bits de los 56 bits que ambos registros contienen.

Ejem.: de patrón de rotación, o nro. de posiciones de desplazamiento de los registros C y D para el i-ésimo paso.

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
ri	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Finalmente de esta manera obtendremos Ci, D1 y consecuentemente Ki para cada una de las etapas.

CRIPTOANÁLISIS

IBM y NSA (Agencia de Seguridad Americana) han validado el DES. Uno de los métodos usados por IBM fue la aplicación de testeos de varianza ("X^2 Distribución Chi-Cuadrada") para determinar la no existencia de correlación entre conjuntos pequeños de bits de clave, entrada y salida.

La mayoría de los métodos utilizados por IBM y todos los utilizados por NSA fueron clasificados como secretos. Sin embargo algunas declaraciones fueron publicadas, entre las que se encuentran:

» El DES es muy bueno para aplicaciones civiles.
» La NSA aseguró que el DES está libre de flaquezas analíticas y estadísticas.
» Las Cajas S son No Lineales.
» Al cambiar cualquier bit de entrada de una caja S, cambian por lo menos 2 bits de salida.

En conclusión, es poco probable romperlo por la vía estadística si nos basamos en los testeos X^2. Analíticamente el ataque también es inviable debido a la No linealidad. La única forma es tratando de encontrar la clave secreta. Existen 2^{56} claves posibles, lo que llevaría 10^{17} criptogramas, es decir 10^6 chips días, a 1 µs por criptograma.

Diffie y Hellan argumentaron que la clave de 56 bits es demasiado pequeña. La forma más simple de resolver este inconveniente es utilizar encripciones múltiples con claves independientes K_1, K_2.

Sea M el texto original de 64 bits, C el texto encriptado y K la clave de 56 bits, la operación básica de DES se representa como:

$$C = DES_K(M)$$

y la doble encriptación como:

$$C = DES_{K2}[DES_{K1}(M)]$$

El algoritmo encripta M bajo todos los posibles valores de K_1 (2^{56}), desencripta C con todos los posibles valores de K_2 (2^{56}) y examina la existencia de coincidencias. Este ataque es llamado "ENCUENTRO EN LA MITAD" .

Algunas investigaciones han descubierto la existencia de ciertas claves que desencriptan el mensaje el mensaje al utilizar la doble encriptación. Esto también sucede si se utiliza una clave maestra para encriptar una biblioteca de archivos encriptados.

Para brindar mayor seguridad Dellie y Hellman sugirieron la encripción triple.

En 1978 Tuchman propuso un método que usaba solo dos claves K_1 y K_2. El texto es encriptado con K_1, desencriptado con K_2 y encriptado nuevamente con K_1

$$C = DES_{K1}\{DES^{-1}_{K2}[DES_{K1}(M)]\}$$

Este método evita el ataque anterior y es compatible con el encriptado simple si se define $K_1=K_2$.

$$C = DES_{K1}\{DES^{-1}_{K1}[DES_{K1}(M)]\}$$

Aunque esta técnica provee más seguridad que la encriptación doble es posible atacarla con 2^{56} operaciones y 2^{56} palabras de memoria. Es recomendable utilizar la "técnica de

encriptación triple" con tres claves independientes.

4 CRIPTOGRAFIA DE DOS CAVES O ASIMETRICO

En 1976 tres matemáticos Hellman, Diffe y Merkle desarrollaron en la Universidad de Standford un método de criptografía donde la/s clave/s de encriptado (pública) es distinta a la de desencriptado (privada). La idea es como tener un cofre donde la llave para cerrarlo es distinta a la que se necesita para abrirlo.

En 1977 la Masachusetts Institute of Technology (MIT) amplió los trabajos realizados y construyó el sistema de dos claves mas ampliamente usado, conocido como RSA (Rivest, Shamir y Adleman).

Los algoritmos asimétricos trabajan con dos tipos de claves: *KB Clave Pública* y *Kb Clave Secreta*

Una clave pública usada por el emisor para encriptar el mensaje y una clave secreta usada por el receptor para desencriptar. Una vez encriptado el mensaje, ni el propio emisor puede descifrar el texto, puesto que se necesita la clave secreta correspondiente a la clave pública del receptor para restaurar el mensaje a su forma original.

Las condiciones que deben cumplir los algoritmos asimétricos son:

1ra El Cálculo del par (clave pública K_B y clave secreta K_b), debe ser ejecutado por el receptor en un tiempo polinomial.

2da El emisor, con la clave pública K_B y el mensaje M, debe determinar el texto encriptado, en un tiempo polinomial

$$C = E_{KB}(M) = E_B(M)$$

3ra El receptor B, usando el texto encriptado y la clave secreta K_b, debe recrear el formato original de M, en un tiempo polinomial.

$M = D_{Kb}(C) = D_B(C) = D_B[E_B(M)]$

4ta El oponente, conociendo la clave pública k_B, no debe poder descubrir la clave secreta y/o la condición inicial.

5ta El oponente, conociendo el par (K_B, C) no debe poder recobrar el mensaje M.

4.1 RSA -Rivest Shamir Adleman-

Como se deja ver en la Fig.:7 existe una clave para el encriptado Pública y otra para el desencriptado Secreta; ambas

determinadas por el usuario B -denotándose el par (PB,SB)-.

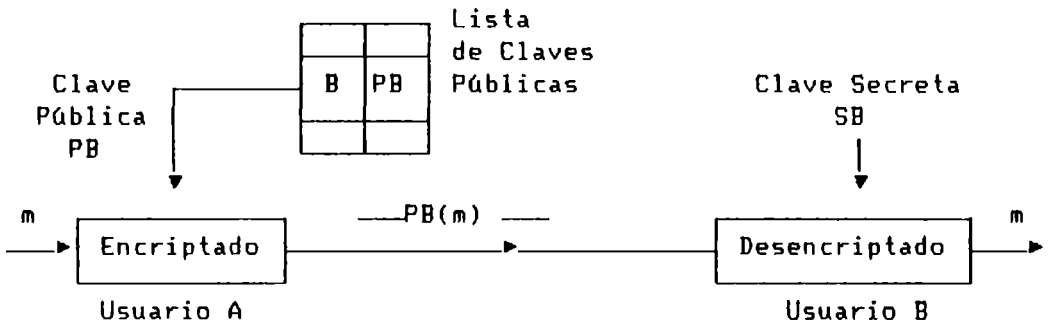


Fig.7: Sistema de encriptado RSA

Las claves están compuestas por pares de números enteros P=(c,n) y S=(d,n). Para su obtención se procede de la siguiente manera:

Paso 1: Se debe generar dos números primos p y q grandes, del orden de 10^{100} cada uno.

Paso 2: Se calcula: $n = p * q$

$$\Phi(n) = (p-1) * (q-1)$$

Paso 3: Se genera un entero c tal que: $1 < c < \Phi(n)$

$$mdc(c,\Phi(n)) = 1$$

(Obtuvimos la clave Pública)

Paso 4: Se genera un entero d tal que: $0 < d < \Phi(n)$

$$(c*d) \bmod \Phi(n) = 1$$

(Obtuvimos la clave Secreta)

Paso 5: Se pública la clave P=(c,n) y se mantienen secretos p, q, $\Phi(n)$, d.

NOTA: mod es el resto de la división entera de dos números enteros.

Para encriptar se procede a dividir el mensaje de a bloques de m bits tal que $0 <= m < n$, una vez obtenido, se le aplica una función matemática en la cual interviene la clave pública "P":

$$m_1 = P(m) = m.c \bmod n$$

Para desencriptar m_1 se utilizar otra función matemática un la cual ahora interviene la clave privada "S":

$$S(m_1) = m_1.d \bmod n$$

Se puede verificar que: S(P(m)) = m tal que 0 <= m < n

Ejemplo:

p = 127 q = 103
n = p*q = 13081 Φ(n) = (p-1)*(q-1) = 12852
c = 59 d = 1307 c*d = 77113
m = 2

Encriptado: P(m) = 259 mod 13081 = 10041

Desencriptado:
S(P(m)) = S(10041) = 100411307 mod 13081 = 2.

CRIPTOANÁLISIS:

Un espía, para poder quebrar el código debe factorial *n* (que es público) para poder obtener *p* y *q*. Ya que *c* también es conocido la clave secreta *d* surge inmediatamente.

La eficiencia del RSA radica en que NO existe ningún algoritmo eficiente para factorizar. Por esta razón es muy importante que el número *n* sea grande del orden de 10^{200}. Ya que el algoritmo de factorización más eficiente conocido, el de R. Schroeppel, demoraría cerca de 109 años a un 1µs por operación aritmética. La Fig.8 muestra una tabla de tiempos estimados para poder factorizar n por R. Schroeppel.

Número de decimales de n = p*q	Tiempo de factorización (1µs/operación aritmética)
50	3,9 Horas
75	104 Días
100	74 Años
200	$3{,}8 \times 10^{7}$ "
300	$3{,}8 \times 10^{15}$ "
500	$3{,}8 \times 10^{20}$ "

FIGURA 8: Tiempo de ejec. para factorear n por el alg. de Schroeppel

Sugerencias a tener en cuenta por los autores del RSA:

- Los números *p* y *q* deben diferir en algunos bits en complemento.Caso contrario estarán muy próximo de n1/2.
- Los números *p-1* y *q-1* deben contener factores primos grandes.
- El *mdc(p-1,q-1)* debe ser pequeño.

SEGURIDAD EN REDES DE COMPUTACIÓN

Una red de computadoras puede ser vista como una "super computadora", en la cual los recursos de hardware y software se encuentran distribuidos sobre una extensa área geográfica. Una importante componente de esta supercomputadora es la red de comunicación que conecta computadoras alejadas. Esta es susceptible a actividades ilegales de extraños. Las dimensiones físicas de la red pueden imposibilitar la protección de los recursos por medios físicos de seguridad. La aplicación de métodos de protección tiene obvias limitaciones, por ejemplo, no se puede usar para proteger información que se comienza a transmitir a través de la red de comunicación. La única clase de método de protección que puede ser aplicado es el método criptográfico. La protección criptográfica no excluye todas actividad ilegal del usuario. Su principal beneficio es la protección de la red contra el efecto de tal actividad.

1 PROTECCIÓN DE LA INFORMACIÓN EN REDES DE COMPUTADORAS

Históricamente, la criptografía era usada para que un texto fuera ilegible. El segundo propósito de encriptar (emcifrar) es la detección de introducciones ilegales, eliminación o cambio de información. De nuestro punto de vista los métodos criptográficos lo usaremos para proteger los canales de transmisión. Estos son de dos tipos:

- *Canales que conectan terminales con su host.*
- *Canales que crea la red de comunicación.*

El nivel y la calidad de seguridad requerida por métodos de protección criptográfica, dependen de la capacidad y del tipo de canal. Especialmente se requiere alta calidad de encriptación cuando la transmisión de información se realiza vía canales satelitales.

1.1 Distribución de Claves Criptográficas

La red de computadora, desde el punto de vista de dos usuarios conectados A y B, es simplemente un canal de transmisión.

Usualmente los usuarios necesitan proteger su información de actividades de extraños. Un usuario puede ser, una simple persona o un grupo de usuarios. Por ejemplo, un grupo de extraños usuarios podría tener control sobre gran parte de la red. Para el diseño de un sistema de protección de información debe tenerse en cuenta la variedad de posibles de accesos ilegales. Entonces, la protección criptográfica de información es aplicada en distinto momentos sobre diferentes niveles de organización del flujo de información. Tratamos con 3 tipos:

- *Encriptación de terminal a terminal.*
- *Encriptación de terminal a host.*
- *Encriptación de host a host.*

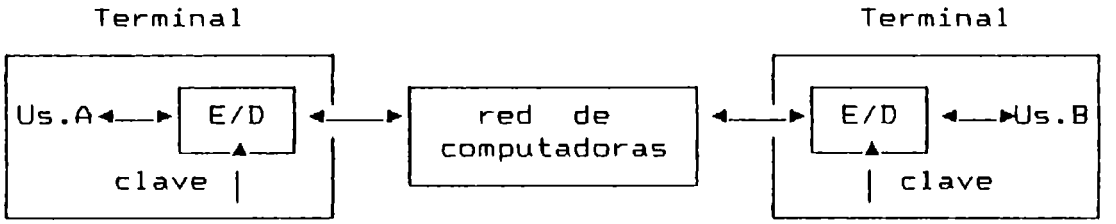


Fig.1: Encriptado de principio a fin.

La encriptación de terminal a terminal asegura que toda información enviada a través de la red es indescifrable dentro de la red de computadoras.

Si el usuario desea usar encriptación terminal a host. Ver Fig.2.

Desde luego, en este caso, la terminal y el host deben compartir una clave. Algunas veces un usuario se encuentra en la terminal de un host extranjero y desea llevar a cabo un proceso de tareas en su propio host.

Puesto que el concepto de red de computadoras surge del deseo del uso uniforme de los recursos, hay flujo de información entre computadoras host. Si una computadora host es sobrecargada con trabajo, sus tareas son asignadas a otra computadora que

pueda realizarlo. Entonces para permitir un flexible orden de trabajo dentro de la red de computadoras con la correspondiente protección de información que se transmite, es necesario compartir claves entre distintos puestos de trabajo.

Fig.2: Encriptado desde terminal extranjera a host.

El primer problema es la distribución de las claves entre dos partes comunicantes es que debe realizarse por medio de canales seguros.

Hay dos métodos para la distribución de claves:

- El primero depende del uso de una red de comunicación separado, exclusiva para la distribución de claves.
- El segundo método consiste en usar la misma red de comunicación para ambas, información y transmisión de claves.

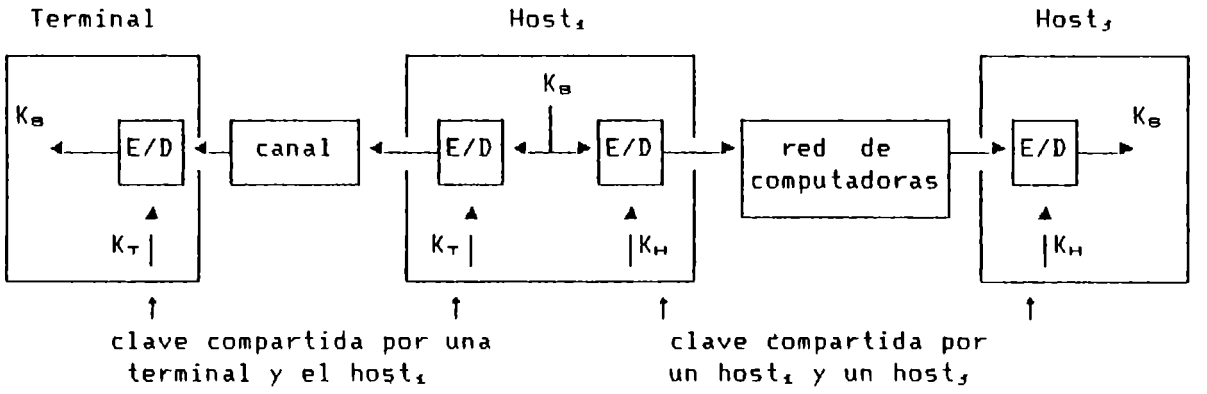


Fig.3: Un ejemplo de distribución de claves entre una terminal y un host.

Para ambos casos, la seguridad del canal es creado por medio de criptografía. En otras palabras, las claves criptográficas son encriptadas y enviadas a las correspondientes partes en forma de criptogramas; por ejemplo la Fig.3.

Ahora consideremos la situación donde un usuario reside en otra terminal y requiere compartir una clave con su propia computadora $host_J$; el requerimiento es aceptado por $host_I$ y este genera una conveniente clave K_S. Después, K_S es enviado al usuario y al $host_J$ por medio de dos canales seguros. Supongamos las claves K_T y K_H usadas para crear dichos canales.

Hay claves diferentes que son usadas simultáneamente para propósitos distintos, pero todas pueden ser clasificadas de las siguientes 3 maneras:

1-Claves de Sesión; son usadas por una única sesión. Estas claves son aplicadas para emcifrar y descifrar mensajes enviados entre usuarios y su computadora host.

2-Claves de Dispositivos; son usados para proteger información que es transmitida entre terminales y host o entre host. Estas claves son también empleadas para crear un canal seguro para la distribución de las claves de sesión.

3-Claves de Host; sirven para encriptar otras claves almacenadas dentro de la computadora host.

No importa que arquitectura de claves es usada en la red de computadora, tenemos dos diferentes métodos para la distribución de claves. La primera se basa en algoritmos simétricos y la segunda en algoritmos asimétricos.

1.2 Distribución de Claves para Algoritmos Criptográficos Simétricos

Supongamos que la distribución de claves es ejecutada usando la misma red de comunicación y que las claves son enviadas en la forma de apropiados criptogramas. La distribución de las claves en la red puede ser controlada de dos maneras distintas:

1-Sistema de distribución de claves centralizado; donde cualquier simple computadora host con un centro de distribución de claves (CKD), ha sido seleccionado por todas las otras computadoras de la red para la distribución de claves.

2-Sistema distribución de claves descentralizada; donde un grupo de computadoras host se encarga de la distribución de claves en forma descentralizada; donde cada uno de estos host

tiene su CKD, haciéndose responsable por la correcta y segura entrega de claves entre entidades que están sujetas a ellos.

Primero consideraremos el caso cuando existe solamente un centro de distribución de claves en una red. Asumimos que cada usuario comparte con el CKD un par de únicas claves. Supongamos que el usuario A requiere protección mientras enviada información al usuario B.

Primero el usuario A transmite un apropiado requerimiento (R_A) junto con un nombre de identificación (ID_A) a el CKD. Respondiendo, el CKD envía a A un criptograma del mensaje que consiste de:

- Una clave K_{AB} que puede ser usada para proteger la transmisión entre los usuario A y B;
- El nombre de identificación de A (ID_A);
- Una copia del pedido de A (R_A);
- Un criptograma del par (K_{AB},ID_A) obtenido por una clave K_B compartida entre B y el CKD es decir, $E_{KB}(K_{AB},ID_A)$.

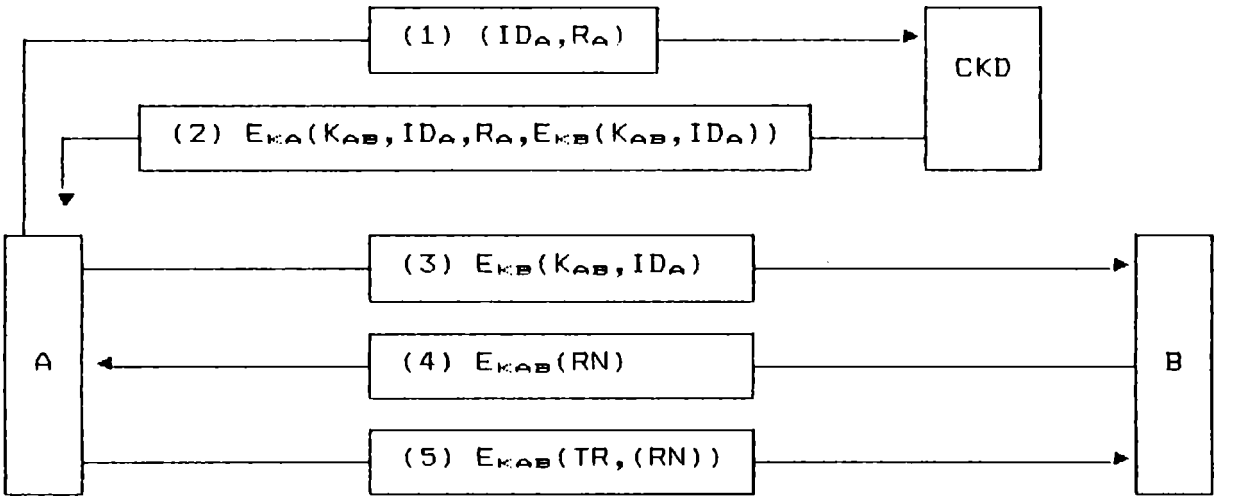


Fig.4: Esquema de la distribución de claves en una red con CKD.

Habiendo obtenido el criptograma, el usuario A descifra y chequéa si el nombre y la copia del pedido se corresponden con el original. Así, A dirige el criptograma $E_{KB}(K_{AB},ID_A)$ al usuario B. El usuario B descifra el criptograma y recrea la forma limpia de ambos K_{AB} y el nombre ID_A (esto posible por que B comparte con el

CKD la clave K_B). Como resultado de estos pasos, entre A y B se tiene la misma clave K_{AB}, así ellos pueden crear un canal común de comunicación seguro. Usualmente teniendo K_{AB}, B inicia una proceso de autenticasión para eliminar la posibilidad de la sustitución de falsos criptogramas.

La otra solución consiste en dar el derecho de la distribución de claves a todas las computadoras host de la red. En este caso cada host tiene tantas claves como otros host haya. Si la red contiene N hosts (N nodos) cada host guarda N-1 claves que son exclusivamente usadas para la comunicación apropiada con la host. De este modo N(N-1)/2 claves diferente pueden ser almacenadas en la red y deben ser intercambiadas de tanto en tanto, usando, por ejemplo, regístros de correo. Si no necesitamos usar canales separados para la distribución de claves, un sistema de distribución de claves jerárquico puede ser utilizado.

Tal organización esta entre la distribución centralizada y descentralizada de claves. Usualmente los host son divididos en tres clases:

1-CKD Global.
2-CKD Regional.
3-CKD Local.

Un CKD local es responsable por la distribución de claves entre todas las entidades de una computadora host. Los hosts con CKD local ubicados dentro de una región están sujetos ó sometidos a un correspondiente host con responsabilidad regional. Luego, todos los host con responsabilidad CKD regional son sujetos a un host con responsabilidad global. Así, si alguna entidad (la entidad puede ser una computadora, un host con responsabilidad CKD local o global) de la red requiere un canal seguro, esta realiza un pedido para el correspondiente host. Respondiendo, esta CKD despacha una clave en el misma manera que antes.

1.3 Distribución de Claves para Algoritmos Criptográficos Asimétricos

Consideraremos una red en la cual la protección se basa sobre un algoritmo criptográfico asimétrico, donde pareciera que los problemas de distribución de claves desaparecerían; sin embargo esto no sucede. En criptografía asimétrica alguna entidad puede generar su propio par de claves, y una clave pública puede ser enviada vía un inseguro canal sin comprometer la seguridad de

una clave secreta. De este modo, el problema de la transmisión de claves vía un canal inseguro puede ser solucionado.

Un transmisor, el cual usa una clave pública, desea estar seguro que los criptogramas serán descifrados por el receptor correcto. Para hacer esto en una red de computadoras con criptogafía de claves asimétrica, la autenticación tiene que ser usada. Una solución es guardar todas las claves públicas en un lugar aparte dentro de la red de computadoras llamado un directorio de claves (KD). El KD es responsable del mantenimiento, puesta al día y distribución de todas las claves públicas usadas en la red de computadoras. Además algún usuario ó entidad recibiendo una clave también debería estar en condiciones de comprobar que la clave sea autentica.

Consideremos una red con protección criptográfica basada en un algoritmo asimétrico; ver Fig.5. Asumimos que dos usuarios A y B han generado sus propios par de claves (K_A,K'_A) y (K_B,K'_B), respectivamente, donde las claves secretas K'_A, K'_B son conocidas exclusivamente por sus propios dueños. Supongamos también, que KD tiene revelada su clave pública K_{KD}. Ahora si un usuario A quiere enviar información a un usuario B, un pedido apropiado es dirigido por RQ_A al KD junto con el tiempo T.

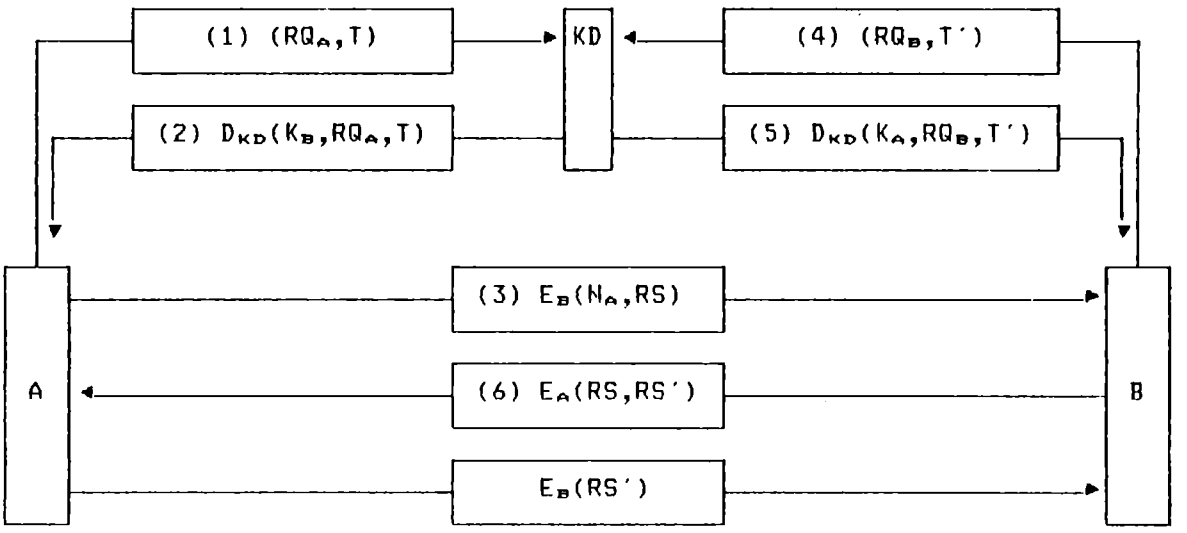


Fig.5: Esquema de distribución de claves en una red con KD

En respuesta, KD envía para A un mensaje emcifrado bajo una clave KD. Observar que KD usa criptografía para autenticación

solamente, o sea, cualquiera puede leer el mensaje pero nadie puede cambiarle el contenido. Conociendo K_{KD}, A descifra el criptograma $D_{KD}(K_B,RQ_A,T)$ y recrea una clave pública auténtica K_B, una copia del pedido RQ_A, y el tiempo T. Ahora, teniendo la clave pública K_B, A envía a B un criptograma $E_B(N_A,RS)$ donde N_A es el nombre de A, y RS es una casual secuencia escogida por A.

Subsiguientemente, para obtener la clave pública auténtica para A, el usuario B dirige un pedido RQ_B junto con el corriente tiempo T' en una forma clara. El directorio de claves KD responde enviando un criptograma $D_{KD}(K_A,RQ_B,T')$, donde K_A es la clave pública del usuario A. Después de descifrar el criptograma, B tiene la clave pública auténtica de A. Luego, B crea un par: La secuencia RS obtenida de el usuario A y una secuencia aleatoria RS' la cual es en si misma generada. El par (RS,RS') es emcifrado bajo la clave pública de A y enviado a este en la forma de criptograma $E_A(RS,RS')$. El criptograma es descifrado por A quien compara la secuencia original RS con la que obtuvo del criptograma. Si esta secuencia son las mismas, A esta segura que el usuario B es auténtico. Finalmente, A retransmite RS' en la forma de criptograma $E_B(RS')$; así, B también compara el original RS' con que recreó el criptograma. Si igual, B confirma la autenticidad de A.

De este modo concluimos que cualquiera de esta dos soluciones (algoritmos basados en criptografía asimétrica y simétrica) requieren la existencia de centros de control que verifiquen la autenticidad de claves e inicialice la comunicación entre entidades de la redes. Para la ejecución de estas funciones, es necesaria la misma cantidad de hardware, software y tiempo. Más aun, en cada caso, la cuestión es la protección de todas las claves que crean canales seguros entre una entidad y un centro. Parecería que un centro de una red, con protección basada en un algoritmo criptográfica simétrico, es más sensible a actividades ilegales, ya que las claves criptográficas son almacenadas aquí y su seguridad juega un importante papel.

En una red con protección basada en algoritmos asimétricos, sabemos que un centro de distribución de claves públicas no es necesitado. Sin embargo, en una red con algoritmos simétricos, todas las claves son almacenadas en la forma de criptogramas bajo la clave principal o maestra del host.

1.4 Generación y Almacenamiento de Claves Criptográficas

Asumimos que las claves son almacenadas y generadas en

algún apropiado lugar de la red. Si consideramos la protección basada en algoritmos criptográficos asimétricos, alguna entidad de la red es responsable por la generación y almacenamiento de sus claves secretas (manejador de distribución de claves). Desde luego la generación de un par (clave secreta, clave pública) dependen del algoritmo asimétrico usado; por ejemplo, en el algoritmo RSA, la clave pública o la secreta es un entero seleccionado aleatoriamente. Si un par de claves (pública y secreta) a ha sido comprometida, entonces la otra par puede ser calculada.

En vista que en una red con protección basada en un algoritmo simétrico, trata con muchas clases diferentes de claves criptográficas. Sin embargo, podemos distinguir dos diferentes clases:

1- Todas las claves que son aplicadas directamente a encriptar mensajes.
2- Todas las claves que son usadas para encriptar otras claves.

Entre todas las claves de la segunda clase, estan las llamadas *Clave Maestra del Host* y *Clave Maestra de la Terminal*. Residen en un host dado o una terminal dada respectivamente, en forma clara. Así, la revelación de sus valores están comprometidos enteramente a la seguridad del host y la terminal. Por otro lado, la destrucción de sus valores medios (recursos) que otras claves emcifradas son mal descifradas, y de este modo, todo mensaje emcifrado puede hacerse ilegible.

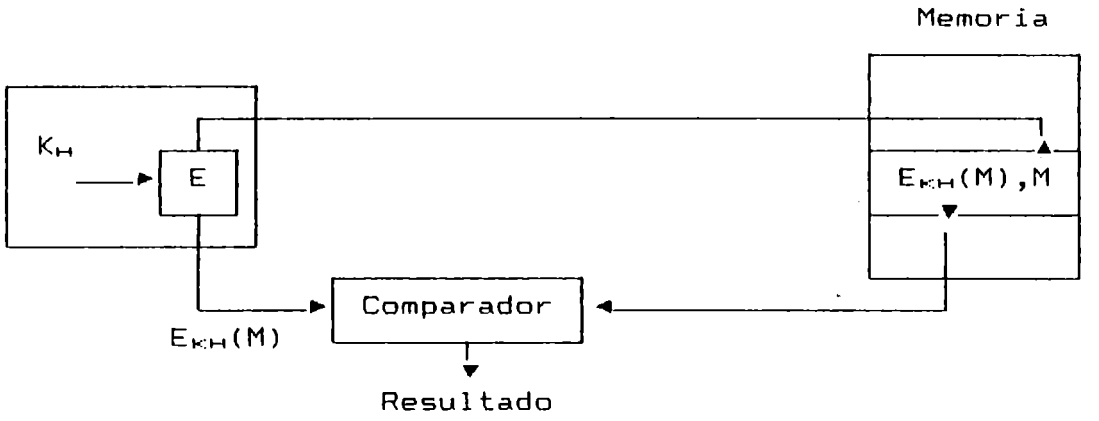


Fig.6: Autenticación de la clave maestra del host.

El valor de una clave maestra es fijo para un largo período de tiempo. Usualmente, su curso de vida cuenta con varias semanas o días. Así la generación y almacenamiento de claves maestras son editadas en protección criptográfica. En la práctica, una clave maestra de host es creada por selección aleatoria entre todas los posibles valores de clave. Luego de elegida de esta manera, una clave es puesta dentro de un protegido sistema criptográfico de solo escritura. Estos sistemas deben ser diseñados de tal manera que, después que una clave a sido puesta dentro de un sistema criptográfico, descubrir la clave es imposible, pero todavía debe haber una manera de chequear si el valor de la clave es correcto.

El problema de autenticación de las claves puede ser solucionado de diferentes maneras.

Un administrador, habiendo cambiado una clave maestra de host, encripta un mensaje dado M bajo K_H. El par (criptograma $E_{K_H}(M)$, mensaje M) es almacenado en memoria. Siempre que se requiera la autenticación de una clave maestra de host, el mensaje M es traído de la memoria y puesto dentro del sistema criptográfico. Un criptograma es obtenido y comparado con el criptograma almacenado en memoria. Si son iguales, la clave es considerada correcta.

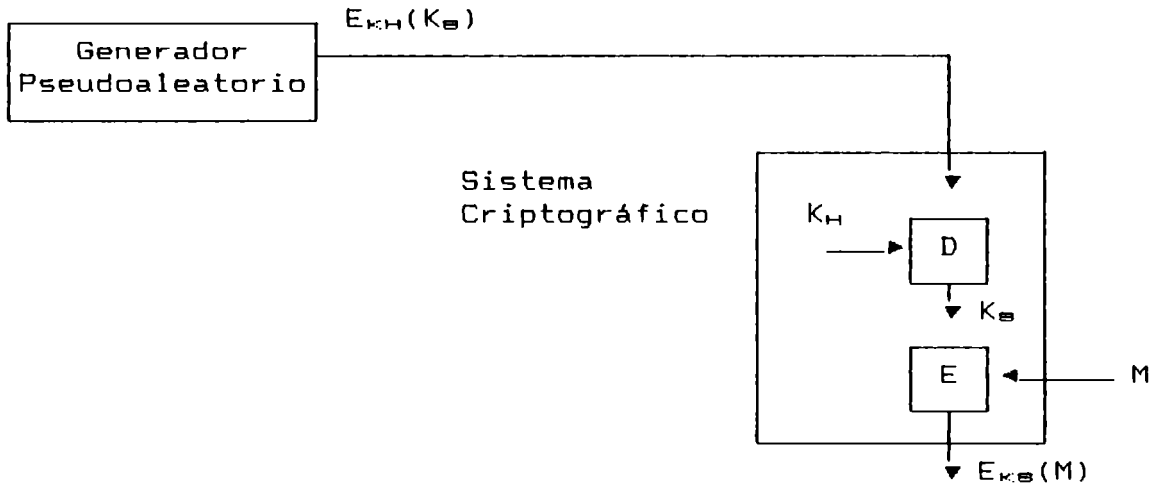


Fig.7: Un método de protección de clave.

Las otras claves subsidiarias son creadas usualmente por medio de un generador pseudoaleatorio y pueden ser almacenadas en un lugar desprotegido. Esto es posible puesto que todas las claves subsidiarias son generadas en forma de apropiados

criptogramas, en lugar de una clave K_s, su criptograma $E_{K_H}(K_s)$ obtenida usando la clave maestra del host K_H. El criptograma es descifrado justo antes que la clave K_s sea usada. Tal solución es dada en la Fig.7.

Si necesitamos emcifrar un mensaje M bajo una clave K_s, ponemos ambos el criptograma $E_{K_H}(K_s)$ y el mensaje M dentro de una entrada del sistema criptográfico. Primero, K_s es recreada y, luego, M es emcifrado usando la forma clara de K_s.

Vemos que la protección de claves subsidiaría depende sobre la seguridad del sistema criptográfico. Un sistema criptográfico debe ser diseñado como un circuito LSI y incluido en un lugar físicamente protegido.

1.5 Protocolos Criptográficos

Hasta ahora consideramos la protección de la información entre diferentes entidades de las redes, como terminales, hosts. La comunicación de subred puede ser tratada como un canal simple cuyo papel es suministrar el medio de transmisión, sin embargo, la protección criptográfica también puede ser introducida en la comunicación de subred entre nodos adecuados, aunque esto presente ciertos problemas.

Consideramos el modelo OSI -Reference Open System Interconnection-, algunas capas pueden ser usadas para crear canales virtuales entre dos entidades y pueden ser aplicadas para proporcionar operaciones criptográficas. Desafortunadamente, en una red de computadoras, alguna información debe ser transmitida en forma limpia, como una dirección adecuada y datos de control que serán usados por la capa más baja; esos datos forman la cabecera del paquetes. Por supuesto, es fija la posibilidad de descifrar las cabezas, pero eso solo puede ser completado antes de la entrada de los paquetes en el canal físico.

Las cabeza de los paquetes no deben ser encriptada por, esta situación asigna un usuario ilegal que actúa desde un nodo la posibilidad de sustituir alguna información en la cabeza.

Vamos a presentar un protocolo aplicado, en una red ARPA, para la creación de un canal virtual, el concepto de canal virtual es usual en una red con intercambio de paquetes. Los mensajes son divididos en una unidad de transmisión llamada paquete, estos pueden enviarse a través de distintas rutas. La secuencia correcta de los paquetes es recreada en el nodo destino, con la información de la cabeza de paquete. Para ambos, el emisor y el receptor, aparece que el mensaje ha sido enviado

vía un canal virtual simple con protección criptográfica. El esquema general de intercambio de información para este protocolo es dada en la siguiente figura, la cual bosqueja sistema manejador de red - NMS: Network Management System -.

La Fig.8 muestra el sistema de manejo de red (NMS) responsable de:

1- Canal de distribución.
2- Control de flujo de información.
3- Retransmisión de información que ha sido distribuida y la cual no puede ser recreada en su forma correcta.

Por su puesto, la calidad de la protección criptográfica depende de las correcciones de los sistemas operativos que supervisan la aplicación de transformaciones criptográficas.

Desde ahora asumimos que los sistemas operativos trabajen correctamente y que están enriquecidos por la adición de 2 subrutinas criptográficas, ENCRYPT y DECRYPT. Supongamos que un cierto programa A del nodo N_1 requiere un archivo de datos para ser transmitido a otro programa B de un nodo remoto N_2. Primero que todo, llama al sistema operativo -OS- del nodo N_1. El SO llama al archivo de datos y lo transforma usando la subrutina ENCRYPT. la información está organizada en paquetes al cual un encabezado es adjuntado y esta colección de paquetes es enviada vía la comunicación de subred usando un algoritmo predeterminado de enrrutamiento. Las cabeceras, las cuales están en forma limpia, hacen lo posible para recrear la secuencia correcta de paquetes en el nodo destino N_2. En ese nodo, luego de chequear la dirección en las cabezas (encabezados), el NMS del nodo N_2 advierte al programa B que ha arribado el archivo de dato y lo almacena en un buffer adecuado de N_2. B llama a su SO local usando la subrutina DECRYPT. El SO, usando una clave criptográfica prestablecida, desencripta el archivo resultante. La forma limpia del archivo es enviada al prog B.

Cuando consideramos un método de clave de distribución entre 2 nodos que se comunican, asumimos que tratamos con una red de computadoras con clave de distribución descentralizada y donde la transformación que encripta es llevada a cabo por un algoritmo criptográfico simétrico. Ahora suponemos que una tabla de claves reside en cada nodo de la red. La tabla, la cual es accedida sólo por un SO local adecuado, contiene descripciones de todos los canales virtuales actualmente existentes, los cuales tienen origen o finalizan en ese nodo. Considerar la tabla en el nodo N_1. La descripción de un simple canal virtual, el cual conecta 2 programas A y B de los nodos N_1 y N_2 respectivamente, consiste

de:

- » El nombre del nodo externo N_2 conjuntamente con el nombre del programa B.
- » El nombre del programa local A.
- » Nombre del canal virtual.
- » El valor de la clave criptográfica.

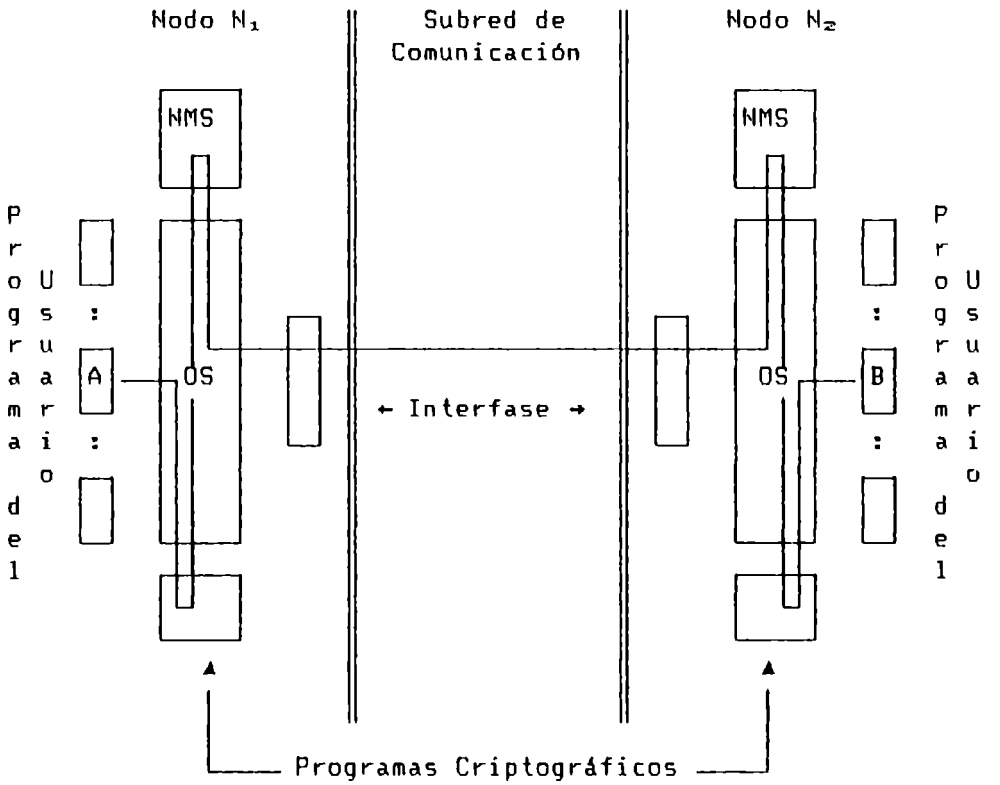


Fig.8: Intercambio de información entre los programas de los usuarios A y B.

Limpiamente el SO de cualquier nodo debería ser enriquecido por la adición de 2 subrutinas las cuales ejecutan 2 operaciones sobre el contenido de la tabla de clave. La primera subrutina CREATE(nombre del programa externo, nombre del programa local, nombre del canal, dato adicional) puede ser llamado de un programa local y escribe el dato adecuado en la tabla de clave. La segunda subrutina CANCEL(nombre del canal) causa el borrado de toda la información concerniente a un canal virtual dado y simultáneamente, cualquier nodo, donde esta operación ha sido llevada a cabo, notifica al nodo externo que corresponda acerca

de la cancelación. Por lo tanto, si 2 programas A y B residen en los nodos N_1 y N_2 respectivamente, la información intercambiada entre 2 nodos tiene la siguiente forma:

1- Programa A de N_1 envía a su NMS local un requerimiento para establecer la comunicación con el programa B de N_2 vía un canal virtual llamado X. Este mensaje es liberado en forma limpia.
2- El NMS de N_1 transmite un mensaje de control al NMS de N_2 usando el mensaje de intercambio de protocolo vía un canal virtual previamente establecido.
3- El NMS de N_2 llama a una rutina input/output vía una interrupción adecuada.
4- El NMS de N_2 lleva a cabo el Paso 2 e inicia el Paso 3 para ser llevado a cabo dentro del nodo N_1. Esto lleva al canal virtual llamado X a establecer si esto es posible. El canal conecta el OS del nodo apropiado y es habilitada la comunicación entre los programas A y B.
5- El NMS de N_1 llama a la subrutina CREATE(B,A,X,dato adicional).
6- El OS de N_1 genera una clave criptográfica y la pone con el dato adicional en la tabla de clave. Luego envía un mensaje adecuado, vía un canal seguro prestablecido, al OS de N_2 el cual transmite la 3-tupla (B,A,X) a su NMS local.
7- El NMS de N_2 llama a la subrutina CREATE(A,B,X,dato adicional), y el OS pone la clave criptográfica en la tabla de clave. Luego, el NMS de N_2 genera una interrupción del programa B y notifica al NMS de N_1, el cual, a su vez, interrumpe el programa A.
8- Los programas A y B se comunican entre si vía el canal virtual X usando protección criptográfica.

El justo perfil del procedimiento para el intercambio de información puede fallar si tanto el canal virtual es ocupado o la tabla de clave no tiene espacio para una nueva entrada. Por lo tanto, el procedimiento para el intercambio de información debería ser modificado sutilmente si la protección criptográfica es basada sobre un algoritmo asimétrico. Sin embargo, cualquier método de protección criptográfica que aplicamos, el diseño de protocolos criptográficos, son pesados y son consumidores de tiempo, por esto es necesario dar consideraciones para la arquitectura de redes y la necesidad de protección.

2 ADMINISTRACIÓN DE LA EMISIÓN DE CLAVES

Se mostró que el uso de criptografía para encriptación de datos, PINs y para autenticar mensajes es virtualmente importantes, manejando un amplio conjunto de claves criptográficas. No solamente hay diferentes claves para cada enlace diferente, también debería haber diferentes claves por cada función o actividad diferente.

La función del administrador de clave es distribuir y poner al día las claves cuando se requiere en la red.

Hay un número de competidores por clave de control en un ambiente simétrico que incluye *claves de transacción* y *claves de sesión*.

Una solución al problema se administración de claves para la protección criptográfica en una red de computadoras depende de:

- El tipo de transformación criptográfica (algoritmo criptográfico) usado.
- Los requerimientos de protección, cuando la protección criptográfica es aplicada a transmisión de datos y/o protección de archivo (base de datos).
- La arquitectura de red.

Ahora describiremos 3 ideas de administración de claves.

2.1 Ejemplo de Clave de Manejo Usando DES

Suponiendo que toda la protección criptográfica es basada en el algoritmo DES. Asumimos que queremos proteger información, la cual es transmitida entre terminales y hosts ó entre hosts. Hacemos la siguiente suposición acerca de la arquitectura de red:

(S1) Hay solo una clave (dentro del host) y es almacenada en forma limpia. Esta es llamada la *clave maestra del host* $-K_H-$.

(S2) Hay solo una clave (dentro de la terminal) y es almacenada en forma limpia. Esta es llamada la *clave maestra de la terminal* $-K_T-$.

(S3) Todas las otras claves están almacenadas como un adecuado criptograma (ellas son encriptadas bajo la correspondiente clave maestra).

(S4) La generación de la clave maestra es hecha por un generador aleatorio en forma limpia.

(S5) La generación de las otras claves es hecha por un generador pseudoaleatorio y la secuencia de producción es considerada como una adecuada clave criptográfica.

Consideraremos los siguientes requerimientos:

(R1) La clave maestra debe ser conservada en lugar seguro.
(R2) Bajo ninguna circunstancia, debe ser posible la revelación de las otras claves.
(R3) Las instrucciones criptográficas, son necesarias para manejar operaciones criptográficas sobre información y las claves, dichas instrucciones no deben ser privilegiadas y su ejecución debe ser llevada a cabo por el OS del host correspondiente.
(R4) La aplicación de algún subconjunto de instrucciones criptográficas no debería permitir a un usuario ilegal obtener información protegida (claves o mensajes).

Nuestro proyecto consiste en 2 etapas. La primera concierne a protección criptográfica de transmisión entre una terminal y un host. La segunda es aplicada a la transmisión entre hosts. El proyecto está basado sobre el esquema de clave de manejo diseñado por W.Ehrasm.

Protección criptográfica de transmisión entre una terminal y un host

Asumimos que hay claves prestablecidas en el host y su terminal. Son K_H y K_T respectivamente. Antes de una sesión una clave de sesión debe ser generada y enviada a la terminal. Suponiendo que una clave ha sido obtenida y, conforme a lo asumido en S3, ésta sería de la forma $E_{KH}(K_S) = E_H(K_S)$. Como K_T existe solamente en la terminal, $E_H(K_S)$ debe ser rencriptada en $E_{KH}(K_S) = E_T(K_S)$. En orden a esto, definimos la primera instrucción criptográfica *rencriptar* desde la clave maestra del host -Reencipher Host Master Key-. Ver Fig.9.

$$RHMK\ [E_H(K_S)\ ,\ E_H(K_T)] = E_T(K_S)$$

Las transformaciones criptográficas que envuelven la instrucción RHMK son presentados en la Fig.9. Así como $E_T(K_S)$ ha sido obtenida y almacenada en la terminal, la transmisión puede tener lugar entre la terminal y su host. Si un host quiere encriptar datos, usa una instrucción criptográfica no

privilegiada de *encriptación de datos* -Encipher Host, la cual tiene la forma:

$$ENCH\ [mensaje\ M,\ E_H(K_s)] = E_{Ks}(M) = E_s(M)$$

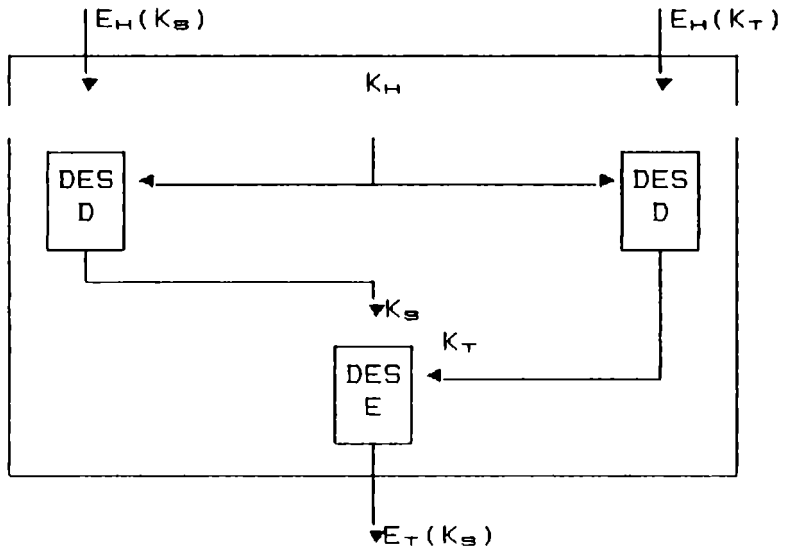


Fig.9: Implementación de la instrucción RHMK

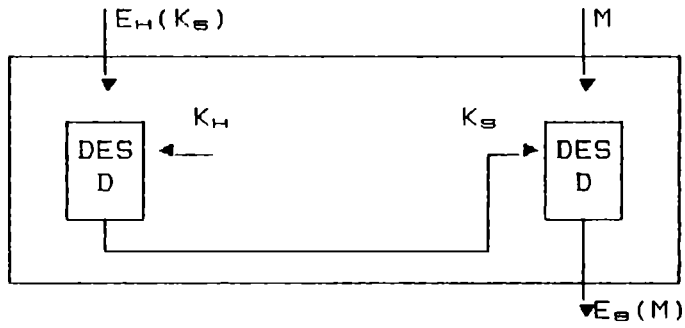


Fig.10: Esquema de una instrucción ENCH.

El criptograma obtenido es transmitido a la terminal. En la terminal, el criptograma es desencriptado por la aplicación de la instrucción *desencriptar el mensaje en al terminal* -Decipher Terminal- como indica la siguiente Fig.11.

$$DECT\ [mensaje\ M,\ E_T(K_s)] = M$$

A su vez, si la terminal desea enviar información, una segunda instrucción *encriptar datos en la terminal* -Encryption Terminal- debe ser definida como:

$$ENCT\ [mensaje\ M,\ E_T(K_B)] = E_B(M)$$

El hardware implementa la instrucción ENCT, es la misma que en la Fig.10. La única diferencia es que ENCT emplea K_T pero ENCH usa K_H.

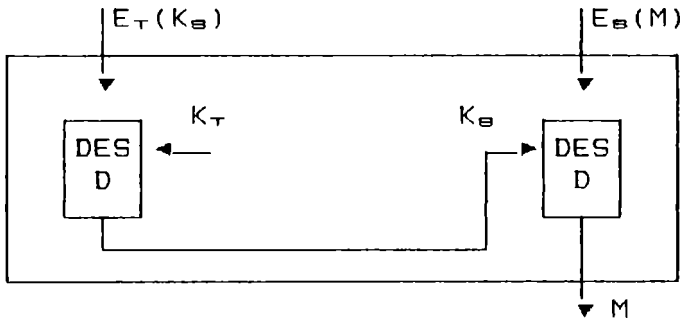


Fig.11: Esquema de una instrucción DECT.

Para desencriptar criptogramas enviados de la terminal, una tercera instrucción criptográfica es definida. Es llamada *desencriptar datos* -Decryption Host- y puede escribirse como:

$$DECH\ [E_B(M)\ ,\ E_H(K_B)] = M$$

Ambos DECT y DECH necesitan el mismo tipo de hard (Fig.11) de antes. La única diferencia es que usan diferentes claves.

Nos permitimos promover la consideración de la instrucción RHMK. Como todas las instrucciones son no-privilegiadas, cualquier usuario, quien resida en el host, puede cambiar la secuencia de entrada de la instrucción RHMK. El usuario, luego obtiene $E_B(K_T)$ en lugar de $E_T(K_B)$. Luego aplicando la instrucción DECH $[E_B(K_T)\ ,\ E_H(K_B)]$, el usuario obtiene en forma limpia el K_T. Esto contradice el requerimiento R2.

Así, esto es deseable a destruir la simetría de la instrucción RHMK. Para esto, podemos introducir 2 variantes de la clave maestra, es decir K_{H0} y K_{H1}. La clave K_{H1} es creada por permutación de los bits de K_{H0}. La primera variante es usada para

proteger las claves de sesión. En otras palabras, las claves de sesión son almacenadas como criptogramas $E_{H\varnothing}(K_s)$. La segunda variante es aplicada a encriptar claves maestras de terminal $E_{H1}(K_T)$. Por lo tanto, la instrucción criptográfica RHMK revisada tiene la forma:

$$RMK\ [E_{H\varnothing}(K_s)\ ,\ E_{H1}(K_T)] = E_T(K_s)$$

Resumiendo, hemos definido 2 operaciones criptográficas en una terminal; la primera ENCT genera criptogramas para mensajes, la segunda, DECT nos permite recrear la forma limpia del mensaje. Por otro lado, en el host, hemos definido una adecuada instrucción de encriptación, ENCH y de desencriptación DECH y también la instrucción RMK que rencripta las claves de sesión.

Protección criptográfica para la transmisión entre hosts

Considérese 2 hosts los cuales tienen terminales propias y la comunicación entre terminales y hosts es protegida usando criptografía. Supongamos que necesitamos enriquecer nuestra protección criptográfica a fin de dar una transmisión segura entre los hosts i y j. Para designar un canal de comunicación seguro entre 2 hosts, una adecuada clave de sesión deberá ser distribuida entre ellos. Supongamos que en los hosts i y j hay una clave prestablecida K^{ij}_{sc}, la cual es llamada clave de comunicación secundaria y es aplicada para crear un canal seguro para claves de sesión entre los hosts i y j.

Usando la suposición S3, observamos que la clave de comunicación secundaria ha de ser almacenada en el host i como un criptograma el $E^{i}_{H\varnothing}(K^{ij}_{sc})$ ó $E^{i}_{H1}(K^{ij}_{sc})$, donde $E^{i}_{H\varnothing}$ propone que, en el host i, dado una clave, sea encriptada bajo la variante de la clave maestra $K_{H\varnothing}$.

Asumimos por el momento que K^{ij}_{sc} es almacenada en el host i como $E^{i}_{H\varnothing}(K^{ij}_{sc})$. Luego, la forma limpia de la clave de sesión K_s puede ser recreada en una terminal del host i. Ahora, esta clave K_s es parte de los hosts i y j y de acuerdo a nuestro requerimiento R2, la forma limpia de las claves nunca debe ser revelada fuera de una área protegida. Revelar una clave es posible si el siguiente procedimiento es aplicado:

1- Aplicar en el host i la instrucción: $RMK[E^{i}_{H\varnothing}(K^{ij}_{sc}),E^{i}_{H1}(K_T)]$. Como un resultado $E^{i}_{T}(K^{ij}_{sc})$ es obtenido. Esto es posible como los criptogramas $E^{i}_{H\varnothing}(K^{ij}_{sc})$ y $E^{i}_{H1}(K_T)$ no están protegidos.
2- Interceptar un criptograma $E^{ij}_{sc}(K_s)$. Hay 2 caminos para

hacerlo. El primero es penetrar en una memoria desprotegida donde los criptogramas de claves están almacenados. El segundo, consiste escuchar secretamente (por medio de dispositivos electrónicos) sobre un canal al cual están conectados los 2 hosts.

3- Habiendo obtenido $E^{i}_{T}(K^{ij}_{sc})$ y $E^{ij}_{sc}(K_s)$, la instrucción DECT $[E^{ij}_{sc}(K_s)\ ,\ E^{i}_{T}(K^{ij}_{sc})]$ es llevada a cabo. Así, la forma limpia de la clave de sesión K_s es recreada.

En otras palabras, las claves de comunicación secundaria no deben ser encriptadas usando la variante $K^{i}_{H\varnothing}$ de la clave maestra. La segunda posibilidad es almacenar las claves de comunicación secundaria bajo el emciframiento de la variante de la clave maestra K^{i}_{H1}. De aquí, suponemos que una clave de comunicación secundaria es almacenada como un criptograma $E^{i}_{H1}(K^{ij}_{sc})$ y $E^{j}_{H1}(K^{ij}_{sc})$ en el host i y el host j respectivamente. Supongamos que el host i quiere transmitir mensajes al host j. El paso inicial release sobre la generación de la clave. En el host i, la clave de sesión es producida como un criptograma $E^{i}_{H\varnothing}(K_s)$. Antes de enviar esta clave al host j. tenemos que remcifrarla. Afortunadamente, en el host i, podemos usar la instrucción RMK como sigue:

$$RMK\ [E^{i}_{H\varnothing}(K_s)\ ,\ E^{i}_{H1}(K^{ij}_{sc})]\ \longrightarrow\ E^{ij}_{sc}(K_s)$$

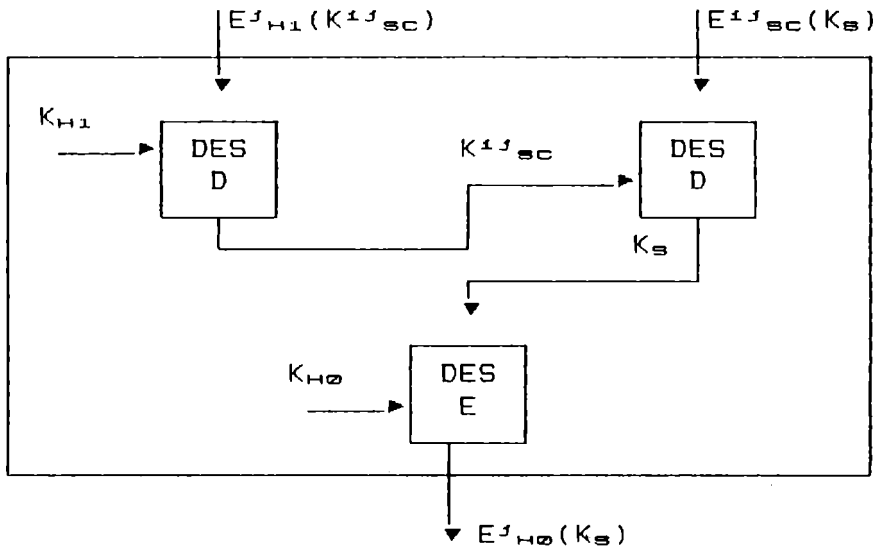
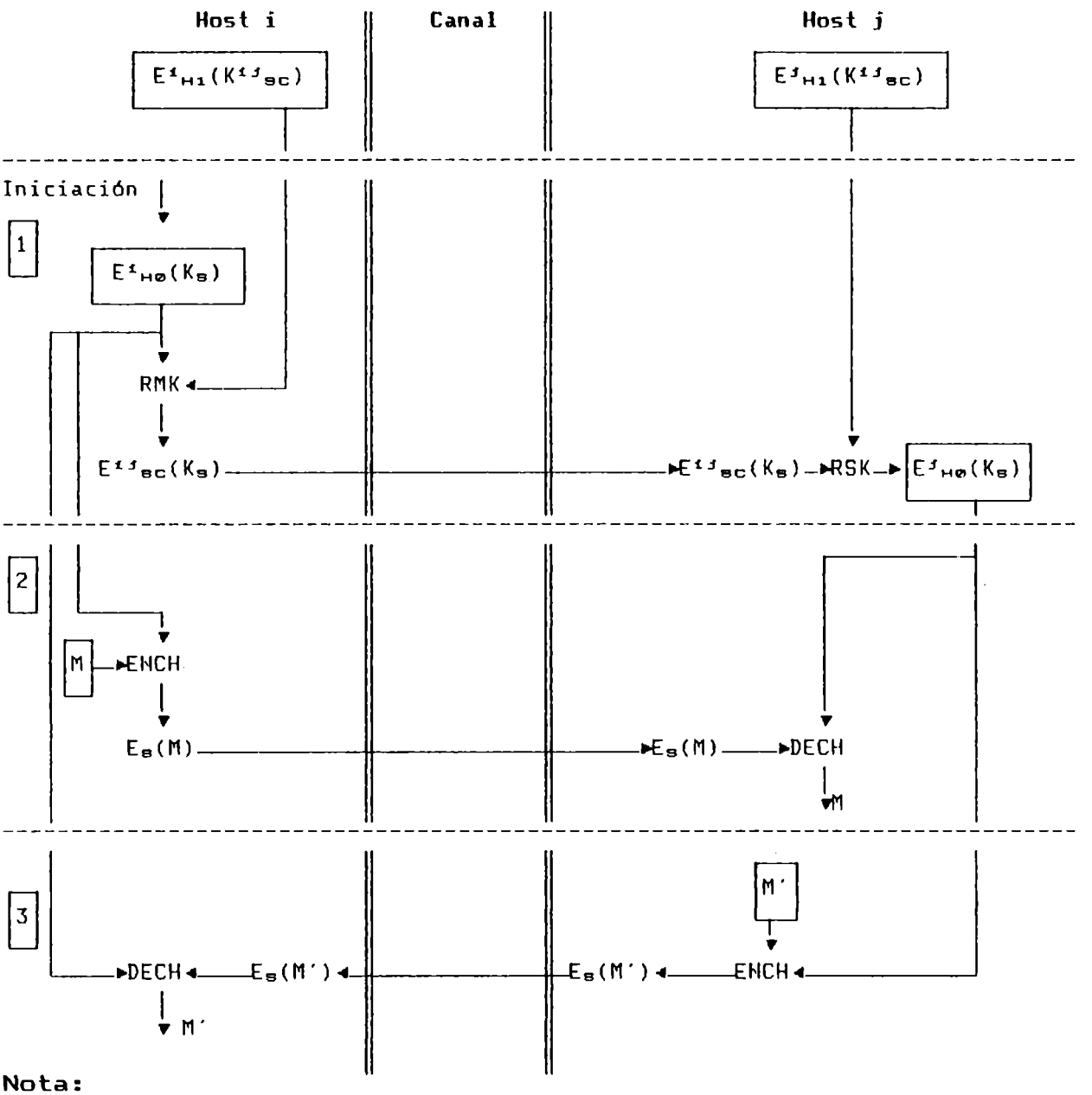


Fig.12: Rencriptado de la clave se sesión vía la instrucción RSK.



Nota:

1-Estado Inicial (cambio de la clave de sesión).
2-Transmisión de un mensaje desde el host i al host j.
3-Transmisión de un mensaje desde el host j al host i.

Fig.13: Esquema de administración de clave durante una transmisión de información entre el Host i y j.

Luego, un criptograma $E^{ij}_{sc}(K_s)$ es enviado via un canal inseguro al host j. El host j tiene que tener la clave K^{ij}_{sc}. Se asume que esta clave es almacenada como un criptograma

$E^{j}_{H1}(K^{ij}_{sc})$, y la clave de sesión K_s es mantenida en la forma $E^{i}_{H0}(K_s)$. Para obtenerla, definimos una nueva instrucción criptográfica llamada *clave de sesión de rencriptado* de la forma:

$$RSK\ [E^{j}_{H1}(K^{ij}_{sc})\ ,\ E^{ij}_{sc}(K_s)] = E^{j}_{H0}(K_s)$$

Un diagrama de la instrucción es dado en la Fig.12. Por supuesto, la instrucción RSK ha sido definida en el host j porque es necesaria cuando el host *j* inicialice la transmisión.

Habiendo obtenido una clave de sesión adecuada (en la forma de criptogramas $E^{i}_{H0}(K_s)$ y $E^{j}_{H0}(K_s)$ mantenida en los host i y j respectivamente) los host pueden transmitir mensajes vía un canal seguro usando las correspondientes instrucciones criptográficas ENCH y DECH (Fig.13).

En resumen, cada host contiene una clave maestra, pero hay dos variantes de estas que son aplicadas. Esta clave crea en nivel más alto de la organización de clave. El segundo nivel comprende las claves terminales maestras y secundariamente las claves de comunicación, las cuales son almacenadas como criptogramas obtenidos por la aplicación de la variante K_{H1} de la clave maestra. El tercer nivel es creado por las claves de sesión, las cuales son protegidas por la aplicación de la variante K_{H0} de la clave maestra (Fig.14).

En los hosts, 4 instrucciones criprográficas no privilegiadas han sido definidas como siguen:

I1) $ENCH\ [M\ ,\ E^{i}_{H0}(K_s)] = E_s(M)$
I2) $DECH\ [E_s(M)\ ,\ E^{i}_{H0}(K_s)] = M$
I3) $RMK\ \ [E^{i}_{H0}(K_s)\ ,\ E^{i}_{H1}(K_T)\ ] = E_T(K_s)$
I4) $RSK\ \ [E^{i}_{H1}(K^{ij}_{sc})\ ,\ E^{ij}_{sc}(K_s)] = E^{i}_{H0}(K_s)$

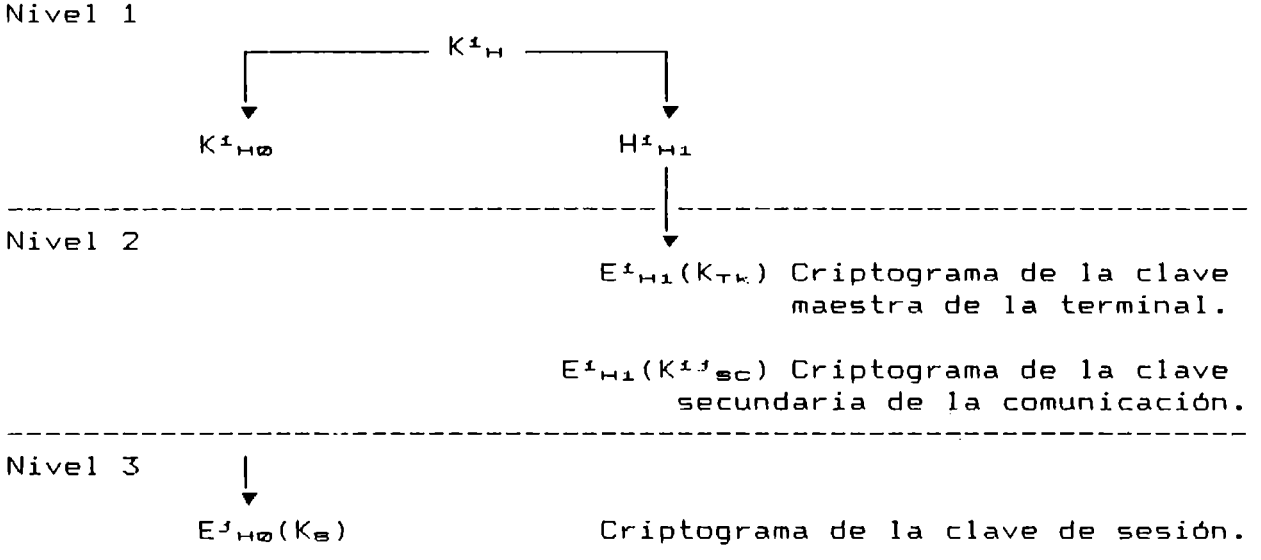


Fig.14: Jerarquía de una clave en el host i.

Las primeras tres instrucciones son usadas cuando los mensajes se transmiten entre un host y su terminal. Pero si queremos introducir protección criptográfica entre hosts, debemos definir una cuarta instrucción RSK. Como hemos visto, las 4 instrucciones criptográficas son aplicadas durante la transmisión de mensajes entre 2 hosts. Por supuesto, tal transmisión solo es posible si ambos hosts prestablecen una clave de comunicación secundaria. Vale la pena notar que ambos hosts comunicados tienen

el mismo acceso al canal. Por eso queremos que ambos hosts puedan enviar y recibir mensajes a través del canal.

2.2 Claves de Transacción

Como el nombre lo indica, una clave de transacción es usada para una única transacción. Las claves de transacción son principalmente aplicadas entre una terminal y un nodo, y no son una alternativa realista para manejar las claves entre los nodos. Por ello en lugar de ser transportadas al otro extremo de un enlace, son generadas separadamente a cada extremo, y preferentemente no transmitidas.

Las entradas usadas para derivar en nuevas claves de transacción son basados en datos reales leídos desde tarjetas plásticas del cliente y los registros almacenados en la terminal son modificados después de cada transacción.

Diferentes claves son generadas para autenticación, encriptado del PIN y encriptado de datos.

Otras Claves

Las claves son derivados de un dato disponible en adición al PIN encriptando:

Clave de Tarjeta. La clave de tarjeta esta basada solo sobre datos ingresados desde la tarjeta de los clientes. Esta clave es conocida como Clave Personal.

Clave de Desacoplo. La clave de Desacoplo es calculada desde la clave tarjeta más la cantidad transacción y el número de verificación de búsqueda del sistema y es específicamente relacionado con una transacción particular. Si 2 transacciones idénticas fueron completados usando la misma tarjeta, la misma clave tarjeta estaría asentada pero la clave de desacoplo estaría cambiada.

Clave de Intimidad. La clave de Intimidad es derivada desde el registro de clave, el identificador de terminal y el número de identificación de búsqueda del sistema, podría ser usado para proveer intimidad (encriptación) de todo o parte de un mensaje. Esta clave es así basada solo en la terminal y el dato de transacción.

Ventajas de la Clave de Transacción

Las ventajas del sistema de clave de transacción son las siguientes:

Realizable de extremo a extremo. Esto es posible usando este sistema por proveer igual protección extremo a extremo del PIN aunque haya algunos enlaces intermedios de tipos de claves de sesión.

Mensaje en cadena. El mensaje en cadena usa el método de resto de MAC provee incremento de seguridad en la selección de mensajes perdidos e insertados y garantía de la correcta recepción de los mensajes previos en una transacción de otro grupo.

Más seguridad para terminal a nodo. Las claves de transacción proveen un ambiente más seguro entre una terminal y un nodo porque no es posible retroceder y la claves son cambiadas en cada transacción.

Futura transacción tarjeta inteligente. Las claves de transacción son adecuadas para proveer transición hacia la futura tecnología de tarjeta inteligente. El esquema de la clave de transacción llegará a ser por si misma más seguro con integrar las tarjetas circuito porque la longitud de otra tarjeta de dato (OCD) que no es transmitida puede ser muy larga.

No se requiere clave de cambio en la transacción. No hay un cambio de clave en la transacción sobre la red o el nodo.

Desventajas de la clave de transacción

Alguna dato de la tarjeta podría no ser transmitida. Por mayor seguridad, algunas Track 2 data desde cualquiera de las claves de transacción calculadas podrían no ser transmitidas en la red. El problema es que hay pocas posiciones disponibles en el Track 2 data que ya no es usado por nadie. Esto es dificultoso para

obtener un acuerdo universal sobre que datos no transmitir. La longitud del campo que no es transmitido, la mayor seguridad proporcionada por el esquema. El número de bytes corrientemente disponibles por la no transmisión propuesta es generalmente considerada también poco para proveer un gran espacio de clave suficiente.

Variación de los requerimientos de transmisión. Estos requerimientos pertenecen a la situación de intercambio, donde una simple adquisición manejaría tarjetas para múltiples usuarios. Diferentes usuarios tienen diferentes requerimientos acerca de cuanto de otras tarjetas de datos -OCD- ellos requieren para transmitir. Cualquier terminal requiere una lista de requerimientos por cada emisor o un indicador sobre la tarjeta respecto de cual dato no es requerido para ser transmitido.

Inicialización. La inicialización es un problema tanto para la transacción de claves así como para la clave de sesión. Una parte inicial de un dato secreto debe ser insertada en la terminal para comenzar la serie. Los problemas lógicos se acrecientan si la terminal está comunicada con múltiples adquirentes.

Nueva generación de terminales. El sistema de clave de transacción requiere del ambiente de una nueva generación de terminales.

Nodo a nodo. Este sistema no ha sido usado para las comunicaciones nodo a nodo. La idea del residuo de MAC podría presentar problemas para grandes volúmenes de enlaces.

Transición desde esquemas (planes) existentes. El mayor número de esquemas existentes en Australia no usa el esquema de transacción de clave. Corrientemente necesita instalar terminales vienen del fin de su vida útil antes debería ser considerado el reemplazo en gran escala. Ellas necesitarán tener un período de superposición entre los diferentes sistemas, lo cual intenta que 2 sistemas adquirentes necesiten ser soportados en paralelo.

2.3 Claves de Sesión

Una clave de sesión es una clave de trabajo usada para protección de datos. En este sistema estas claves son usadas en un algoritmo simétrico y son claves secretas. Estas claves pueden

ser usadas por una o más transacciones y ellas mismas son cambiadas.

Las claves son generadas por un proceso pseudorandom al extremo de un enlace y deben ser transportadas sin riesgo al otro extremo del enlace.

La diferencia esencial entre las claves de sesión y las claves de transacción es que las claves de sesión son usadas para múltiples transacciones y son transportadas encriptadas bajo claves encriptandoras de claves, mientras las claves de transacción son usadas para solo una transacción y son derivadas independientemente a cada extremo del enlace.

El uso de las claves de sesión varía entre los enlaces nodo a nodo y nodo a terminal. En los enlaces nodo a nodo, el proceso es simétrico y cada nodo las crea, envía y recibe claves desde el otro extremo del enlace. Sin embargo, en los enlaces de terminal a nodo el proceso es usualmente, pero no necesariamente, asimétrico. El adquirente usualmente genera todas las claves y las transmite y/o instala a/en la terminal.

Clave jerárquica (capas)

La necesidad de transmitir claves encriptadas bajo otras claves lleva al concepto de una jerarquía de claves. Esto puede ser referido como clave de capas. Comenzando por la capa más baja, están las claves de trabajo o claves de sesión, las cuales son usadas para encriptar datos, PINs, autenticar mensajes y alguna otra función criptográfica ya definida. Cuando estas claves son encriptadas por requerimientos de transmisión o almacenamiento, son encriptadas bajo la siguiente capa, las *claves encriptadoras de clave.*

Las claves encriptadoras de clave siempre deben ser usadas como claves de trabajo y viceversa. Por máxima seguridad esta separación de función siempre debe ser mantenida. En realidad las especificaciones standard de diferentes tipos de claves de trabajo (encriptación de dato, autenticación, etc.) siempre debe ser encriptado bajo diferentes versiones de las claves encriptadoras de clave.

Las claves encriptadoras de clave usadas para transportar claves entre nodos son conocidas como *claves de dominio de transporte.* Es usual tener 2 claves de dominio de cruce por enlace, una para cada dirección. Por lo tanto cada nodo tendrá una clave de envío de dominio de transporte y una clave de recibo de dominio de transporte para cada enlace mantenido con otro nodo.

En la capa superior de la jerarquía está la *clave maestra.*

Esta clave es usada para encriptar las claves encriptadoras de clave donde son requeridas para ser almacenadas fuera de un criptograma seguro, así como un disco. Hay solamente una clave maestra (algunas veces llamada *clave de dominio maestra*) en cada computadora.

Ventajas de la claves se sesión

Operable nodo a nodo. Las claves de sesión son idealmente buenas para operaciones nodo a nodo y hoy en día están siendo usadas para este proposito.

Los métodos pueden ser convinados. En el enlace de terminal a nodo puede usarse un sistema de clave de transacción, mientras que en un enlace de nodo a nodo podría usarse claves de sesión.

Existe tecnología provada. Las claves de sesión están siendo muy usadas y hasta ahora presentan pocos problemas.

Desventajas de la claves se sesión

Las desventajas del esquema de claves sesión son las siguientes:

Seguridad KEK -Key Encrypting Key- en las terminales EFTPOS. La clave encriptada usada para transportar la clave de sesión a la terminal debe ser garantizada en la terminal. Si la clave es extraída desde la terminal por un intruso, el o ella puede decodificar por intercepción todo tráfico en la línea.

De principio a fin. Por el uso del sistema de clave de sesión, es imposible tener la protección del mensaje de principio a fin. La clave de sesión son cambiadas entre nodos adyacentes y por lo tanto el mensaje requiere ser desencriptado en una clave para ser encriptado en otra clave, esto en cada nodo que atraviesa. Amenos que este proceso de traslación es portado por dentro de una facilidad de seguridad, cada uno de estos módulos de control de seguridad, hay exposición de seguridad en cada nodo.

Retrocediendo al último cambio de KEK. Retrocediendo puede portarse a la salida anteriores claves encriptadas.

Inicialización. Inicialización implica la instalación en la terminal de la clave inicial. Este debe ser un proceso secreto y involucra un considerable problema logístico dentro de la arquitectura de la institución. Si la terminal se conecta con múltiples arquitectura entonces el problema se magnifica. La inicialización también puede requerir 1 después de servicio o cuando una terminal pierde sincronismo por mal funcionamiento o problemas en la red.

Cambio de claves. Una transacción separada es requerida para

el cambio de claves. Si este proceso finalizó claramente, entonces será grabada sobre la red y puede incrementar el tiempo de respuesta de la terminal.

Tal ves menos seguro en algunos casos. Aunque el sistema de claves de sesión sea el adecuado para enlazar nodo a nodo, porque estos están generalmente localizados en áreas seguras.

Requiere seguridad de hard en cada nodo. Debe proveer una aceptable seguridad en los nodos intermedios, un costoso hard de seguridad debe ser instalado en cada uno para proveer funciones de traducción.

3 TRANSFERENCIA ELECTRÓNICA DE FONDOS (*EFT*)

Todos los días en este mundo competitivo, muchas entidades financieras buscan atraer clientes a través de nuevas aplicaciones tecnológicas; un buen ejemplo son la instituciones bancarias. Al comienzo de los '70 se empezo a examinar la solución para el acceso a fondos de una entidad financiera por medio de terminales remotas; este procedimiento hoy es llamado *EFT, Transferencia Electrónica* de Fondos.

Los clientes teóricamente pueden acceder a sus fondos vía la terminal de la red bancárea. Por otro lado, son muchas las instituciones financieras que buscan de integrar sus propias redes dentro de una gran red para que el cliente acceda a varias instituciones bancáreas y financieras.

Por lo tanto las redes de computadoras son capaces de proveer todos los servicios de los bancos con terminales orientadas a dicho propósito. Sus habilidades son limitadas a determinadas transacciones. Las cuales pueden proveer el retiro de dinero, vender cheques de viajero, pagar cuentas, facturas, etc.

Entonces muchos problemas de seguridad deben de ser resueltos por la red EFT instalada, por ejem. identificar a los clientes antes de realizar las transacciones. Para que las transacciones sean seguras es común la identificación del cliente a través del uso de un Número de Identificación Personal -PINs- secreto; como las tarjetas bancáreas que portan otra identificación necesaria para las transacciones. Obviamente, esta debe permanecer secreta en todo momento, para todos, dentro de la red EFT.

Usualmente, los poseedores de tarjetas al iniciar una transacción, la introducen en dispositivos especiales, e ingresan su PINs por el teclado de la terminal. Si el PIN es igual al número impreso en el renglón magnético entonces la transacción es

iniciada.

3.1 Número de Identificación Personal (*PIN*)

La protección del PIN y la tarjeta bancárea es crucial para la seguridad del EFT. Las tarjetas pueden ser extraviadas o fraguadas, en estos casos se le deniega acceso al correspondiente PIN como medida preventiva.

La longitud del PIN debe ser larga para que la probabilidad de adivinarla sea baja; pero por otro lado desde ser memorizable por el dueño de la tarjeta bancária. Por eso se estima razonable una longitud entre 4 y 8 dígitos. Asumiendo una longitud de 4 dígitos, se necesitarán miles de pruebas para fraguar le clave. Si el número de ensayos incorrectos es limitado a 5 por día por tarjeta, el enemigo tiene menos de 1:2000 chances; pero al otro día sus chancees incrementan a 1:1000 y así irá incrementando sus posibilidades en los sucesivos días. Entonces, muchos bancos han impuesto un límite absoluto para el ingreso incorrecto del PIN, y cuando se supera esta límite la tarjeta es automáticamente retenida.

El método de selección del PIN es importante para la seguridad dentro del EFT. En general el PIN puede ser seleccionado por el banco o bien por el propietario de la tarjeta, para es primero de los casos los PINs son generados criptoanalíticamente desde el número de cuenta de la tarjeta. La ventaja es que no es necesario guardar el número de cuenta dentro del sistema EFT; pero la desventaja es que un cambio en el PIN es como entrar un nuevo cliente generando una nueva clave criptoanalíticamente.

El principal requerimiento de seguridad es que el PIN deberá ser memorizado por el cliente y jamas deberá almacenarse en algún lugar legible. Pero los humanos no son perfectos y frecuentemente se olvidan de los PIN; entonces los bancos recuperan los PINs o generan uno nuevo.

3.2 *PIN* Orientados a los Sistemas de *EFT*

Supongamos el ejemplo de una transacción, un cliente usando el sistema EFT desea transferir $50.000 de su cuente a otra. Primero inserta su tarjeta bancárea en el dispositivo de la terminal la cual lee primeramente su número de cuenta (PAN) escrito en esta; que este no solo provee el número del cliente sino también la identificación de institución bancárea a la que

pertenece. Luego el cliente ingresa su PIN por teclado y toma el detalle de la transacción. Toda esta información es transmitida al host del banco al cual pertenece el cliente. Finalmente si coincide el PIN y el PAN del usuario la transacción es efectuada y además reduce se cuenta en $50 automáticamente por el uso del servicio.

Del punto de vista de la transacción, la sesión consta de dos pasos:

1ro - La identificación del cliente.
2do - Autenticidad del mensaje.

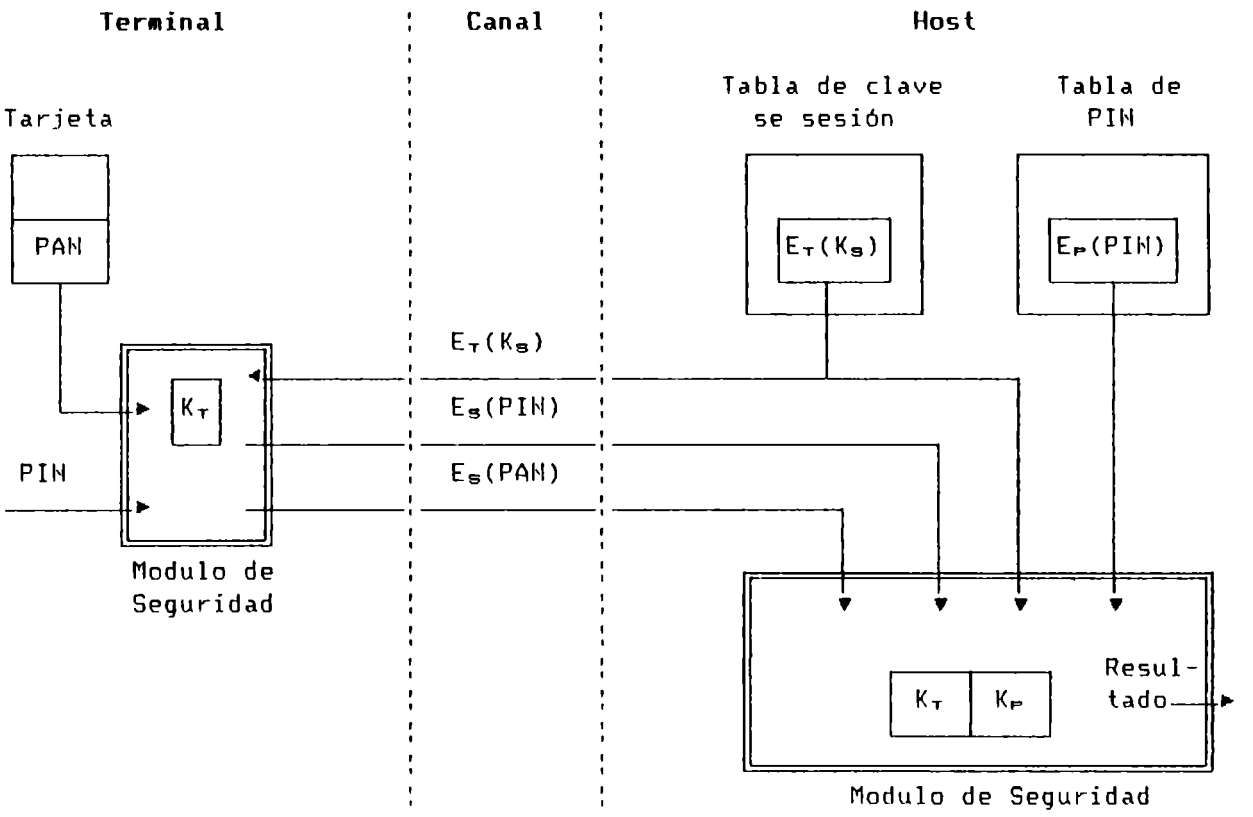


Fig.15: Proceso de PIN y PAN durante una sesión de transición local.

La identificación del cliente, en este simple caso, depende de la correspondencia entre el PIN ingresado por el cliente y el PAN escrito en la tarjeta. Para la autenticidad del mensaje, este porta un Mensaje de Código Automático -MAC-; puede ser tratado

como un resumen mensaje de un mensaje y agregado al mensaje transmitido. Puede validarse la transacción con la edición del par (mensaje y su correspondiente MAC) y la manera conque se generó el MAC.

En general una transacción puede ser procesada desde un local o host remoto. Suponemos que la red usa protección criptográfica para EFT; ver Fig.15. Toda terminal usa un clave maestra (K_T) la cual es cargada en el interior del módulo de seguridad. Así el propietario de la tarjeta tiene activada la terminal desde temprano, el host provee la clave K_S así el criptograma $K_T(K_S)$. El usuario inserta la tarjeta y entra su PIN. El modulo de seguridad teniendo (K_T, $E_T(K_S)$, PIN, PAN) produce los criptogramas $E_S(PIN)$ y $E_S(PAN)$ aplicando la adecuada instrucción criptográfica.

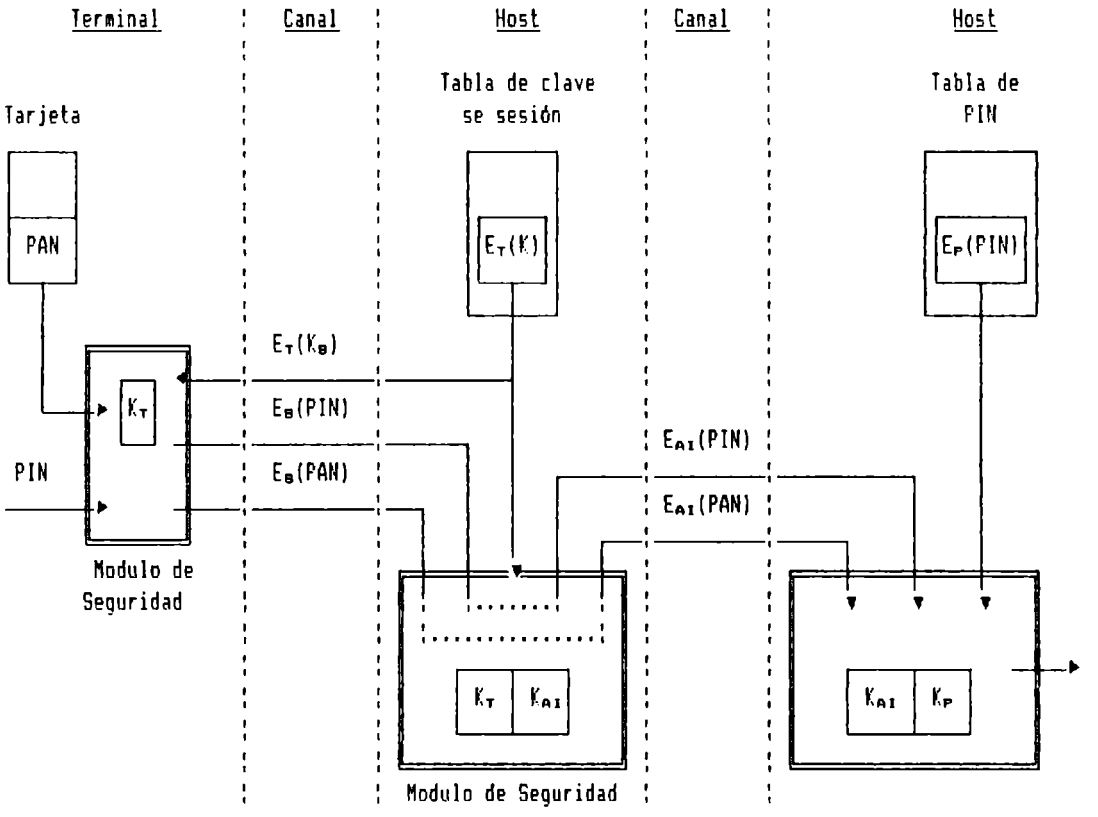


Fig.16: Proceso de PIN y PAN durante una transición de intercambio de sesión.

El par [($E_S(PIN)$; $E_S(PAN)$)] es transmitido al host. Como

los PINs no pueden ser almacenados en forma clara (desencriptada) en cualquier parte, el host conserva todos los PINs como $E_{K_P}(PIN)=E_P(PIN)$, donde K_P es la clave maestra de PIN (todos los PIN son cargados usando una simple clave maestra PIN). La cuádruple $[E_B(PIN),E_B(PAN),E_P(PIN),E_T(K_B)]$ se cargan en el modulo de seguridad del host. Conociendo la clave maestra de la terminal K_T y la PIN clave maestra K_P, el modulo descrito requiere criptogramas y comparar las PINs. Si ellos no son iguales o el PIN no se corresponden con el PAN, la identificación del poseedor de la tarjeta es negativa.

Una transacción requiere de que el mensaje sea protegido usando alguna clave de sesión y entonces MAC no es necesario. Pero si el mensaje es transmitido como texto plano, este MAC debe ser encriptado en orden y el host debe ser capaz de autenticar el mensaje.

La transacción con un host remoto es similar a lo visto, ver Fig.16. Asumimos que hay un canal seguro, creado por la clave interhost K_{AI}, en medio del adquirente y el emisor. El proceso es el mismo hasta los criptogramas $E_B(PIN),E_B(PAN)$, $E_T(K_B)$ alcanzando el modulo de seguridad del adquirente. El PAN indica el emisor así el modulo rencripta PIN y PAN dentro de $E_{AI}(PIN),E_{AI}(PAN)$ usando la clave de interhost K_{AI}. Después de que los criptogramas han sido insertados dentro del modulo de seguridad emisor, este trabaja saliendo la decisión final. La transacción requiere mensajes transmitidos como antes.

3.3 *PIN* y Clave Personal Orientado a Sistemas *EFT*

Asumiendo que las terminales son seguras, la correcta inicialización de los procesos depende de la seguida de los PINs y las tarjetas. Para aumentar la protección de un sistema EFT, especialmente durante la inicialización, podemos o alargar los PINs o introducir tarjetas cuya duplicación es dificultosa y cara. Investigaciones han mostrado que si la longitud del PIN es de 8 digito decimales de largo, el número de inicialisaciones erróneas se incrementa debido a la dificultad de memorizar los PINs.

Cada tarjeta debe contener más información. Considerar una solución en la cual una tarjeta contiene dos mensajes impresos: el número de cuenta PAN y la clave personal -PK: Personal Key-, ver Fig.17. PK es seleccionada aleatoriamente por el emisor y la longitud depende del algoritmo de encriptado utilizado. Si DEA es usado, el PK es de 56 bit de largo. En la etapa de inicialización, el propietario de la tarjeta inserta su tarjeta y

entra su PIN. El par (PIN y la clave personal PK) es sumada modulo 2 bit a bit. El resultado es aplicado como una clave encriptada por el PAN. El criptograma $E_{PK+PIN}(PAN)$ es enviada vía un canal seguro (establecido por una sesión de clave) para el emisor quien compara dentro de la computadora original.

Otra solución, una tarjeta bancárea incluye dos mensajes PAN, PK como antes y uno tercero llamado código de autenticidad PAC. Mientras la tarjeta esta siendo emitida, un PAC es creado como:

$$PAC = E_{IK}(E_{PK+PIN}(PAN))$$

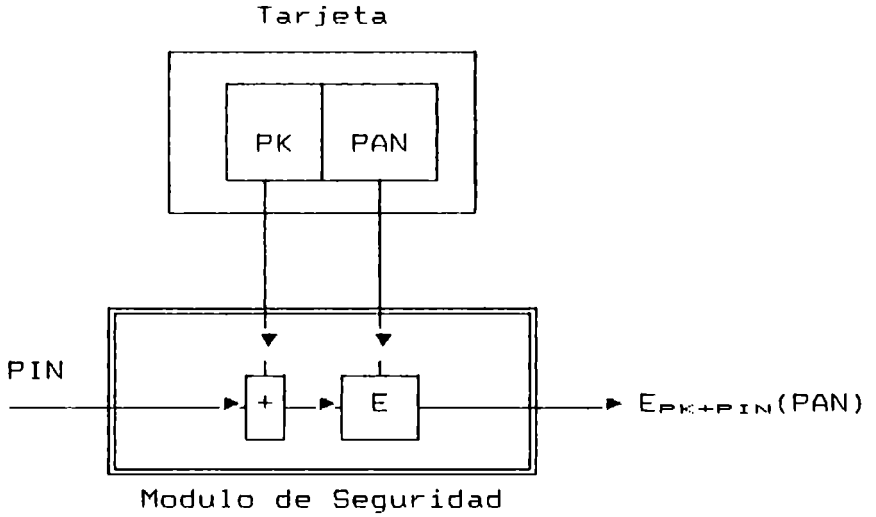


Fig.17: Proceso de la PIN y el mensaje de la tarjeta durante la inicialización.

Donde IK es la clave secreta emitida. En la etapa de inicialización, después un propietario de la tarjeta provee un PIN y entrado en su tarjeta, el par ($E_{PK+PIN}(PAN)$ y PAC) es transmitido vía un canal seguro. Entonces emisor usa la clave secreta, IK, para calcular la referencia PAC*. Si PAC=PAC*, el propietario de la tarjeta acepta como genuina.

Comparando con la primera, la segunda solución tiene la ventaja que esta verifica un propietario de tarjeta, el emisor aplica la única clave secreta, IK, para toda la sesiones. En la primera solución, sin embargo, el emisor puede guardar un registro de criptograma $E_{PK+PIN}(PAN)$ para todos los clientes.

3.4 Requerimientos de Protección en Sistemas *EFT*.

En las primeras etapas del desarrollo de los EFT, las instituciones financieras introdujeron seguridad basado en un extenso y diferente estándar. Sin embargo la tendencia a integrar un simple sistema EFT dentro a sistemas nacional de EFT. Las ventajas de tener múltiples sistemas son obvias; fácil acceso a recursos financieros para algunos clientes, en cualquier ciudad y simultáneamente la chance de incrementar el rango de servicios bancarios.

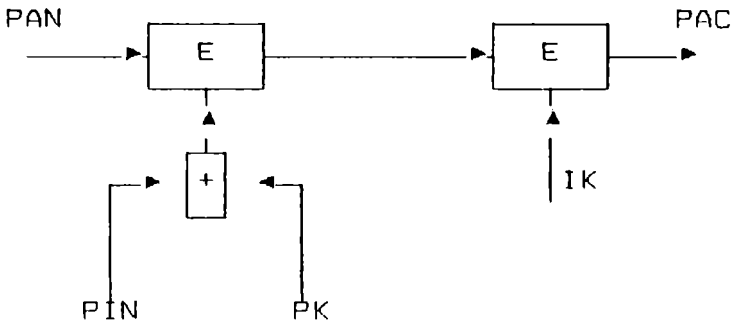


Fig.18: Un esquema de la generación del PAC.

Para facilitar la integración de procesos no solo dentro de una ciudad pero también entre diferentes ciudades, hay una urgente necesidad para preparar un standard nacional y internacional. El básico elemento de un sistema de protección para EFT es el algoritmo de encriptación. Hasta ahora a sido común que la DEA (DES) debería adoptarse como una encriptación estándar internacional. Esto es ahora obvio que la DEA quiera hacerse insegura en un futuro próximo. Por lo tanto, es posible adoptar:

- Una algoritmo de clave asimétrica (sistema RSA).
- Un nuevo algoritmo simétrico (similar al DEA (DES) pero basado en 128 bits).

La integración de procesos es esencial para la seguridad del sistema EFT y debe ser completo dentro de cada camino provisto por los EFT subsistemas (que posee un simple institución financiera) es independiente de la medida de seguridad que garantiza otro subsistema EFT. Al mismo tiempo, integrar sistemas de terminales para EFT (ATM automatic teller machine o POS point-

of-sale), debe ser capaz de ser usado por clientes de diferentes instituciones financieras (bancos, sociedades, etc.).

El punto más vulnerable de un sistema EFT son las terminales de entrada, estas pueden ser accesibles por todas las personas y en algún momento ellos deben ser capases de distinguir un legítimo usuario de un intruso. La identificación de un usuario en un sistema EFT debe ser confiable, eficiente y barato. La identificación es confiable si la probabilidad de negarle el servicio a un legítimo cliente y garantiza el servicio a un intruso son las bajas, estas entradas son previstas por los diseñadores de los EFT. Una eficiente identificación puede ser cumplido por medio de un mínimo número de referencias a la publicada.

Una identificación puede ser portadora de alguna cosa que identifique al usuario por características personales (impresión digital, voz, cara, etc.) son muy caros e incluso algunas desconfiarles. Esto es por que no puede ser aplicado en una terminal estándar.

Hay varios requerimientos de seguridad, respecto a la identificación del usuario, dados en PINs y tarjetas bancáreas, son la siguientes:

- El PIN permanecer en secreto en todo tiempo.
- El PIN no debe ser escrito en la tarjeta de crédito.
- Una tarjeta deberá contener un único número PAN el cual debe relacionarse con el PIN.
- Si la tarjeta contiene el código de autenticidad personal (PAC),este debe depender del PIN, pero recobrar este a partir de la base del PAN debe ser impracticable.
- El proceso de identificación debe depender de la hora (para frustrar el playback).

El próximo conjunto de protecciones se refieren a una 2ª de la comunicación la cual involucra un código de autenticidad de mensajes (MAC). Los requerimientos son los siguientes:

- Un MAC debe contener una descripción de la transacción (una transacción requiere de un mensaje), el PIN del propietario de la tarjeta, datos corrientes y hora.
- El cálculo de un PIN debe ser difícil de manejar desde la base del MAC.
- El instante de referencia del MAC debe posibilitar una determinada independientemente por el emisor.
- Un mensaje/s involucrados durante la identificación del propietario de la tarjeta debe ser representado asociando

criptográficamente con el mensaje generando dentro de la 2da etapa de una sesión de comunicación.

Permitimos retorno para la seguridad de la terminal. Deberíamos recalcar la necesidad de una seguridad física. Tal que la seguridad debe ser provista en todas las terminales en su modulo de seguridad el cual cada uno preserva la clave maestra y la de sesión (transacción). La clave maestra de la terminal es cargada en forma limpia. Sin embargo, la clave de sesión K_s es transmitida desde la terminal al host y almacenada en forma de criptograma, $E_T(K_s)$, encriptadas dentro de la terminal. Así toda operación criptográfica son ejecutadas dentro de el modulo de seguridad, toda información generada por el conjunto (teclado - PIN, tarjeta leída - PAN, PK, o/y PAC) son transmitidas como texto sencillo a este. Es innecesario decir el flujo de información desde la entrada hasta la salida de la terminal y el modulo de seguridad debe ser cuidadosamente designado y procesado.

Un paso futuro para aumentar la seguridad del sistema EFT ha sido realizado por inclusión de tarjetas inteligentes. La aplicación de estas tarjetas pueden virtualmente eliminar la desprotección del flujo de información desde el lector de la tarjeta y el modulo de seguridad en diferentes etapas.

4 CONCLUSIÓN

4.1 Conclusiones Generales

- La seguridad en una red de computadoras se logra mediante criptografía y el uso de adecuados protocolos de comunicación, los cuales deben garantizar la autenticación del usuario y establecer un canal seguro.
- La protección del canal se hace por medios criptográficos de la información, esta depende de las claves utilizadas y que sean cambiadas con cierta regularidad.
- Las terminales o puestos de trabajo y los host poseen una Clave Maestra que reside en forma limpia o clara, usada para encriptar y desencriptar el resto de las claves y/o mensajes; por lo tanto es crucial la seguridad del hardware donde esta se alojará.
- Las terminales o puesto de trabajo dependen de distintos Centros de Distribución de Claves o Directorios de Claves - CKD- organizados jerárquicamente que residen en una computadora host. Caso contrario cada terminal o nodo debería

tener almacenado en memoria cada una de las claves de acceso de los restantes puntos de la red; siendo esto poco seguro y complejo de administrar.

» Tanto al iniciar una comunicación como al ejecutar sus funciones, se requiere de la misma cantidad de hardware, software y tiempo. Por lo tanto los algoritmos simétricos y asimétricos proporciona soluciones equivalentes.

» La información viaja a trabes de la red en paquetes (frame) donde ciertos datos (por ejem.: los de cabecera que hace al errutamiento) no pueden ser encriptados; por lo tanto se hace inevitable el desvío de un paquete por parte de un intrusos.

» La utilización claves de secundaria (sesión, transmisión, tarjeta, desacoplo, etc.) si son usadas en forma indiscriminada torna complejos los protocolos, poco confiable y reduciría la performace de la comunicación.

» En los EFT es crucial la seguridad sobre el PIN y la correcta memorización por parte de los usuarios. La seguridad de los EFT es reforzada con el número de cuenta bancaria o el uso de mensaje de código automático.

4.2 Protocolo Básico de Distribución de Claves

Se tiene por objetivo el diseño de un esquema básico para establecer un enlace seguro ente dos usuarios A y B conectados en las terminales (remotas o no) T1 y T2 respectivamente. Teniendo en cuenta:

» Atenticación del usuario. Validar que la persona conectada no sea un intruso.

» Establecer un canal seguro. Garantizar que ningún intruso no pueda interpretar o decodificar la información transmitida por un canal.

Para ello consideramos:

1ro Centros de Distribución de Claves -CKD- ordenados jerárquicamente.

2do Nos abstraemos del medio de comunicación, arquitectura y protocolos de comunicación de la red.

3ro Consideraremos algoritmos criptográficos simétricos.

4to Cuando un usuario se conecta, establece una cesión de trabajo, generandose para esta una clave adecuada y finaliza con la desconección del usuario.

5to Denominaremos con mayúscula las claves maestras -K_{Mi},

K_{Tp}- y con minúscula las compartidas -K_{hi}, K_{tp}-.

6to La terminal posee su identificación -ID_{Tp}-, sus claves K_{Tp} y K_{tp}, la clave compartida con el host K_{hi}.

7mo La identificación de las terminales como la de los host puede estar formada por una nomenclatura estándar identificatoria, más un código de autenticación.

7mo No consideran los mensajes de control.

El usuario al conectarse se identifica -ID_A- en la terminal Tp (ver Fig.19), cuando este genera un requerimiento explícita o implícitamente, es interpretado por la terminal la cual genera un requerimiento -RQ_{Tp}- y envía el siguiente criptograma al host bajo su clave compartida.

$$E_{hi}(ID_A, ID_{Tp}, RQ_{Tp})$$

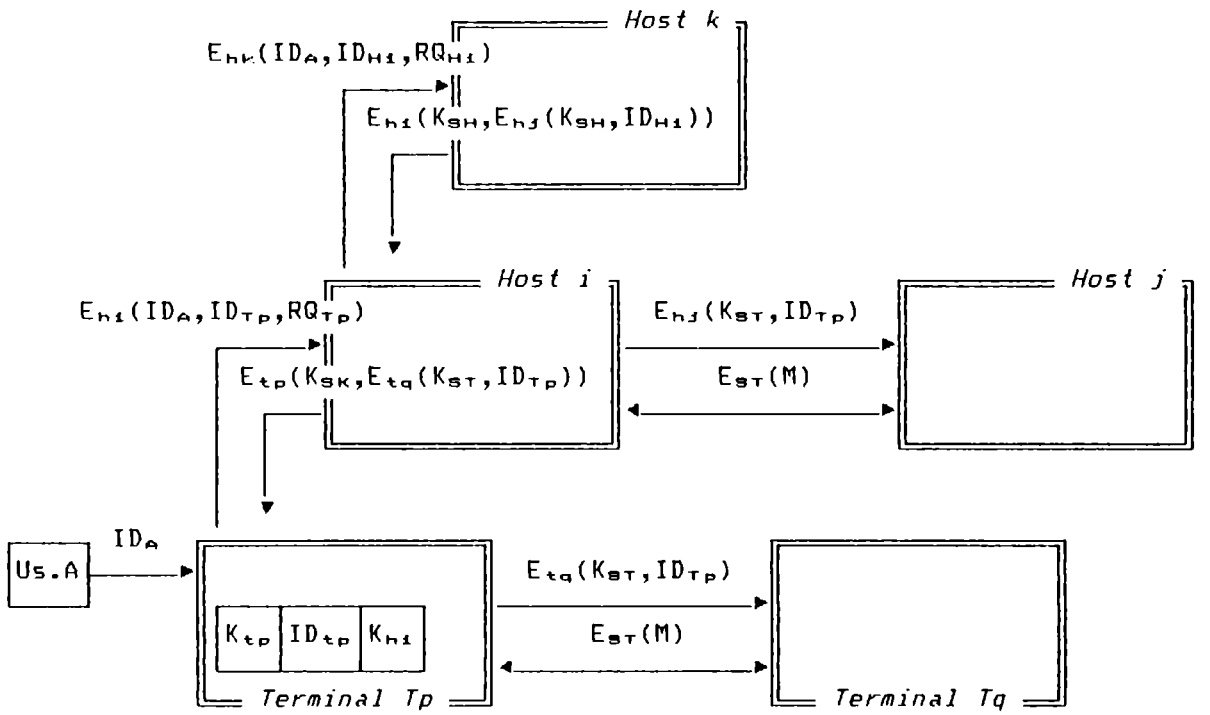


Fig.19: Esquema Básico de Distribución de Claves.

El host al desencriptar obtiene ID_A y ID_{Tp} para la autenticación de ambos, interpreta RQ_{Tp}, generando uno de los

siguiente criptogramas:

I $E_{tp}(K_{ST},E_{tq}(K_{ST},ID_{Tp}))$. Para establecer un enlace a otra terminal de su ambiente.

II $E_{Hk}(ID_A,ID_{Hi},RQ_{Hi})$. Para establecer un enlace con un host el cual posea un CKD de nivel superior.

Si RQ_{Tp} solicita establecer una comunicación con la terminal Tq dependiente del host Hi, este genera el criptograma -*I*- con una clave para la sesión K_{ST} y el criptograma bajo la clave K_{tq} (terminal con que se desea establecer el enlace) conteniendo la clave de sesión y la identificación de la terminal emisora. La terminal Tp desencripta -*I*- obteniendo K_{ST} y almacenándola en forma segura, luego transmite el criptograma $E_{tq}(K_{ST},ID_{TP})$; de esta manera la terminal Tq obtiene la clave de sesión, estableciéndose un canal seguro para la comunicación entre ambas.

Si RQ_{Tp} requiere establecer una comunicación con el host Hj, el host Hi genera el criptograma -*II*- al host Hk poseedor un CKD de jerarquía superior. El host Hk desemcritpa y obtiene ID_A y ID_{Hi} para la autenticación, interpreta RQ_{Hi} y genera el siguiente criptograma de respuesta bajo la clave pública del host Hi:

$$E_{Hi}(K_{SH},E_{Hj}(K_{SH},ID_{Hi}))$$

El host Hi obtiene una clave para la sesión K_{SH} y el criptograma bajo la clave K_{Hj} (host con que se desea establecer el enlace). El host Hj desencripta $E_{Hj}(K_{SH},ID_{Hi})$ obteniendo K_{SH} y almacenandolo en forma segura, obteniendose un canal seguro para la comunicación entre ambos host.

4.3 Esquema Funcional

Para este esquema, ver Fig.20, consideraremos 8 funciones básicas:

1-Encrip tar Requerimiento de Comunicación

$$ERQC(E_{Tp}(ID_A),E_{Tp}(ID_{Tp}),RQ_{Tp},E_{Tp}(K_{Hi}))=E_{Hi}(ID_A,ID_{Tp},RQ_{Tp})$$

$$ERQC(E_{Hi}(ID_A),E_{Hi}(ID_{Hi}),RQ_{Hi},E_{Hi}(K_{Hk}))=E_{Hk}(ID_A,ID_{Hi},RQ_{Hi})$$

Función por la cual la *Terminal Tp* o el *Host Hi*, respectivamente, encripta un requerimiento de comunicación.

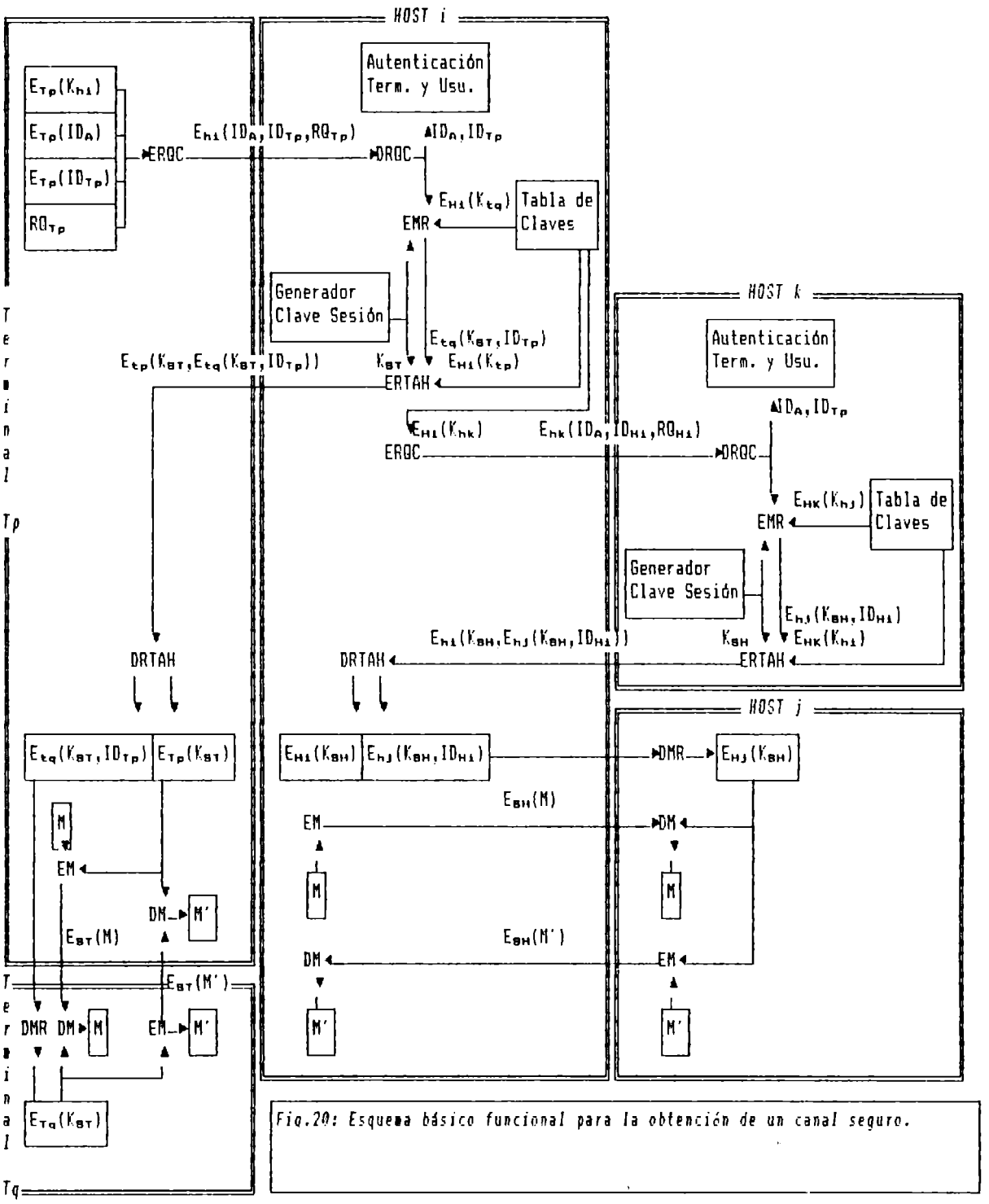


Fig.20: Esquema básico funcional para la obtención de un canal seguro.

2-Desencriptar Requerimiento de Comunicación

$$DRQC(E_{Hi}(ID_A, ID_{Tp}, RQ_{Tp}), E_{Hi}(K_{Hi})) = \left| \begin{array}{l} ID_A \\ ID_{Tp} \\ RQ_{Tp} \end{array} \right.$$

$$DRQC(E_{Hk}(ID_A, ID_{Hi}, RQ_{Hi}), E_{Hk}(K_{Hk})) = \left| \begin{array}{l} ID_A \\ ID_{Hi} \\ RQ_{Hi} \end{array} \right.$$

Función donde el *Host Hi* o el *Host Hk* obtine en forma limpia la información para autenticación y el requerimiento de la terminal para su interpretación.

3-Encriptar Mensaje al Receptor

$$EMR(K_{SH}, ID_{Tp}, E_{Hi}(K_{Tq})) = E_{Tq}(K_{SH}, ID_{Tp})$$

$$EMR(K_{SH}, ID_{Hi}, E_{Hi}(K_{Hj})) = E_{Hj}(K_{SH}, ID_{Hi})$$

Función que genera el criptograma en el host *i* o *k* a ser desencriptado en el nodo receptor, la *Terminal Tp* o el *Host j*, respectivamente.

4-Desencriptar Mensaje en el Receptor

$$DMR(E_{Tq}(K_{ST}, ID_{Tp}), E_{Tq}(K_{Tq})) = \left| \begin{array}{l} E_{Tq}(K_{ST}) \\ E_{Tq}(ID_{Tp}) \end{array} \right.$$

$$DMR(E_{Hj}(K_{SH}, ID_{Hi}), E_{Hj}(K_{Hj})) = \left| \begin{array}{l} E_{Hj}(K_{SH}) \\ E_{Hj}(ID_{Hi}) \end{array} \right.$$

Función que desencripta el mensaje para el receptor *Terminal Tq* o *Host j* respectivamente. Obteniendo la clave de sesión y la identificación del nodo emisor de modo tal de poder establecer un canal seguro para la comunicación.

5-Encriptar Respuesta del Host

$$ERTAH(K_{ST}, E_{Tq}(K_{ST}, ID_{Tp}), E_{Hi}(K_{Tp})) = E_{Tp}(K_{ST}, E_{Tq}(K_{ST}, ID_{Tp}))$$

$$ERTAH(K_{SH}, E_{Hj}(K_{SH}, ID_{Hi}), E_{Hk}(K_{Hi})) = E_{Hi}(K_{SH}, E_{Hj}(K_{SH}, ID_{Hi}))$$

Función que encripta la respuesta del host *i* o *k* para el requerimiento de comunicación de la *Terminal Tp* o el *Host Hi* respectivamente.

6-Desencriptar Respuesta del Host

$$DRTAH(E_{tp}(K_{ST},E_{tq}(K_{ST},ID_{Tp}),E_{Tp}(K_{tp}))=\begin{cases} E_{Tp}(K_{ST}) \\ E_{tq}(K_{ST},ID_{Tp}) \end{cases}$$

$$DRTAH(E_{hi}(K_{SH},E_{hj}(K_{SH},ID_{Hi}),E_{Hi}(K_{hi}))=\begin{cases} E_{Hi}(K_{SH}) \\ E_{hj}(K_{SH},ID_{Hi}) \end{cases}$$

Función por la cual la *Terminal Tp* o el *Host i* obtienen la clave de sesión y el criptograma para la *Terminal Tq* o el *Host j*

7-Encriptar Mensaje

$$EM(M,K_{Tp}(K_{ST}))=E_{ST}(M)$$

$$EM(M',K_{Tq}(K_{SH}))=E_{ST}(M')$$

Función por la cual la *Terminal Tp* encripta el mensaje M o la *Terminal Tq* encripta el mensaje M', respectivamente. En los host *i* y *j* es equivalente.

8-Desencriptar Mensaje

$$DM(E_{ST}(M),K_{Tq}(K_{ST}))=M$$

$$DM(E_{ST}(M'),K_{Tp}(K_{ST}))=M'$$

Función por la cual la *Terminal Tq* desencripta y obtiene el mensaje M o la *Terminal Tp* obtiene el mensaje M', respectivamente. En los host *i* y *j* es equivalente.

5 ABREVIATURAS

ATM: Automatic Teller Machine.
CKD: Center of Key Distribution.
EFT: Electronic Fund Transfer.
$E_{M,B}$ ó E_B: Criptograma bajo la clave K_B.
DM: Desencriptar Mensaje.
DMR: Desencriptar Mensaje de Respuesta.
DRQC: Desencriptar Requerimiento de Comunición.
DRTAH: Desencriptar Respuesta del Host.
EM: Encriptar Mensaje.
EMR: Encriptar Mensaje de Respuesta.
ERQC: Encriptar Requerimiento de Comunicación.
ERTAH: Encriptar Respuesta del Host.
ID_A: Identification User A.
ID_B: Identification User B.
K_{AB}: Clave de identificación entre los usuarios A y B.
K_A: Key of A or Key public of A.
K_B: Key of B or Key public of B.
K'_A: Key private of A.
K'_B: Key private of B.
K_D: Key Directory; clave pública del CKD.
KEK: Key Encrypting Key.
K_H: Key Host.
K^{ij}_{SC}: Clave de sesión secundaria entre el host i y j.
K_S: Key Session.
K_T: Key Terminal.
MAC: Mensaje Autentication Code.
N_A: Name A.
PAC: Presonal Autentication Code.
PAN: Persinal Number Acces ó número de cuenta.
PIN: Personal Identification Number.
PK: Personal Key.
POS: Point-of-Sale terminal.
R_A ó RQ_A: Request A.
R_S ó R_S : Secuencia aleatoria.

I I I FACTORES QUE AFECTAN LA PERFORMANCE DE LA RED

Contenido Páginas

FACTORES QUE AFECTAN LA PERFORMANCE DE LA RED

Los factores que afectan la performance de la red están relacionados con los elementos de hardware y software que la componen.

1 COMPONENTES DE HARDWARE

- Subsistema de disco: velocidad y capacidad de almacenamiento.
- Adaptadores de red: velocidad, ancho del bus, interfase del bus.
- Tecnología de acceso a la red.
- Velocidad del servidor y las estaciones .

La performance no solo se ve afectada por el hardware del servidor, también influyen el tipo de adaptador de red del servidor y las estaciones, los métodos y velocidades de acceso a la red (CSMA/CD,Token Access, etc.). Los elementos más importantes son el subsistema de disco, y el tipo de adaptador de red dentro del servidor. Cualquier mejora sobre el hardware del servidor afecta directamente la performance de la red.

1.1 Subsistema de disco

La función más importante del servidor de NetWare es proveer servicios de archivos. Los archivos del servidor son almacenados en el disco del servidor, un subsitema de disco veloz da como resultado rápidos servicios de archivos, y desde el punto de vista del usuario, una red rápida.

La velocidad de transferencia de datos entre la CPU y el disco está determinada por el disco, el controlador de disco, y la interfase del bus.

Los discos que se usaban antes en las PC AT utilizaban codificación MFM (Modulación de Frecuencia Modificada). En la actualidad los discos más usados usan codificación RLL (Longitud Limitada de Ejecución: empaquetan un 50% más de bits en el mismo espacio que el método MFM).

Este método de codificación magnético RLL, se utiliza con las interfases:

IDE (Integrate Device Electronics) Electrónica de Unidades Integradas: Unidad de disco que contiene su propia electrónica de control, eliminando la necesidad de expansión adicional en la computadora.

ESDI (Enhanced Small Devices Interface) Interfase Mejorada de Dispositivos Pequeños : Transfiere datos a una velocidad de 3Mb/seg. Considerado como una interfase de alto rendimiento y alta calidad para computadoras.

SCSI (Small Computer System Interface) Interfase Pequeña de Sistemas de Computadoras: Interfase para más de siete periféricos (disco, cinta, CD ROM, etc). Es un interfase de bus de 8 bits para más de ocho dispositivos, pero el adaptador principal, que se conecta al bus de la computadora, también cuenta como un dispositivo. El bus SCSI permite que dos dispositivos se comuniquen a la vez (de principal a periféricos, de periférico a periférico).

SCSI-1 proporciona una velocidad de transferencia de datos de hasta 5MBytes/seg y puede conectar muchos periféricos mientras que toma solamente una ranura de expansión en la computadora.

SCSI-2 proporciona colas de órdenes y una opción síncrona "Fast SCSI" que proporciona una transferencia de datos a 32MBytes/seg.

El dispositivo SCSI puede transferir a una velocidad de 32Mbps, la que es mejor que la velocidad de Ethernet (10Mbps), Token Ring (16Mbps), o ARCnet PLUS (20Mbps). En la actualidad, estos dispositivos logran velocidades de transferencia de datos menores si se usan con bus del tipo ISA[1], debido a que el bus ISA está limitado a la velocidad de transferencia de 4Mbps a 8Mbps. Por lo tanto, para mejorar la performance, es conveniente utilizar dispositivos SCSI con bus MCA[2] o bus EISA[3] que

---

[1]Industry Standar Architecture - Arquitectura de Estándar Industrial.

[2] Micro Channel Arquitecture - Arquitectura Micro Canal

[3]Extended Industry Standar Arequitecture - Arquitectura Extendida de Estándar Industrial.

soportan altas velocidades de transferencia de datos.

Hay un nuevo tipo de subsistema de disco, RAID. (Redundant Arrays of Inexpensive Disks), es un agrupamiento de discos en la que los datos se copian en muchas unidades. Proporciona un caudal más rápido, tolerancia de fallos (espejos) y corrección de errores.

Especificaciones del disco duro

Interfase	Codificación*	Velocidad de Transferencia (Bytes/seg)	Capacidad Típica (Bytes)
ST506	MFM	625K	5M - 100M
ST506 RLL	RLL	937K	30M - 200M
IDE	RLL	.625-2M	20M - 500M
ESDI	RLL	1-3M	80M - 2G
SCSI-1	RLL	1-5M	20M - 1.5G
SCSI-2	RLL	1-40M	40M - 3G

*(La mayoría de los discos utilizan RLL, pero el método de codificación no va prescrito con la interfase).

1.2 Adaptadores de Red (NICs)

En un ambiente de red, los adaptadores de red determinan la velocidad de transferencia de datos a través de la red.

Los siguientes aspectos afectan la performance de un adaptador de red:

- Esquema de acceso al medio
- Velocidad de bit cruda
- Procesador en la tarjeta
- Transferencia host-to-Nic (Network Interface Card)

El esquema de acceso al medio se refiere al mecanismo de arbitraje, inherente en las LANs de banda base, que limita como los datos son situados sobre el medio en cada momento. Los ej. de esto son CSMA/CD utilizado en Ethernet e IEEE 802.3, y Token Access usado en IBM Token Ring (IEEE 802.5). El Token Access da una performance determinística a una baja carga de la red, mientras que CSMA/CD es susceptible a reducir su rendimiento debido a colisiones bajo carga fuerte. Bajo cargas livianas, por otro lado, el método de acceso CSMA/CD es más simple y rápido que Token Access.

La velocidad de bit cruda es la velocidad de bit máxima posible sobre un medio dado. La velocidad de bit efectiva (teniendo en cuenta los retardos por overhead de protocolo y colas y procesamiento) es mucho menor. La velocidad de bit cruda representa el límite superior para el medio.

El uso de un procesador en la tarjeta puede incrementar la velocidad del adaptador de red. Si el chip de memoria para la NIC es pobremente escrito, puede tener efectos opuestos a los que se conocen en los anteriores adaptadores LAN de banda ancha de IBM PC. Algunos distribuidores implementan protocolos para las capas superiores que procesan sobre la misma tarjeta NIC mejorando sobre todo el rendimiento efectivo.

Los datos que llegan al adaptador de red necesitan memoria dentro de la computadora host para ser transferidos. La conexión entre el NIC y el canal del host puede ser implementado usando memoria compartida, DMA (Acceso directo a memoria), o puertos de E/S. Los NICs pueden usar uno de estos métodos o una combinación de ellos. Las observaciones muestran que memoria compartida es lo más rápido, le sigue los puertos de E/S, y luego DMA como lo más

lento. Entonces se debe evitar NICs que usen exclusivamente DMA para transferir datos ya que son probablemente los más lentos. Algunas tarjetas usan una combinación de memoria compartida y DMA, que les da un cierto nivel de paralelismo que puede mejorar su performance.

El ancho del bus de datos de la interfase tiene un dramático efecto sobre la velocidad de transferencia host-to-NIC. Los anchos son de 8, 16, o 32 bits.

EISA y Micro Channel NICs son más rápidos que NICs ISA. Es una buena estrategia utilizar EISA o placas Micro Channel para el servidor , por que el tráfico de la red se concentra en este punto.

Hay que tratar de evitar mezclar adaptadores de red de diferentes fabricantes sobre una LAN. Aunque todos los fabricantes pretender seguir un standard, existen entre ellos importantes diferencias de implementación que pueden afectar la performance de la red. Algunos fabricantes de placas Ethernet implementan algoritmos de reintento de tiempo aleatorio de diferente formas.

La utilización de la red se define como el porcentaje de tiempo consumido en transferir datos, sin incluir el tiempo consumido en procesar los paquetes y en el retardo de la transmisión. Uno puede estimar la máxima utilización factible bajo la mejor condición posible, considerando cero retardo en el proceso y colas de espera, pero no puede evitar el retardo de transmisión debido a la velocidad de propagación de la señal.

El tiempo de transmisión del dato (Tx) es para una transmisión dada el tiempo que tarda en entregarse el dato por la LAN. El retardo de transmisión (Td) depende de la velocidad de propagación finita de la señal en la LAN. El mensaje consiste de un tiempo de uso del canal Tx y tiempo de retardo de la propagación Td. La utilización del canal se obtiene utilizando la siguiente formula:

$$U = Tx / (Tx + Td)$$
$$= 1 / (1 + Td / Tx)$$
$$U = 1 / (1 + a) \qquad \text{donde } a = Td / Tx.$$

Si la velocidad de transmisión de la LAN para un paquete de longitud P bits es D bps, entonces:

Tx [segundos] = P / D

Si la velocidad de propagación de la señal en el medio es V m/seg y la longitud de la LAN es L metros, luego

Td [segundos] = L / V

La longitud promedio de los paquetes para la mayoría de las aplicaciones LAN es de 128 Bytes.

En LAN 10BASE-T (Fig. A), la máxima longitud de la LAN puede ser 200 metros, sin incluir los cables de interconexión de las plaquetas (backplane). El backplane del bus CSMA/CD en el concentrador de la 10BASE-T tiene una longitud de alrededor de 0.5 metros. Usando las formulas para la LAN 10BASE-T, y asumiendo una velocidad de propagación de la señal de 0.7 veces la velocidad de la luz en el vacío, se pueden realizar los siguientes cálculos:

P (long del paquete) = 128 x 8 bits = 1024 bits
D (velocidad de transferencia del dato) = 10 Mbps
(fijado por Ethernet)
L (Longitud de la LAN) = 200,5 metros
V (velocidad de propagación de la señal) = 0,7 x la vel. de la luz en el vacío
= 2,1 x 100.000.000 m/seg
Tx = P/d = 1024/10Mbps = 102,4 microsegundos
Td = L/V = 200,5/(2,1 x 100.000.000) = 0,95 microsegundos
a = Td/Tx = 0,95/102,4 = 0,0093

U(10 BASE-T) = 1/(1+a) = 1/(1+0,0093) = 99%

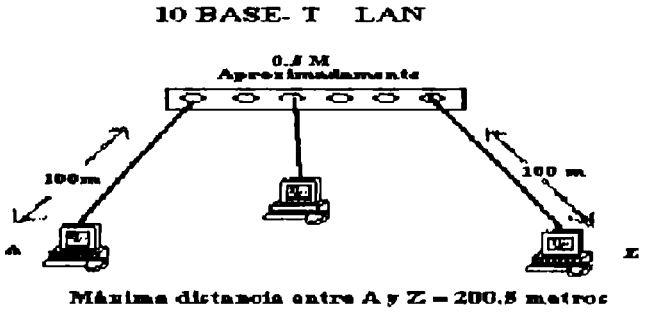


Fig A

En la figura B se muestra la longitud de la LAN, tal como lo establece la norma IEEE 802.3 para 10BASE-5 LAN que tienen una longitud máxima de 2.800 metros. Replanteando los cálculos se obtiene:

P (long del paquete) = 128 x 8 bits = 1024 bits
D (velocidad de transferencia del dato) = 10 Mbps (fijada por Ethernet)
L (Longitud de la LAN) = 2.800 metros
V (velocidad de propagación de la señal) = 0,7 x la vel. de la luz en el vacío
= 0,7 x 3 x 100.000.000 m/seg
= 2,1 x 100.000.000 m/seg

Tx = P/d = 1024/10Mbps = 102,4 microsegundos
Td = L/V = 2,800/(2,1 x 100.000.000) = 13,3 microsegundos
a = Td/Tx = 13,3/102,4 = 0,13

U(10 BASE-5) = 1/(1+a) = 1/(1+0,13) = 88,4%

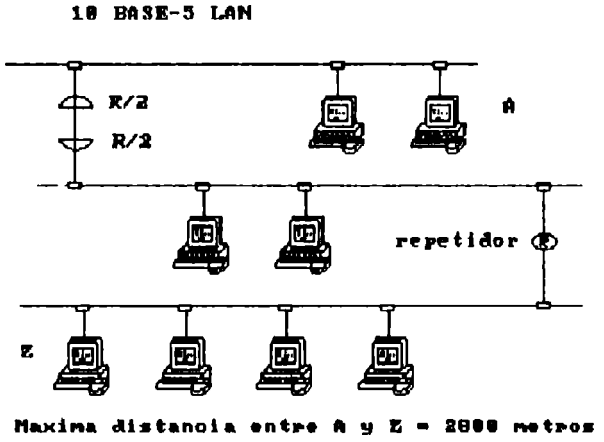


Fig B

Se observa una caída en la performance de alrededor de un 10% entre 10 BASE-T (99%) y 10 BASE-5 (88%). Estos cálculos no incluyen tiempo de procesamiento ni tiempo de espera en cola de los adaptadores de red. Actualmente la utilización de la red disminuye si dichos tiempos son tenidos en cuenta. Una solución para este problema es conectar 10 BASE-T y 10 BASE-5 a través de bridges, para que el mecanismo de CSMA/CD sea dividido en demandas separadas.

No todos los adaptadores para una tecnología de red son creados iguales. Existen muchos fabricantes de adaptadores de Ethernet, por ej; aunque todos ellos cumplen con el standard de Ethenet de 10Mbps, estos tienen diferentes velocidades efectivas de transmisión de datos. Los adaptadores de redes de 3COM son populares, pero hay al menos dos tipos de adaptadores para PCs IBM: 3COM EtherLink y 3COM EtherLink Plus; La segunda es una tarjeta de alta performance y por lo tanto más cara, tiene paquetes de buffers y circuitería más rápida. Idealmente, todas las estaciones sobre una red deberían tener tarjetas de alta performance para obtener la velocidad máxima posible de transferencias de datos. Sino, al menos el servidor y los ruteadores externos deberían tener tarjetas de alta performance.

1.3 Velocidad de la CPU del servidor

Un servidor de archivos rápido ejecuta velozmente el código del sistema operativo NetWare mejorando la performance. La performance de la CPU del servidor es medida en MIPS (millones de instrucciones por segundo).

Cuando se carga NetWare v3.x, usando el programa SERVER.EXE, este ejecuta un test de velocidad del sistema. Los propósitos de este test es informar al administrador de la red la velocidad de operación del servidor . Algunas máquinas basadas en Intel 80386 vienen con selector de velocidades; cuando se inicia el sistema la velocidad puede estar entre 6MHz o 8MHz. La baja velocidad indica que el servidor no opera con su máxima velocidad de reloj y esto puede afectar la performance.

Actualmente, la clasificación de velocidad esta en función de los siguientes factores:

- Tipo de CPU
- Velocidad del reloj de la CPU
- Velocidad de la memoria
- Estado de espera de la memoria
- Velocidad y longitud del cache de la CPU
- Diseño total del sistema.

La siguiente tabla[4] muestra la clasificación de velocidades de diferentes máquinas servidoras.

Computadora	Chip	Velocidad del reloj	Estado de espera	Clasificación
Compaq	386S	80386SX	1	98
Novell	386AE	80386	1	121
Compaq	386/25	80386	0	242

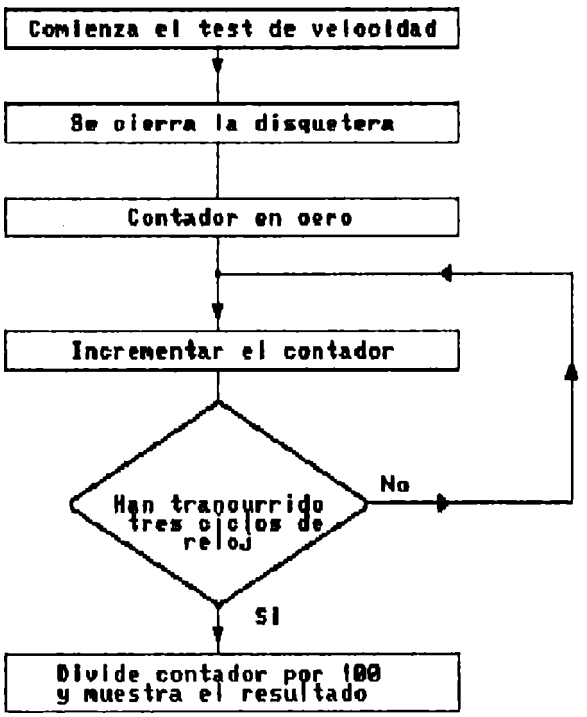
Se puede ver en esta tabla que Compaq 386S con 80386sx de 16MHz, tiene una clasificación de velocidad de 98, donde con un 80386 la clasificación corriendo a 16 MHz tiene una clasificación de velocidad de 121.

Una máquina 80486 puede tener una clasificación de velocidad de alrededor 600. La clasificación de velocidad indica que se deben evitar como máquinas servidoras, las computadoras que requieren estado de espera en el acceso memoria.

[4]Extraída de [6].

El test de velocidad del servidor es un simple loop que corre por aproximadamente 0.16 segundos y cuenta el nro de veces que una parte del código puede ser ejecutada en menos de 2/100 de segundos. Un nro muy grande de iteraciones indica una máquina muy rápida.

La siguiente figura muestra el test de velocidad:



Antes de comenzar el test de velocidad, la diskettera es desconectada (algunas computadoras automáticamente reducen la velocidad cuando se accede el sistema de diskettera, lo cual afecta la clasificación de velocidad). El código ejecutable verifica si 3 ciclos de reloj del timer interno de la Pc (0.16 segundos) han transcurrido. Esta operación implica un nro de instrucciones que mueven datos entre los registros de la CPU y la memoria. El contador es dividido por 1000; el resultado se muestra como un índice de velocidad. Una clasificación de velocidad de 242 indica que el contador fue incrementado 242.000 veces en 3 ciclos de reloj.

2 Aspectos de Software

Los factores de software que determinan el rendimiento de la red están relacionados principalmente con el Sistema Operativo de Red -Netware Operating System-. El rendimiento del software usualmente depende de un buen ajuste del NOS que opera en el servidor.

Aspectos que afectan el rendimiento del servidor :

- Manejador de memoria del servidor .
- Server file caching, directory caching y hashing.
- Optimización de búsquedas en disco.
- Ajuste de los parámetros del servidor .

2.1 Comparaciones de NetWare v 2.11 con NetWare 3.11

2.1.1 Memoria del Servidor

La memoria del servidor está asignada en NetWare v3.x sobre una base dinámica. NetWare v3.x optimiza la asignación de memoria mejorando por completo el rendimiento del sistema.

El diseño de asignación de memoria en NetWare v2.x no es flexible. Muchos de estos problemas ocurren por los segmentos de 64Kb de la arquitectura del microprocesador Intel 80286 sobre el cual NetWare v2.x está basado. NetWare v2.x está escrito en lenguaje assembler. Para mantener un código eficiente, se intenta evitar el uso de segmentación aritmética en el acceso de datos. Esto significa que internamente el segmento de datos finaliza en el limite superior del segmento de memoria de no más de 64Kb. con alguna consecuencia directa en el rendimiento de NetWare v2.x, tales como limitar el número de -File Servise Processes- en el servidor, el cual limita el número de archivos que puedan ser manejados simultáneamente.

La segmentación permanece con NetWare v3.x y el microprocesador Intel 80386, pero el tamaño máximo de segmento permitido por este microprocesador es de 4Gb (Gigabytes). NetWare usa segmentos de 4Gb para código y datos (llamado Modelo de Memoria Plana.)

2.1.2 Server-Caching, Hashing

La memoria remanente después de la asignación de espacio para el núcleo del sistema operativo de NetWare, tablas de datos, NLM -NetWare Laodable Modulo- procesos y aplicaciones es usada por el file caching. La diferencia de velocidad entre la RAM y el disco es del orden de 100 a 1. Con el uso de parte de la RAM como cache se puede mejorar la velocidad entre 10 y 100 veces en el acceso a disco.

En el file-caching, los archivos leídos del disco son mantenidos en la RAM del servidor para futuros accesos hasta que el espacio de memoria ocupado sea necesitado para datos leídos recientemente. Esta técnica puede mejorar el rendimiento del servidor sobre las lecturas del disco.

Las entradas del directorio del disco son mantenidas en la memoria del servidor de archivos, eliminando la necesidad de acceder a los discos para la búsqueda del directorio, esto se llama Directory-Caching.

Se puede usar una función de hashing de directorio para localizar la búsqueda de una entrada de directorio. Este esquema es llamado Directory-Hashing.

Ambos, NetWare v2.x y v3.x implementan el file-caching, directory-caching y directory-hashing, siempre que exista espacio disponible en la RAM.

2.1.3 Optimización de Búsqueda en Disco

NetWare utiliza un optimizador de búsqueda, para minimizar excesivos movimientos del brazo del disco, al satisfacer múltiples requerimientos de E/S de varias estaciones . Si las operaciones de E/S fueran ejecutadas en el mismo orden de arribo, dado que los requerimientos de disco arriban en orden aleatorio, se realizarían excesivos movimientos de la cabeza (llamado trillado de disco -Disk Trashing-). Un forma de solucionar el problema del movimiento del brazo, es ordenar los pedidos de I/O del disco basándose en la ubicación del sector y la pista del disco, relativas a la actual posición de la cabeza del disco, antes que por su orden de arribo; en consecuencia la cabeza del disco realizará menos barridos.

2.2 Optimización de la performance de NetWare v3.x

NetWare v3.x monitorea sus recursos y cambia parámetros automáticamente tratando de mejorar la performance del servidor . Los parámetro especifican los limites máximo y mínimo para los recursos del sistema.

Los parámetros pueden ser controlados o visualizados desde el comando Set. Cuando se tipea el comando SET en el prompt del DOS, NetWare v3.x lista el siguiente menú de categorías y pregunta cual desea ver:

1-Communications.
2-Memory.
3-File Caching.
4-Directory Caching.
5-File System.
6-Locks.
7-transaction tracking.
8-Disk
9-Misellaneous.

Desde el punto de vista de performance, muchos de estos parámetros pueden estar asociados. Los comandos SET para memoria, cache de archivo, y cache de directorio afectan la performance de la memoria del servidor . Los comandos SET para el file system, locks y transaction tracking, por ejemplo, afectan la performance del sistema de archivos. Los comandos SET son agrupados y discutidos de la siguiente manera:

- Optimizando la Administración de Memoria.
- Optimizando el Sistema de Archivos.
- Optimizando las Comunicaciones.
- Parámetros de Umbral y Aviso.

El comando SET para cambiar el valor de los parámetros es:
SET param. descrip. = nuevo valor
Para ver el corriente seteo del parámetro, se usa:
SET param. descrip.

Ciertos parámetros no pueden ser asignados en el prompt de la consola o dentro del archivo AUTOEXEC.NCF. Deben ser asignados dentro del archivo STARTUP.NCF. La siguiente es una lista de estos parámetros:

AUTO REGISTERS MEMORY ABOVE 16 MEGABYTE
AUTO TTS BACKOUT FLAG
CACHE BUFFER SIZE
MAXIMUN PHYSICAL RECEIVER PACKET SIZE
MAXIMUN SUBDIRECTORY TREE DEPTH
MINIMUN PACKET RECEIVE BUFFERS

2.2.1 Optimizando la Memoria para NetWare v3.x

El manejador de memoria para NetWare v3.x es mucho más flexible que el de NetWare v2.x.

Bajo Netware la memoria del servidor se divide en tres áreas de aplicación:

1-File Cache Buffer Pool.
2-Pool de Memoria permanente.
3-Pool de Memoria Alloc.

Los pools de memoria permanente y memoria alloc crecen o disminuyen dependiendo de la demanda. La memoria que no se asigna a estos pools ni al núcleo del NOS es usada por el file caching.

El file caching buffer pool es usado para almacenar bloques de datos de los archivos más frecuentemente usados. Dos pools, llamados movible y no-movible, interactúan con el file cache buffer pool. Ellos obtienen memoria del file cache buffer pool, y cuando terminan retornan dicha memoria. El pool movible es usado para tablas del sistema, las cuales cambian de tamaño dinámicamente, tales como las tablas FAT[5]. Las tablas y objetos en el pool movible pueden ser movidos en la memoria para prevenir excesiva fragmentación. El pool no-movible es usado por NLM -NetWare Loadable Modulo-.

El pool permanente es usado cuando se necesita memoria por un largo periodo, como para los buffers cache de directorio y los buffers de recepción de paquetes. Aunque se llama memoria permanente es bastante dinámica. Los buffers de comunicación son

[5]Tabla de Asignación de archivos

un ejemplo del uso de pool de memoria permanente. Cuando se carga NetWare, este asigna un cierto número de buffer de comunicación. Al incrementarse la actividad de la red, NetWare asigna adicionales buffers de comunicación para almacenar paquetes de datos extra. Estos buffers asignados en memoria permanente, una vez asignados, no serán liberados hasta la reinicio del servidor .

El pool de memoria alloc es usado para almacenar la siguiente información:

- Mapeo de los controladores.
- Buffer de requerimiento de servicios.
- Apertura/bloqueo de archivos y semáforos.
- Tablas NLM.
- Tablas SAP (Service Advertising Protocol).
- Información de conexión de usuarios.
- Tablas para manejar colas.
- Colas de mensajes a transmitir.

La alloc memory es usada temporalmente para cumplir las necesidades de procesos cortos. Durante la ejecución de NetWare, los objetos de datos pueden necesitar ser creados dinámicamente y eliminados después que el NOS finalice su uso. La memoria usada para el pool alloc es manejada por un lista enlazada de bloques de memoria libre. Usando esta lista enlazada, el NOS puede rápidamente hallar la memoria que necesita. Cuando se requiere memoria, esta es asignada de la lista de bloques libres. Cuando se liberada, se retorna a la lista de bloque libres para ser reusada.

NetWare trata de asignar memoria para NLMs del alloc pool. Para satisfacer el requerimiento de memoria para NLM , NetWare pide memoria del file cache buffer pool. Una vez tomada esta memoria no es retornada al file cache pool, aún que los NLMs sean descargados. El peor de los casos se da cuando NLMs son frecuentemente leídos y descargados. Bajo estas circunstancias, permanentemente se pide memoria del área del file cache y podría agotarse, lo cual degrada la performace. NetWare tiene mecanismo de control para prevenir esta perdida de memoria. El comando SET puede ser usado para limitar el tamaño del alloc pool. El alloc pool tiene un tamaño por defecto de 2MB pero se le puede asignar un tamaño entre los 50KB y los 16MB.

El siguiente comando SET limita el tamaño del buffer del alloc pool y puede ser usado para resolver el problema de perdida del file cache buffer pool:

SET MAXIMINIMUN ALLOC SHORT TERM MEMORY=n

Donde n es el tamaño máximo (en byte) para el alloc pool memory. El rango va desde los 50KB a 16MB. Por defecto el valor es de 2 MB.

El file caching también afecta el rendimiento de la memoria del servidor . El file caching permite que bloques de datos de archivos que comienzan a ser escritos o leídos permanezcan en RAM, de este modo se agiliza el acceso a estos archivos. El file caching es sincronizado con el número de buffer de file caching, escritura concurrentes en disco y sincronización del disco (retardo de la escritura del cache).

MAXIMUN ALLOC SHORT TERM MEMORY=n

Determina la cantidad de memoria (*n*) que el servidor asigna para alloc pool. El rango va desde 50KB a 16MB y por defecto de 2 MB. Normalmente no se debería cambiar estos valores, pero puede disminuirse de necesitarse memoria adicional, tal como la lectura de otros NLMs. Determinar si incrementamos estos valores es fácil; si hay insuficiente memoria asignada al pool, el servidor comienza a emitir mensajes de precaución : que la operación no puede ser completada. Incremente el valor en unidades de 1Kb hasta que el mensaje desaparezca.

AUTO REGISTER MEMORY ABOVE 16 MEGABYTE=[ON¦OFF]

Las máquinas EISA pueden soportar más de 16Mb de RAM. Bajo ciertas circunstancias, se puede usar este parámetro para permitir que NetWare maneje memoria adicional correctamente. La asignación de este parámetro se realiza en el STARTUP.NCF y no puede ser asignado en el prompt de la consola o dentro del AUTOEXEC.NFC. NetWare necesita conocer toda la memoria disponible cuando se inicia. El valor por defecto del parámetro es ON. Si se está usando una placa que solo puede direccionar 24 bit de RAM, se puede asignar OFF al parámetro.

CACHE BUFFER SIZE=n

Desde que NetWare v3.x puede soportar bloque de disco de tamaño variable, se puede asignar el tamaño del cache buffer que pueden tener los bloques de disco. Por defecto el tamaño es de 4KB que es un bloque de disco. Si una longitud del tamaño del bloque de disco esta siendo usada, se puede aumentar este valor para mejorar la performance. Este parámetro debe ser asignado en el STARTUP.NCF y no puede ser asignado en el AUTOEXEC.NCF, como tampoco en el prompt de la consola.

Rango de Valores [4KB a 16KB].

MINIMUN FILE CACHE BUFFERS=n

El file caching puede ser utilizado para mejorar la performance de las operaciones de entrada/salida. NetWare usa la memoria para file caching, después de haber asignado memoria para el sistema operativo y NLMs. En NetWare v3.x se puede asignar una cierta cantidad de memoria para el file caching. El file caching actúa también como una reserva de memoria. El servidor reservada esta memoria para sus necesidades; el problema de esto es que la memoria para file cache puede agotarse, lo cual puede afectar seriamente la performance del servidor . Una cierta cantidad de memoria siempre deberá ser reservada para file caching (este parámetro puede ser usado para asignar la cantidad minima reservada para file cache). Si las operaciones de E/S de los archivos parecen lentas, o ocurren largas esperas y time-out para las operaciones de le red, el file cache puede haber caido por debajo del valor mínimo. Si esto sucede, el valor de este parámetro debe ser incrementado. Por defecto el valor del parámetro es 20. El número deberá ser incrementado gradualmente hasta un máximo de 1.000. Un valor alto para este parámetro daría como resultado agotar la memoria y tendría un impacto sobre otras estructuras de datos que necesitan memoria.

2.2.2 Optimizando el Sistema de Archivos para NetWare v3.x.

El caching de archivo permite un rápido acceso de los archivos más frecuentemente usados, manteniendo los bloques de disco en memoria RAM (más rápida para las operaciones de lectura y escritura).

Un proceso de fondo escribe estos buffer de disco (buffer llenos) al disco, no durante los momentos críticos pero dentro de un tiempo especificado por el parámetro Dirty Disk Cache Delay Time .

El file caching afecta la performance de la memoria y del sistema de archivos.

El directory caching también juega un rol importante en la performance del sistema de archivos, permitiendo un rápido acceso al directorio más frecuentemente usado. La tabla de directorio son mantenidas en el directory buffer dentro de la RAM, y el servidor usa un pequeño "algoritmo residente" para mantener las entradas de directorio en memoria. Cuando el servidor es encendido, NetWare comienza con un mínimo valor para el *directory cache buffer*, 20 por defecto. Cuando dicho valor asignado para el directory cache buffer es usado, el servidor debe esperar un tiempo para la asignación de más espacio -*directory cache allocation wait time*-, por defecto 2.2 segundos antes de concretar la asignación de más buffer. El número de entradas de directorios puede incrementarse hasta un máximo dado por *maximun directory cache buffers* de 400 por defecto. Así, cuando los buffers directory cache se incrementan, el nº de buffer disponible para el file caching disminuye. Por lo tanto, hay una relación inversa entre el directory cahing y el file caching; ellos deben ser cuidadosamente balanceados para un óptimo rendimiento.

Los siguiente parámetros SET afectan la performance del sitema de archivos.

MAXIMUN CONCURRENT DISK CACHE WRITES=n

A diferencia del DOS, NetWare puede ejecutar escrituras en disco concurrentemente. Una razón de esto es que NetWare no depende de una simple conexión con el BIOS para ejecutar las operaciones de E/S. El dispositivo de disco para NetWare permite múltiples conexiones de E/S concurrentemente. Para acelerar las

operaciones de E/S del disco, un número de requerimientos pueden ser puestos sobre el mecanismo de optimización. Este mecanismo optimizador permite a la cabeza del disco realizar un barrido continuo a través del disco, siendo más eficiente que un movimiento aleatorio de la cabeza del disco. Este mecanismo puede ser usado para asignar el número de requerimientos de escritura (*n*) que puede ser puestos en la cola del optimizador de disco, para que con un simple barrido a través del disco sea suficiente. Un valor alto para este parámetro hace más eficientes los requerimientos de escritura; un valor bajo hace más eficientes los requerimientos de lectura. Un manera de decidir que valor asignar a este parámetro, es monitorear con el MONITOR el número de cache buffer usados. Si este número dice que los buffer están usados a un 70% ó más, la escritura predomina más que la lectura. En este caso usted puede mejorar el rendimiento incrementando este parámetro.

Rango de Valores [10 a 100]. El valor por defecto 50.

DIRTY DISK CACHE DELAY TIME=n

Para mejorar las E/S del disco, el servidor realiza las escrituras con un retardo de tiempo, primero escribe en RAM y luego al disco cuando este está desocupado. NetWare permite controlar este período de espera. Si hay muchos requerimientos pequeños, se almacenan en buffers dentro de la RAM. Un tiempo de espera en el disk cache demasiado pequeño puede incapacitar el mecanismo de retardo. De esta manera el retardo de escritura puede perjudicar la performance del servidor. El valor por defecto es de 3.3 segundos, es suficiente para una típica cantidad de operaciones de E/S dadas en el servidor. El valor puede ser tan pequeño como 3.1 segundos, pero no más de 10 segundos. Un valor extremadamente alto puede hacer al servidor de archivos más vulnerable a una coalición, por que muchas operaciones de E/S pudieron estar en RAM sin ser depositadas en el disco.

MINIMUN FILE CACHE BUFFER REPORT THRESHOLD=n

NetWare emite un mensaje de precaución si el número de file cache buffer cae por debajo de cierto umbral. El primer mensaje de precaución de que el número de cache buffer es bajo, es emitido cuando la siguiente condición se cumple:

Número corriente de cache buffer	-	Mínimo file cache buffer	<=	Mínimo file cache buffer reportado como umbral

Supongamos que el minimo de file cache buffer es asignado en 20 y el mínimo de file cache buffer reportado como umbral es asignado en 30, cuando el corriente número de cache buffer es 50, la condición previa se cumple y el mensaje de precaución es emitido.

Rango de Valores [0 a 1000].

El valor por defecto 20.

MINIMUM FILE DELETED WAITE TIME=n

Cuando un archivo es deleteado, este no será purgado inmediatamente del volumen. Este parámetro controla la mínima cantidad de tiempo que un archivo deberá permanecer en el volumen antes de ser purgado.

Rango de Valores [0 a 7 días].

El valor por defecto 1 minuto 5,9 seg.

FILE DELETED WAIT TIME=n

Este parámetro determina la cantidad de tiempo que el archivo debe ser retenido en el volumen antes de ser purgado. El servidor mantiene al menos 1/32 de espacio libre en disco par nuevos archivos y comienza purgando archivos (comienza con el archivo más viejo). El rango del parámetro es de 0 segundos a 7 días, por defecto es de 5 minutos 29,6 segundos. Un pequeño valor para este parámetro haria más lento al servidor , ya que muchos archivos serian deleteados. Un valor muy grande podría tener un impacto adverso en la disponibilidad de espacio en disco.

MINIMUN DIRECTORY CACHE BUFFER=n

En NetWare, los directorios son almacenados en RAM para realizar las búsqueda y actualización de archivos más rápidamente. Los directorios son mantenidos dentro del directory cache buffer. El número mínimo es controlado por este parámetro. Asignar un número alto determina realizar más rápidamente las búsquedas en el directorio, pero un número demasiado alto, consecuentemente determina que la porción de memoria no usada no estará disponible para el file caching. si el nro es muy bajo el servidor consume tiempo asignando nuevos buffers para directorio.

El rango de este parámetro va desde 10 a 2.000.

Por defecto es de 20.

MAXIMUN DIRECTORY CACHE BUFFERS=n

Dependiendo del número de entradas de directorio y archivos, el servidor asigna los directory cache buffers. Para prevenir que todo el espacio disponible sea usado por el directory cache, se permite asignar un limite máximo al número de directory cache buffer. Si el servidor continúa siento lento, este parámetro debe incrementarse después de consultar el MONITOR NLM sobre el cache directory usado. Si el MONITOR NLM reporta una falta de RAM disponible y este parámetro es demasiado alto entonces debe ser reducido. El servidor no libera memoria automaticamente, si el valor de este parámetro es bajo, el servidor debe recomenzar para liberar la memoria.

El rango del valor del parámetro va desde los 20 a 4.000.

Por defecto es de 500.

DIRECTORY CACHE BUFFER NONREFERENCED DELAY TIME=n

El espacio del Directory cache buffer es finito. Cuando una nueva entrada de directorio es leída, las anteriores pueden ser sobrescritas.

Este parámetro controla la cantidad de tiempo que una entrada de directorio permanecerá en RAM en un estado no referenciado antes que esta pueda ser sobrescrita por otra entrada de directorio. Un valor alto indica que una entrada de directorio es más probable que esté en RAM cuando se necesite. Un bajo valor puede hacer lento el acceso al directorio.

El rango del parámetro va desde 2 a 5 minutos.

Por defecto es de 5.5 segundos.

DIRTY DIRECTORY CACHE DELAY TIME=n

Este parámetro controla la cantidad de tiempo que un directorio puede permanecer en RAM antes de que sea escrito en disco.

Con un alto valor para este parámetro se realizarán rápidas escrituras del directorio, pero habra más chances de que las tablas de directorio sean tomadas fuera de sincronismo con el disco, produciendo una coalición. Un valor muy bajo para este parámetro disminuye la posibilidad de que las tablas se hallan dañado, pero la frecuentes escrituras reducen la

TURBO FAT RE-USE WAIT=n

TURBO FATs (tablas indexadas) son usadas para indexar archivos (archivos con más de 64 entradas) que toman tiempo en construirse. Si un número de operaciones indexadas están siendo ejecutadas liberar el TURBO FATs cuando estas pueden ser usadas nuevamente no tienen sentido. Este Parámetro puede ser usado para determinar el períodos de tiempo que el TURBO FAT, puede permanecer en memoria después que el archivo indexado es cerrado. Después de un tiempo de espera, el buffer del TURBO FAT puede ser reusado por otro archivo indexado.

El valor del parámetro va de 0.3 segundos hasta 1 hora 5 minutos 54.6 segundos, por defecto toma 5 minutos 29.6 segundos.

AUTO TTS BACKOUT FLAG=[ON¦OFF]

En NetWare v3.x, el TTS *-Transaction Tracking System-* es una característica desarrollada dentro del sistema operativo. Dentro de los sucesos de coalición estos archivos han sido marcados para restaurarlos a un estado previo consistente. Por defecto toma el valor de OFF, el cual quiere decir que el usuario es avisado con el mensaje *"Incomplete tranaction(s) found. Do you wish to back them out?"*.

Este parámetro puede ser asignado solamente en el STARTUP.NCF

TTS ABORT DUMP FLAG=[ON¦OFF]

Este parámetro permite recuperar las escrituras incompletas de una transacción de archivo, producidas por una falla en el servidor de archivos. Cuando este parámetro está seteado en ON, NetWare crea un archivo llamado TT$LOG.ERR en el volumen SYS, para restaurar las escritura de transacciones de archivos. El valor por defecto es OFF, de esta manera la información no se resguarda; produciendo rápidas escrituras y menos reusabilidad.

MAXIMUN TRANSACTION=n

Se puede establecer el número de transacciones que pueden ser ejecutadas simultáneamente en el servidor . Hay que decrementar este valor si pocas transacciones son ejecutadas en el servidor .

Rango de Valores [100 a 10000].

El valor por defecto 10000.

TTS UNWRITTEN CACHE WAIT TIME=n

Se puede determinar la cantidad de tiempo que los datos de transaccionales pueden ser retenidos en memoria.

Rango de Valores 11 segundos 10 minutos 59.1 segundo.

El valor por defecto 1 minuto 59 segundo.

TTS BACKOUT FILE TRANSACTION WAIT TIME=n

El TTS BACKOUT FILE restaura los archivos sujeto a la información resguardada por archivo, este archivo no puede ser restaurado si hay una falla en el servidor . NetWare borra el archivo resguardado si no se usa. NetWare permite que se determine cuanto tiempo un bloque permanece disponible por TTS cuando este archivo no es usado.

Rango de Valores [1 minuto 5.1 segundos hasta 1 día 2 horas 21 minutos 51.3 segundo].

El valor por defecto 59 minutos 19.2 segundo .

ENABLE DISK READ AFTER WRITE VERIFY=[ON¦OFF]

Este parametro permite habilitar y dehabilitar el mecanismo de hot-fix, el cual ejecuta una lectura después de una escritura, y compara el bloque leído con el de memoria.

Se debería dejar este parámetro con el valor por defecto de ON. Si se utiliza disco reflejado o duplicado de manera de asegurar la recuperacion del disco, se puede setear el parámetro en OFF para mejorar la velocidad de las escrituras.

MAXIMUN RECORD LOCKS PER CONNECTION=n

NetWare permite a una estación bloquear registros. Todos los registros bloqueados consumen recursos del servidor . NetWare permite limitar el número de registros bloqueados en una estación de trabajo. El número de registros bloqueados es controlado por el MONITOR NLM.

El parámetro podría ser incrementado si una aplicación falla mientras se bloquean registros. El parámetro podría ser decrementado si una estación de trabajo consume muchos recursos del file servidor. El rango va desde 10 a 10.000 y por defecto es de 500.

MAXIMUN FILE LOCKS PER CONECTION=n

NetWare permite a la estación de trabajo bloquea archivos. Se puede limita el número de archivos bloqueados que una estación de trabajo puede haber, para prevenir que una estación use muchos recursos. El número de bloqueos puede ser monitoreado desde el MONITOR.NLM. El parámetro debería ser incrementado si una aplicación falla mientras está bloqueando registros. El parámetro podría ser decrementado si la estación de trabajo consume muchos recursos. El valor del parámetro va desde 10 a 10.000 y por defecto es de 250.

MAXIMUN RECORD LOCKS=n

Este parámetro global controla el número total (n) de registros bloqueados que pueden ser procesados simultáneamente. El propósito de este parámetro es prevenir tener muchos registros bloqueado consumiendo muchos recursos. Se deberia incrementar este número si una aplicación falla recibiendo el mensajes de insuficientes bloqueos de registros. Este parámetro va en un rango desde 100 a 100.000 y por defecto es de 20.000.

2.2.3 Optimizando las Comunicaciones

Los parámetros de comunicación controlan características de los buffers de comunicación. Los buffers de comunicación son áreas dedicadas en la RAM del servidor para mantener paquetes. Los paquetes permanecen en memoria antes de ser procesados por el File Server Prosses (FSPs).

Describiremos los parámetros SET que pueden afectar la performance en la comunicación del servidor .

MAXIMUN PHYSICAL RECEIVE PACKET SIZE=n

El tamaño del paquete que puede ser transmitido, está determinado por el mecanismo de acceso físico de la red y limitaciones del driver. En el servidor, NetWare le permite definir el máximo tamaño (n) de un paquete que puede ser procesado por el servidor de archivos. Cuando una estación de trabajo realiza una conexión al servidor, el tamaño del paquete es negociado, basandose en el driver de red que esta usando la estación de trabajo.

El valor del parámetro necesita ser grande para adecuarse al tamaño máximo del paquete usado por la estación de trabajo. El rango de valor de parámetro va desde 618 a 4.202 byte, por defecto es de 1.130 byte.

Generalmente un tamaño grande del paquete puede acelerar la comunicación, pero consume más RAM. Este parámetro puede ser solamente seteado en el STARTUP.NCF.

MAXIMUN PAQUET RECIVE BUFFERS=n

El servidor necesita mantener un cierto número de buffers de recepción de paquetes en RAM para evitar ser invadidos por datos. Normalmente el servidor asigna buffers de recepcion dinámicamente, basandose en las necesidades. Netware permite definir el limite superior (n) de buffer de recepción de paquetes que el sistema operativo puede asignar. El NOMITOR NLM puede ser usado para monitorear el uso de este parámetro, si está cerca del máximo valor, se debe incrementar el valor hasta que pueda recibir al menos un paquete por estación de trabajo. Para OS/2 y MS Window incrementa este valor, basandose sobre el número de las aplicaciones de red que están corriendo simultáneamente en las estaciones de trabajo.

Permite un minimo de un buffer por aplicación. El MONITOR NLM puede usarse para monitorear los mensajes de error *NO ECB available count*. Si estos errores son reportados, se debe incrementa este parámetro de a 10. Para el servidor que utiliza plaquetas con bus EISA y Microchanel, incrementar este parámetro permite de 5 a 10 buffers de recepcion de paquetes en la plaqueta de la red. Si el número de proceso de servicio de archivo reportados por el MINITOR NLM es cercano a su máximo, se puede incrementar dicho valor para reducir la necesidad de más buffers de recepcion de paquetes.

Rango de Valores [50 a 2.000].

El valor por defecto 100.

MINIMUN PACKET RECEIVE BUFFERS=n

NetWare le permite definir el número minimo (n) de buffers de recepción de paquetes en el servidor. Se puede usar MONITOR NLM para monitorear el valor actual de buffers de recepcion de paquetes. El valor por defecto de este parametro es 10.

Si el error *NO ECB available count* es reportado desde el MONITOR después que el servidor es iniciado, entonces se debe incrementar este parámetro.

Para los servidores con bus EISA y Microchanel incrementar este parámetro permite por lo menos recibir 5 paquetes en el buffer.

El rango del parámetro va desde 10 a 1.000.

NEW PACKET RECEIVER BUFER WAIT TIME =n

NetWare permite asignar un período de espera para satisfacer un nuevo requerimiento de buffers de recepción de paquetes.

Novell recomienda que este parámetro no sea cambiado para EISA. Rango de Valores [0.1 a 20 seg]. Por defecto 0.1 seg.

DELAY BETWEEN WATCHDOG PACKETS=n

Este es el intervalo de tiempo (n) que existe entre paquetes de control. Después que el servidor envia el primer paquete de control, espara un tiempo antes de enviar un nuevo paquete de control a las estaciones de trabajo, al no recibir respuesta por el primer paquete de control.

El paquete de control es enviado para ver si la estación está todavía viva en la red. Si el parámetro es seteado con un valor muy bajo podría generar un exceso de tráfico, por que los paquetes de control serían enviados a todas las estaciones asignadas al servidor. El rango del parámetro va desde 1 segundo a 10 minutos 26,2 segundos, por defecto es de 4 minutos con 56,6 segundos.

Normalmente el valor por defecto es el adecuado para la mayoría de las redes. El tiempo, de espera entre paquetes de control, debe ser seteado más grande que el tiempo que tarda un paquete en viajar de ida y vuelta a la estación.

DELAY BEFORE FIRST WATCHDOG PACKET=n

El servidor envía paquetes de control a una estación que esté inactiva por algún tiempo. NetWare le permite asignar que tiempo se debe esperar antes de testear una estación inactiva, entonces el paquete de control es enviado para ver que la estación este todavía viva.

Rango de Valores [15,7 seg. a 20 minutos 52,3 segundos].

NUMBER OF WATCHDOG PACKETS=n

Número (n) chances antes de que una estación de trabajo sea dada por muerta, eliminando la conexión. El servidor de NetWare repite paquetes de control si una estación no responde al primer

paquete de control. En una red excesivamente congestionada, la estación responde al paquete de control y el servidor no la recibe.

Rango de valores [5 a 100].

El valor por defecto es 10.

Asignando un bajo tiempo de espera entre paquetes de control con un número grandes de paquetes puede causar un excesivo tráfico y afectar el rendimiento de la red.

2.2.4 Parámetros de umbral y Advertencia

En NetWare v3.x exiten varios parámetros que generan mensajes de advertencia cuando se atravizan ciertos umbrales. Estos parámetros no afectan directamente la performance del sistema, pero son importantes los mensajes de estado de alerta y operación del sistema. Los comandos SET de umbral y parámetros de advertencia son:

```
CONSOLE DISPLAY WATCHDOG LOGOUT=[ON¦OFF]

INMEDIATE PURGE OF DELETED FILE=[ON¦OFF]

VOLUMEN LOW WARN ALL USES=[ON¦OFF]

VOLUMEN LOW WARNING THRESHOLD=n

VOLUMEN LOW WARNING RESET THERSHOLD=n

MAXIMUN PERCENT OF VOLUME USED BY DIRECTORY=n

MAXIMUN PERCENT OF VOLUME SPACE ALLOWED FOR EXTENDED ATTRIBUTES=n

MAXIMUN EXTENDED ATTRIBUTES PER FILE OR PATH=n

MAXIMUN SUBDIRECTORY TREE DEPTH=n

ALLOW UNENCRYTED PASSWORD=[ON¦OFF]

DISPLAY SPURIUS INTERRUPT ALERT=[ON¦OFF]

DISPLAY SPURIOUS INTERRUPT ALERTS=[ON¦OFF]

DISPLAY LOST INTERRUPT ALERTS=[ON¦OFF]
```

```
DISPLAY DISK DEVICE ALERTS=[ON¦OFF]

DISPLAY RELINQUISH CONTROL ALERTS=[ON¦OFF]

DISPLAY OLD API NAMES=[ON¦OFF]

SET MAXIMINIMUN ALLOC SHORT TERM MEMORY = n
```

3 RESUMEN DE LOS FACTORES QUE AFECTAN LA PERFORMANCE DE LA RED

La siguiente tabla (estractada de [5]) es un resumen de las actividades del servidor que deberian considerarse para realizar un buen monitoreo de la red.

Tipo de actividad	Area afectada	Explicación
Razón de acierto del cache	Disco	Cuantas veces se encuentra en memoria un bloque leído o escrito al disco.
Fallo del cache	Disco	Cuantas veces no esta en memoria un bloque leido o escrito al disco
Errores de Lectura y escritura del cache	Disco	Cuantos errores de disco se reportan después que el cache pide al manejador que realice una lectura o escritura
Desborde del cache	Disco	No se pudo asignar un bloque del cache porque todos están siendo usados actualmente
Utilización del cache	Disco	Porcentaje de bloques de información del disco que están en la memoria del cache
Fallo en el canal n del disco	Disco	Los servidores de archivos de NetWare pueden tener hasta 5 canales. Los errores pueden ocurrir durante el arranque o cuando el SFT Level II detecta un fallo en un disco reflejado.
Dispositivo del disco n desactivado	Disco	El dispositivo es tipicamente el controlador. El mensaje de error puedde aparecer cuando el servidor está operando o cuando el SFT Level II detecta un fallo del disco-canal reflejado.
E/S de disco pendientes	Disco	Número de accesos al disco (peticiones de lectura o escritura)
Espacio en disco por volumen	Disco	El espacio total en disco y el área disponible de reparación activa antes de cargar datos

Errores de FAT	Disco	Errores durante una escritura a la tabla de asignación de archivos (FAT)
Fecha y Hora de arranque del servidor / tiempo de encendido	Servidor de Archivos	Muestra la ultima vez que se encendido el servidor y el tiempo que ha estado encendido.
Utilización del CPU del servidor	Servidor de archivos	Debe mostrar la utilización actual, promedio y máxima del CPU.
Paquetes de entrada o salidos perdidos.	Tarjeta de redes	No se puede recibir un paquete porque el servidor carece de buffers de comunicación
Paquetes de entrada o salida desechados	Tarjeta de redes	El paquete es desechado si ha cruzado 16 puentes distribuidores o si su destino es desconocido.
Prueba de IPX de nodo a nodo	Tarjetas de redes	Prueba si las comunicaciones IPX(Internet Packet Exchange) trabaja bien entre estaciones.
E/S de LAN	Tarjetas de redes	La información incluye los paquetes recibidos, transmitidos y distribuidos, retransmisiones de NetBIOS, etc.
Cantidad máxima /pico/actual:		
Objetos del Bindery	Sistema operativo	La información incluye los IDs de usuario, contraseñas, asignaciones del administrador, IDs de grupo, y nombres de servidores.Ayuda a registrar los limites de los objetos.
Conexiones	Sistema operativo	Los usuarios y servicios(servidores de impresión de faxes)conectados. Ayuda a registrar la utilización de usuario a servidor.
Memoria dinámica	Sistema operativo	La cantidad de memoria RAM usada por los componentes del sistema operativo. Ayuda a registrar la utilización de memoria del servidor cuando añade RAM.
Archivos abiertos	Sistema operativo	Demasiados archivos abiertos pueden degradar el rendimiento del servidor.

Archivos con índice	Sistema operativo	Util para archivos de 1MB o superior. Lee la información de la posición del archivo del disco a la memoria.
Buffers de distribución	Sistema operativo	Permite el almacenamiento de paquetes de entrada o salida.
Transacciones	Sistema operativo	Número simultaneo de transacciones. El limite se fija durante la instalación inicial.
Impresora desactivada	Servidor de archivo e impresora	La impresora o esta desactivada o responde incorrectamente.
Servidor no responde	Servidor de archivos	La computadora cliente no se puede comunicar con el servidor.
Estado del registro de transacciones.	Servidor de archivos	Función de base de datos NetWare que copia los registros cambiados a un archivo de resguardo y registra el esquema de recuperación y resguardo.El estado puede ser TTS activado o desactivado.
Fallo en peticiones de retroceso en el TTs.	Servidor de archivos	El procedimiento de retroceso del registro de transacciones de Netware automáticamente devuelve la ultima entrada del servidor.

Los programas de monitoreo detectan problemas en el servidor de archivos y le notifican cuando estos ocurren. Por ejemplo si la capacidad del disco sobrepasa un nivel predefinido, se lo indican mediante un localizador (beeper) o con un mensaje de correo electrónico. También reportan las estadisticas del servidor por un periodo de tiempo, para que puede detectar tendencias.

Los servicios de administración SYSCOM y FCONSOLE que vienen incluidos en Novell reportan gran parte de la misma información que los productos de monitoreo, pero no se puede fijar alarmas con SYSCOM ni FCONSOLE, ni puede ver el rendimiento a través del tiempo.

IV NETWARE.

SEGURIDAD Y SERVICIOS

Contenido Páginas

NETWARE.

SEGURIDAD Y SERVICIOS

1 SEGURIDAD EN NETWARE

Debido a que la seguridad se vuelve crucial en un ambiente multiusuario, NetWare permite a los supervisores de red controlar quien puede acceder a determinados directorios y archivos.

la seguridad del servidor del archivos está manejada en cuatro niveles:

1- Clave de Acceso a la Red.
2- Trustee Seguridad.
3- Seguridad a Nivel de Directorios.
4- Seguridad de Atributos de Archivos.

Estos métodos pueden ser usados separadamente o en forma combinada.

1.1 Clave de Acceso a la Red.

Se aplica a todos los usuarios. Para conectarse (Log-In) al file server, se debe conocer el "nombre del usuario" y su correspondiente clave de seguridad (password). El nombre del usuario representa un identificador de usuario que el server reconoce; las claves de seguridad son opcionales.

Cuando se incorpora un usuario, se le asigna un nombre de usuario, el cual será usado para su posterior conexión. Si se ingresa el nombre de usuario en forma incorrecta, o se ingresa un nombre de usuario que no existe en el server, será denegado el acceso al server; si se tiene una password, el acceso será denegado hasta que se ingrese la password correcta. Esto prevé que usuarios no autorizados ingresen al sistema.

El supervisor de la red puede configurar distintas opciones restringiendo las conexiones de los usuarios.

Restricciones de Conexión	Explicación
Restricciones de tiempo	El supervisor puede limitar las horas durante las cuales el usuario puede estar conectado en la red.
Restricciones de estación o puesto de trabajo.	El supervisor puede asignarle a los usuarios las estaciones de trabajo donde este pueda conectarse.
Conexiones concurrentes	El supervisor puede limitar el número de conexiones simultáneamente que un usuario pueda realizar.
Inhabilitar el Account	La primera vez que el usuario se conecta al file server, un account es creado para ese usuario. El supervisor puede inhabilitar el account del usuario de tal forma que no pueda conectarse.
Expulsión de Intrusos	El supervisor puede activar la opción de Detección/Expulsión de Intruso, la cual ayuda a prevenir conexiones en el sistema de usuarios no autorizados. Si un intruso trata de conectarse varias veces con una password incorrecta, será "expulsado".

Las clave de acceso puede o no ser usada en un file server. Si son usadas, se la debe entrar después de la identificación del usuario. Al ingresar la clave de acceso, esta no es visualizada en pantalla y si se ingresa en forma incorrecta, entonces el acceso a la red será denegado.

El supervisor puede establecer como norma de seguridad la longevidad de una clave de acceso, forzando a los usuarios cambiarlas cada un período de tiempo determinado.

1.2 Trustee Seguridad

Estos derechos son usados para un control individual de la

habilitación de cada usuario para trabajar con los archivos en un directorio dado.

Que es un Trustee?

Es un usuario al cual se le ha asignado privilegios para trabajar en un directorio, esos derechos se extiende hacia abajo a través de todos los sucesivos subdirectorios, hasta que ellos sean redefinidos en algún nivel inferior.

Por ejemplo:

1. No tiene derechos sobre el directorio.
   [ ]
2. El supervisor asigna derechos.
   [RWOCP]
3. El usuario es ahora un Trustee
   [Read, Write, Open, Create, Parental]

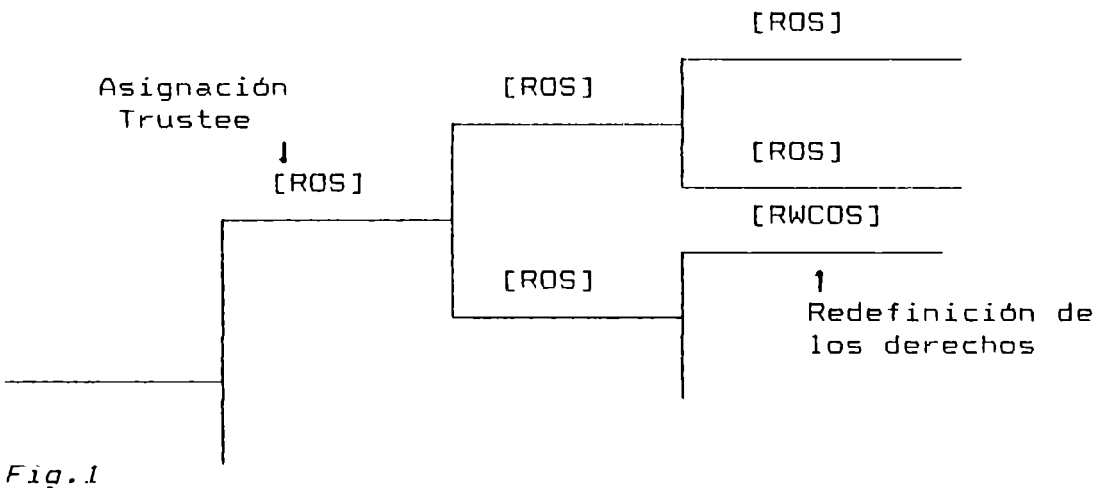


Fig.1

Los Derechos del Usuario pueden ser concedidos directamente a los usuarios o a grupos de usuarios, o pueden ser concedidos indirectamente, a través de equivalencia de seguridad. Cuando los derechos son concedidos a un grupo, estos derechos son automáticamente asignados cada miembro del grupo.

Los derechos son:

- **R** Lectura de archivos abiertos
- **W** Grabar sobre archivos abiertos
- **O** Apertura de archivos
- **C** Crear nuevos archivos
- **D** Borrar archivos
- **P** Parental, que incluye:

Crear, renombrar, borrar subdirectorios del directorio
Poner trustee y los derechos del directorio sobre el directorio.
Poner trustee y los derechos del directorio en sus subdirectorios.

S Explorar el directorio.

M Modificar los atributos de los archivos.

Los derechos de Trustee pueden ser concedidos indirectamente vía la equivalencia de seguridad. La equivalencia de seguridad se refiere a que un usuario ejerce derechos equivalentes a otros usuarios. La misma puede ser asignada por grupos.

1.3 Seguridad a Nivel de Directorio.

La seguridad de directorio es usada para controlar los derechos de todos los usuarios (con excepción del Supervisor) sobre un directorio dado.

Cada directorio tiene una "máscara de máximos derechos". Cuando un directorio es creado, la máscara de derechos contiene algunos de los ocho derechos que se aplican para seguridad de trustee.

Para efectivizar la seguridad de un directorio, el supervisor de la red (o algún usuario con derechos de Parental) borra, el correspondiente derecho, de la máscara de máximos derechos del directorio dado. Esto impide a los trustees de ejercer algunos (o todos) de los derechos que no han sido concedidos para él en ese directorio.

Las restricciones de directorio tienen precedencia sobre las asignaciones de trustee.

Las restricciones de seguridad se aplican *solo a un directorio específico*. La seguridad de directorio *no* se extiende a los correspondientes subdirectorios. Ejemplo:

[RWOCDPSM]	1. Los nuevos directorios tienen todos los derechos.
P [RWOCD SM]	2. Derecho de directorio es borrado.
[RWOCD_SM]	3. El directorio tiene limitado los derechos.

1.4 Seguridad de Atributos de Archivos

Regula si un archivo podrán ser modificado o compartido. Este tipo de seguridad se utiliza principalmente para prevenir cambios o borrado accidental de archivos. Es particularmente usado para proteger la información de los archivos.

Aunque varios atributos de archivos afectan a los usuarios de la red, solo cuatro son aplicados para seguridad: *Read-Write/Read-Only* y *Shareable/Non-shareable*.

- » Si el archivo es marcado Read-Write, los usuarios con derechos apropiados de trustee pueden leer de él, escribir en él, renombrarlo, o borrarlo
- » Si el archivo es marcado Read-Only, los usuarios solo podrán leer de él.
- » Los archivos marcados Shareable pueden ser leídos por más de un usuario a la vez. Este atributo es usado con archivos de datos y archivos marcados Read-Only.
- » El atributo Non-shareable permite que solo un usuario a la vez pueda acceder al archivo.

Los atributos de seguridad por defecto de un archivo creado es Non-shareable/Read-Write.

Derechos Efectivos.

Los derechos efectivos son los derechos que un usuario puede ejercer en un directorio dado. Estos son determinados por una combinación de los derechos de usuario y seguridad de directorio.

Para que un usuario tenga un particular derecho efectivo sobre un directorio, deben existir dos condiciones:

- » El usuario debe tener: su derecho de acceso o vía equivalencia de seguridad, y
- » La máscara de máximos derechos del directorio debe contener ese derecho.

Recordemos que, los derechos de Trustee sobre un directorio se extienden a los correspondientes subdirectorios. Sin embargo, los derechos de directorio deben ser establecidos en todos los subdirectorios; no se extienden a los subdirectorios.

Los atributos de archivos tienen precedencia sobre un derecho efectivo del usuario.

	R	W	O	C	D	P	S	M
Derechos del Usuario	*		*	*			*	
Derechos de Directorio	*	*	*		*	*	*	*
Derechos Efectivos	*		*				*	

Fig.2

2 UTILITARIO SYSCOM

Permite ver y modificar la información de usuarios y grupos del server e información de seguridad y estructura de directorio.

2.1 Accounting

Permite definir el costo de los recursos que el server ofrece para realizar balances de la utilización de los mismos por los usuarios de la red. La función de accounting no es automáticamente instalada.

Opciones de Accounting

Se pueden cambiar las opciones en función a lo más adecuado para su ambiente de red. Estas opciones son:

- Cada vez que un usuario efectúa conexiones y desconexiones al file server, quedan registradas estas operaciones. Esta opción queda automáticamente seleccionada al instalar el accounting en el file server.
- Cargar para cada usuario los recursos consumidos, ya sea del file server, print servers, job servers o gateways.
- Registrar para cada usuario la cantidad de espacio en disco consumido. Al usar esta opción, necesitará especificar la cantidad que debería ser cargada por almacenamiento en disco. Se deberá especificar con que frecuencia y cada cuanto tiempo el file server debería medir el espacio en disco usado y cargar esos informes.
- Registrar para cada usuario el tiempo de uso del file server, así como la cantidad de trabajos requeridos al file server.

La cantidad de tiempo y trabajo que el usuario consume es medido por el file server de 4 formas:

- La cantidad de tiempo que el usuario está conectado al file server
- La cantidad de datos (programas y/o información) que el usuario requiere al file server para leer en el disco
- La cantidad de datos (programas y/o información) que el usuario requiere al file server para grabarlos en el disco
- El número de requerimientos que el usuario hace al file server

2.2 Establecer el Valores de la Carga.

El file server puede efectuar cargas por 5 tipos de servicios:

» Lectura de bloques:

La opción de lectura de bloques determina el valor cargado por la cantidad de datos leídos del server. Se puede especificar la cantidad cargada cada intervalos de 1/2 hora. La carga es asignada por block leído. Un block es igual a 4096 bytes, o 4KB.

» Escritura de bloques:

Esta opción determina el valor cargado por la cantidad de datos escritos en el server. Se puede especificar la cantidad cargada por intervalos de 1/2 hora. La carga es asignada por block escrito. Un block es igual a 4096 bytes, o 4KB.

» Tiempo de conexión:

Esta opción determina el valor cargado por la cantidad de tiempo que el usuario está conectado en el file server. Se puede especificar la cantidad cargada por intervalos de 1/2 hora. La carga es asignada por minuto.

» Almacenamiento en disco:

Esta opción determina el valor cargado por cada block (4096 bytes o 4KB) que es almacenado en el disco por un día. Se pueden asignar diferentes cargas por intervalos de 1/2 hora. La carga es asignada por block-day (número de bloques almacenados en un día).

» Requerimientos al server:

Esta opción determina el valor cargado por requerimientos

al server. Se puede especificar la cantidad cargada por intervalos de 1/2 hora. La carga es asignada por requerimiento recibido.

2.3 Determinar los Valores Cargados.

Si se planea cobrar a los usuarios por los servicios provistos por el file server, se necesitará calcular la cantidad a cargar a los usuarios.

Antes de establecer el valor a cargar por el server , se debe determinar lo siguiente:

1- Determinar cual es el costo y la cantidad que se quiere cargar sobre un período de tiempo determinado.
2- Determinar que servicios se quiere cargar y la cantidad que se quiere ganar sobre ese servicio.
3- Estimar cuanto servicios son usados. Requerirá monitorear el file server por un período de tiempo.
4- Al finalizar el período de monitoréo, usar el utilitario ATOTAL para ver los totales usados por cada servicio.
5- Después que se haya determinado la cantidad que se quiere cargar por cada servicio y se tiene una buena estimación de la utilización del servicio, se puede calcular el valor de carga.

   Los valores de carga son especificados como consiente (multiplicador y divisor). La unidad de carga es completamente arbitraria.

6- Usar la siguiente fórmula para calcular el valor cargado:

$$\frac{\text{(Total que se quiere cargar por el servicio)}}{\text{(Total estimado usado por el service)}} = \frac{\text{(Valor de carga multiplicador)}}{\text{(Valor de carga divisor)}}$$

2 SERVICIOS DEL SUPERVISOR. COMANDOS DE LÍNEA

Para utilizar los servicios de supervisor, se debe estar en el directorio SYS:SYSTEM. Se puede controlar el uso de estos servicios restringiendo el acceso al directorio.

Comando: ***ATOTAL***

Propósito:

Se usa para visualizar el consumo total de los recursos de la red. El ATOTAL recopila información de los registros del sistema de accounting y lista lo siguiente:

- *Total de bloques leídos.*
- *Total de bloques escritos.*
- *Total de tiempo de conexión (en minutos).*
- *Total de requerimientos de servicios.*
- *Total de almacenamiento en disco (bloques por día).*

NOTA: se entiende por bloque una cantidad fija de Kb.

Los totales serán listados siempre que el accounting esté instalado.

Uso:

1) En el directorio SYS:SYSTEM, tipée
   ATOTAL <Enter>
2) Se puede redireccionar la salida de ATOTAL a un archivo.
   ATOTAL > filename <Enter>

Comando: ***PAUDIT***

Propósito:

Se usa para visualizar los registros del sistema de accounting. El archivo NET$ACCT.DAT, contiene registros cronológicos de todas las funciones de accounting.

Uso:

1) En el directorio SYS:SYSTEM, tipée
   PAUDIT <Enter>
2) Se puede redireccionar la salida de PAUDIT a un archivo.
   PAUDIT > filename <Enter>

Se puede borrar o copiar el archivo NET$ACCT.DAT después de haber redireccionado la información a otro archivo. El sistema generará un nuevo archivo NET$ACCT.DAT, cuya información es almacenada en forma binaria.

V SISTEMA PARA EL CONTROL Y VISUALIZACION DE LOS RECURSOS DE LA RED

Contenido Página

SISTEMA PARA EL CONTROL Y VISUALIZACION DE LOS RECURSOS DE LE RED

El incremento del número de redes de computadoras como así también el incremento en la extención de las mismas en las organizaciones, conduce a la necesidad utilizar herramientas que permitan al administrador controlar los recursos de la red.

Como se vio en el capítulo IV, el sistema operativo NetWare es suficiente para administrar la seguridad de la red y cuenta con servicios para registrar información sobre los recursos utilizados por cada usuario. Pero estos servicios no filtran, cuantifican, ni toman desiciones con la información.

Dada la importancia de detectar tendencias sobre el uso de los recursos de la red, es que se creyo necesario el desarrollo de una herramienta que realice las tareas de procesamiento de dicha información.

1 Análisis

1.1 Objetivos

- » Determinar en forma general y detallada la distribución de recursos consumidos de la red.
- » Facturación automática.
- » Obtener información inherente a la seguridad en la red como: nro. de veces que un usuario es detectado intruso, relación entre nodos físicos y usuarios lógicos, etc.

Nota: Se entiende por "Recursos": carga, tiempo de conexión, requerimientos, bloques leídos, bloques escritos y en disco.

1.2 Restricciones

- » Este herramienta está diseñado para un sistema operativo NetWare V2.11 y posteriores.
- » El ACCOUNTING previamente instalado.

1.3 Características del Usuario

Hay tres tipos de usuarios:

- » Supervisor de la red, con privilegios ilimitados;
- » Usuarios regulares, con privilegios que dependen de las necesidades de trabajo;
- » Operador de la red, con privilegios especiales.

1.4 Resumen de la Información Relevada

El siguiente es un resumen de la información con la actividad de la red, suministrada por Novell. Esta fue relevada en dos instalaciones y bajo características distintas.

Lugar: ALBANO COZZUOL S.A.

Observaciones:

El accounting fue instalado para que registre: los recursos consumidos por los usuarios, detección de intrusos y testéo cada media hora de bloques en disco consumidos por cada USUARIO. Observándose:

- » Conexión y Desconexión del USUARIO; con FECHA, HORA, NOMBRE del USUARIO y DIRECCION DE LA TERMINAL correspondiente.
- » El formato de la FECHA es MM/DD/AA.
- » El formato de la HORA es HH:MM:SS.
- » La dirección de la terminal esta indicada en hexadecimal.
- » Que la Información suministrada por NOVELL no indica el nº de terminal, sino, la dirección de la conexión o desconexión.
- » Al desconectarse el usuario, NOVELL registra: La CARGA, TIEMPO DE CONEXION, REQUERIMIENTOS, BYTE LEIDOS y BYTE ESCRITOS.
- » El TIEMPO DE CONEXION se registra en minutos.
- » Los BYTES LEIDOS Y ESCRITOS indicados en hexadecimal.
- » Al detectar un intruso NOVELL registra FECHA, HORA y NOMBRE del USUARIO.
- » Cada media hora se registra, CARGA, NOMBRE del USUARIO y cantidad de BLOQUES EN DISCO.

7/2/92 18:26:30 File Server COZZUOL
NOTE: about User FER during File Server services.
Login from address 00000001:444649091C7.

```
7/2/92 18:27:44  File Server COZZUOL
   CHARGE: 1268 to User FER for File Server services.
   Connected 1 min.; 312 requests; 000000004625h bytes read;
000000000000h bytes written.
7/2/92 18:27:44  File Server COZZUOL
   NOTE: about User FER during File Server services.
   Logout from address 00000001:4446490091C7.
7/2/92 18:28:15  File Server COZZUOL
   NOTE: about User FER during File Server services.
   Account intruder lockout caused by address
00000001:4446490091C7.
7/2/92 18:30:21  File Server COZZUOL
   CHARGE: 1133 to User SUPERVISOR for File Server services.
   13603 disk blocks stored for 1 half-hour periods.
7/2/92 18:30:22  File Server COZZUOL
   CHARGE: 2794 to User HORACIO for File Server services.
   33537 disk blocks stored for 1 half-hour periods.
7/2/92 18:30:23  File Server COZZUOL
   CHARGE: 384 to User ADRIANA for File Server services.
   4608 disk blocks stored for 1 half-hour periods.
7/2/92 18:30:23  File Server COZZUOL
   CHARGE: 448 to User JCC for File Server services.
   5376 disk blocks stored for 1 half-hour periods.
7/2/92 18:30:24  File Server COZZUOL
   CHARGE: 405 to User ZARINI for File Server services.
   4864 disk blocks stored for 1 half-hour periods.
7/2/92 18:30:25  File Server COZZUOL
   CHARGE: 3520 to User ESTEBAN for File Server services.
   42244 disk blocks stored for 1 half-hour periods.
7/2/92 18:30:26  File Server COZZUOL
   CHARGE: 42 to User OSVALDO for File Server services.
   512 disk blocks stored for 1 half-hour periods.
```

Lugar: L.I.D.I.

Observaciones:

El accounting fue instalado para que no registre los recursos consumidos por los usuarios. Observándose:

- » Conexión y Desconexión de la RED; con FECHA, HORA y NOTA correspondiente.
- » Conexión y Desconexión del USUARIO; con FECHA, HORA, NOMBRE del USUARIO y DIRECCION DE LA TERMINAL correspondiente.
- » Conexiones simultáneas en más de una terminal.

```
9/29/92 15:07:25  File Server GRANDE_SERVER
   NOTE: about User SUPERVISOR during File Server services.
   Login from address 00000001:000000000002.
9/29/92 15:07:33  File Server GRANDE_SERVER
   NOTE: about User SUPERVISOR during File Server services.
```

```
   Logout from address 00000001:000000000002.
9/29/92 15:11:45  File Server GRANDE_SERVER
   NOTE: about User CLAUDIA during File Server services.
   Login from address 00000001:000000000020.
9/29/92 15:12:34  File Server GRANDE_SERVER
   NOTE: about User HUGO during File Server services.
   Login from address 00000001:000000000002.
9/29/92 15:13:37  File Server GRANDE_SERVER
   NOTE: about User CLAUDIA during File Server services.
   Login from address 00000001:000000000007.
9/29/92 15:15:22  File Server GRANDE_SERVER
   NOTE: about User SILVIA during File Server services.
   Logout from address 12345678:000000000002.
9/29/92 15:15:23  File Server GRANDE_SERVER
   NOTE: about User CLAUDIA during File Server services.
   Logout from address 00000001:000000000007.
9/29/92 15:15:32  File Server GRANDE_SERVER
   NOTE: about User CLAUDIA during File Server services.
   Logout from address 00000001:000000000020.
9/29/92 15:17:02  File Server GRANDE_SERVER
   NOTE: about User HUGO during File Server services.
   Logout from address 00000001:000000000002.
9/29/92 15:17:14  File Server GRANDE_SERVER
   NOTE: about User SUPERVISOR during File Server services.
   Server downed.
9/29/92 15:18:07  File Server GRANDE_SERVER
   NOTE: about User SUPERVISOR during File Server services.
   Server booted.
9/29/92 15:18:21  File Server GRANDE_SERVER
   NOTE: about User SILVIA during File Server services.
   Login from address 12345678:000000000002.
9/29/92 15:18:44  File Server GRANDE_SERVER
   NOTE: about User HUGO during File Server services.
   Login from address 00000001:000000000002.
9/29/92 15:18:49  File Server GRANDE_SERVER
   NOTE: about User CLAUDIA during File Server services.
   Login from address 00000001:000000000007.
9/29/92 15:28:43  File Server GRANDE_SERVER
   NOTE: about User HUGO during File Server services.
   Login from address 00000001:000000000002.
```

1.5 Definición del Problema

Novell almacena la información en el ACCOUNTING de distintos sucesos que hacen a la seguridad. El contenido de este, depende de la configuración realizada por el supervisor:

» Fecha y hora de conexión (Log In) y desconexión (Log Out) para cada usuario en el file server (esta opción es seleccionada automáticamente).

» Carga de los recursos consumidos por cada usuario al file server o a otro tipos de servers.
» Cargar al usuario la cantidad de espacio de disco usado del file server. El supervisor especifica con que frecuencia y en que tiempo el file server testéa el espacio en disco consumido.
» Carga al usuario la cantidad de tiempo y trabajo consumido de 4 maneras diferentes:
» La cantidad de tiempo que el usuario se mantiene conectado.
» La cantidad de datos que el usuario lee del file server.
» La cantidad de datos que el usuario escribe en el file server.
» El número de requerimientos que el usuario realiza al file server.

El archivo NET$ACCT.DAT contiene los registros cronológicos de todas las acciones ya descritas.

Para poder visualizar dicha información es necesario usar los comandos de línea ATOTAL y PAUDIT; este último también registra intrusos que tratan de conectarse al file server.

Esta información se puede clasificar de dos maneras:

1) ***Información Global***: resumen diario del uso total del sistema y un total semanal provisto por el comando de línea ATOTAL. Contiene:

- Fecha;
- Tiempo de conexión;
- Requerimientos del server;
- Bloques leídos;
- Bloques escritos;
- Bloques almacenados en disco por día.

2) ***Información Detallada***: discriminada por usuario, provista por el comando de línea PAUDIT, contiene:

- Fecha;
- Hora;
- Nombre del Server;

Alguna de las siguientes posibilidades:

- Carga: Se produce a intervalos regulares. Contiene:
  - Un nº de carga;
  - Nombre del usuario;
  - Cantidad de bloque almacenados en disco.
- Carga: Se produce antes de un log out. Contiene:
  - Un nº de carga;
  - Nombre del usuario;
  - Tiempo de conexión (minutos);

- Requerimientos;
- Bytes leidos;
- Bytes escritos.

Nota: Se produce cuando se arranca la Red. Contiene:
- Nombre del usuario;
- Mensaje de arranque de la Red.

Nota: Se produce cuando se desconecta la Red. Contiene:
- Nombre del usuario;
- Mensaje de desconexión de la Red.

Nota: Se produce al realizarse una conexión, desconexión o detección de intruso. Contiene:
- Nombre del usuario;
- "Conexión" / "Desconexión" / "Intruso";
- Dirección de la terminal.

1.6 Fuentes de Información

» Manuales de NOVELL
» LIDI:
» ALBANO COZZUOL S.A.

Nota: Empresa ALBANO COZZUOL S.A. relevada con una red Novell V2.11 instalada. A los efectos de obtener las necesidades de un usuario real respecto de la seguridad e la red y de su carga de trabajo.

1.7 Estrategia de Solución

Este sistema se divide en dos partes:

La 1ra. parte; será construir una fuente de datos tal que tenga una estructura manejable. Para ello se utilizarán los comandos de *Novell*: PAUDIT y ATOTAL.

Los cuales toma al archivo de *Novell* "NET$ACCT.DAT" y los convierte en archivos de tipo texto "NET$ACCT.TXT" y "NET$TOT.TXT" respectivamente. Luego convertiremos estos archivos, ya manejable de tipo texto, en otros dos del tipo record.

La 2da. parte; comprenderá el proceso de la información de modo tal que esta cubra todos los objetivos y alcances ya definidos.

1.8 Funciones de Programación

Del sistema:

- » Generar archivos record con la información del ACCOUNTING.
- » Borrar información de los archivos a pedido del supervisor.
- » Generar copias de resguardo de la información.

Uso de recursos:

- » Información de totales de recursos consumidos de la Red para un mes dado (discriminado por día).
- » Información general de recursos consumidos en la Red, discriminado por usuario en un mes dado.
- » Información para un usuario de los bloques almacenados en disco a lo largo de un día.
- » Información por usuario de recursos consumidos de la Red a lo largo de un mes.

Seguridad en la red:

- » Determinar por usuario las conexiones realizadas en un mes.
- » Determinar en que terminal/es estuvo conectado un usuario en un mismo período de tiempo.
- » Detectar intrusos en la red.
- » Detectar Desconexiones incorrectas.

Facturación:

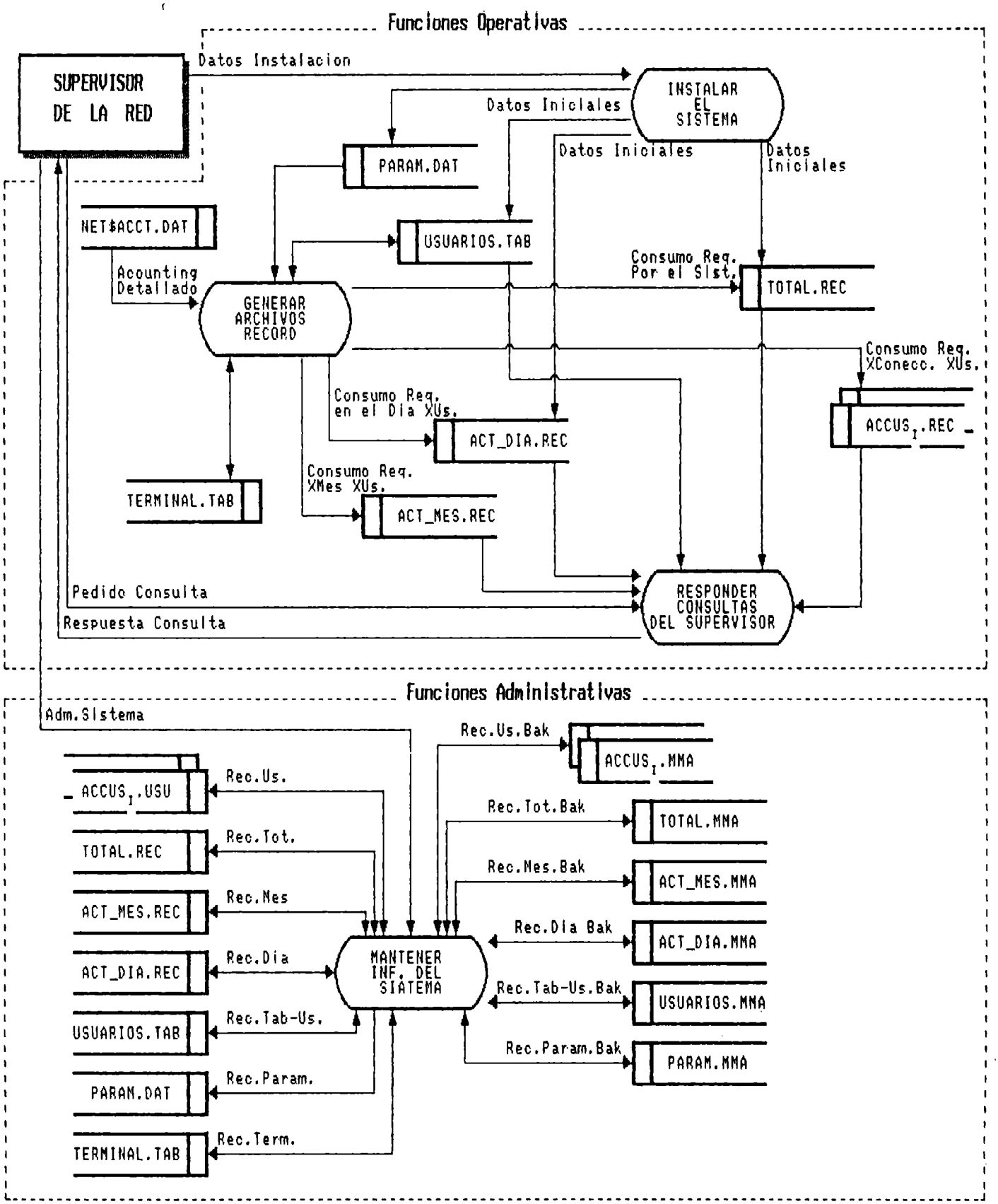
- » Facturación a los usuarios para un mes dado.

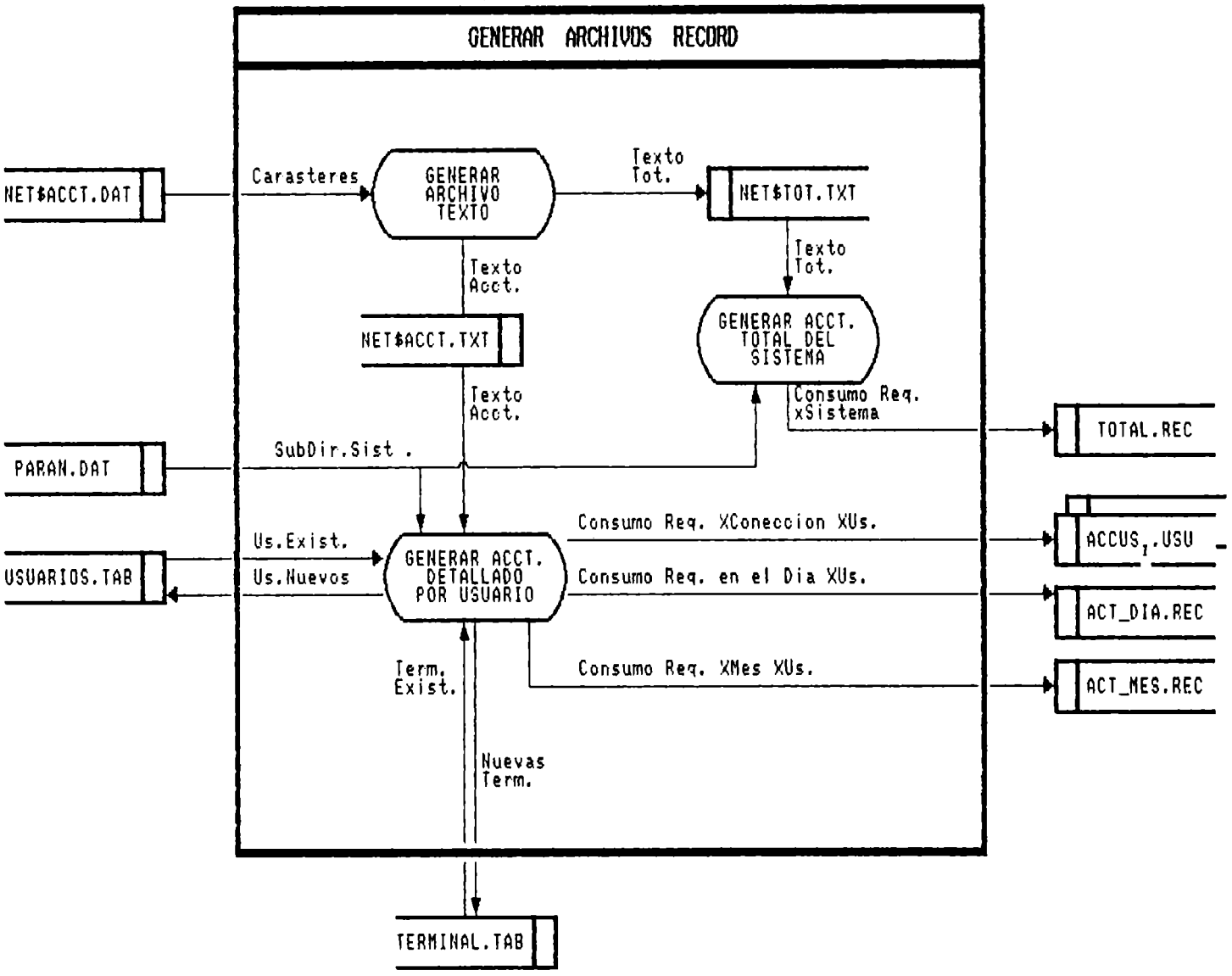
1.8 Consideraciones Generales a Tener en Cuenta por el Supervisor(es) y los Usuarios

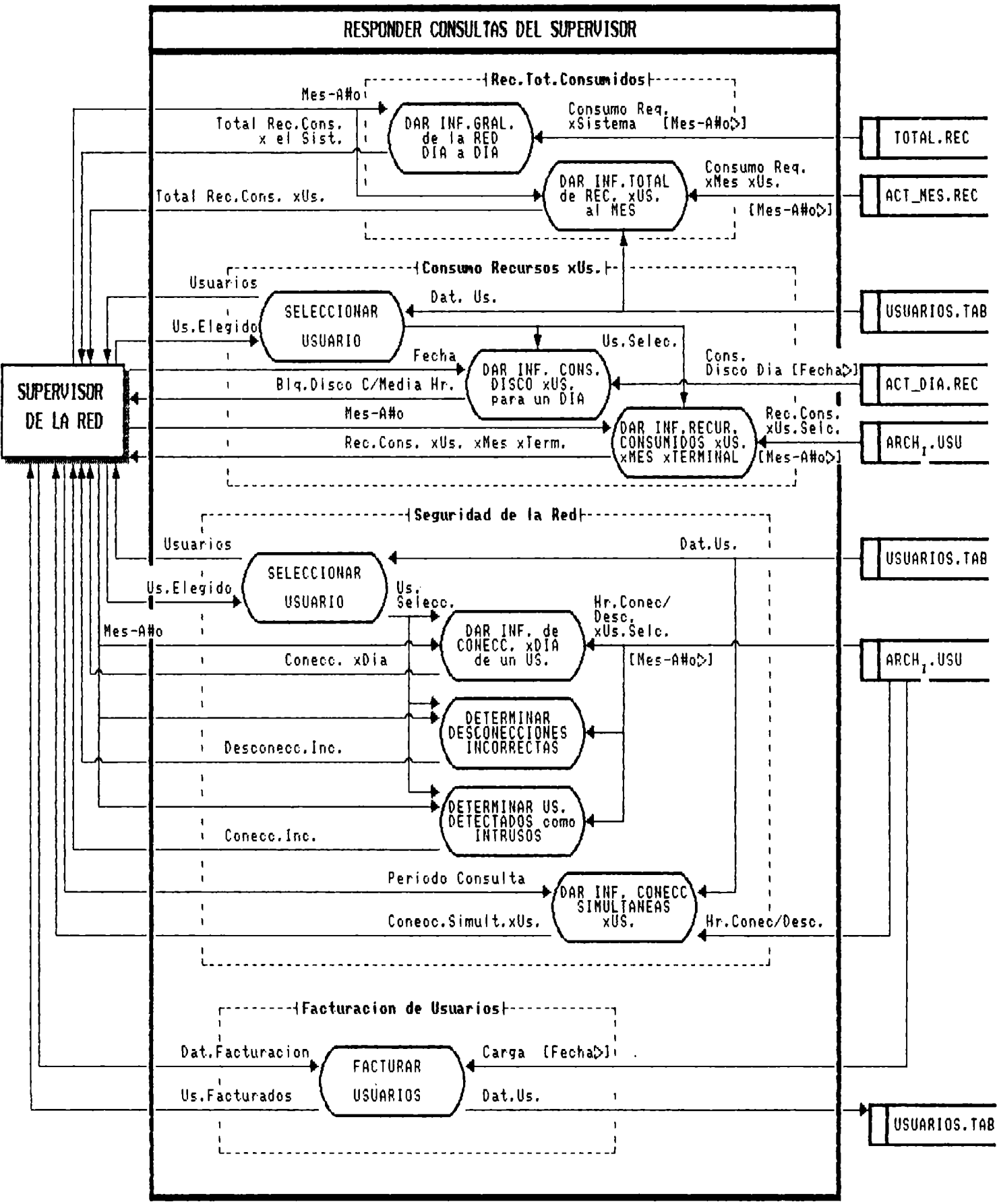
1- Que el sistema se corra con cierta frecuencia, preferentemente todos los días. Esta consideración está fundada en el considerable tiempo que demora *Novell* al ejecutar los comandos "PAUDIT" Y "ATOTAL" cuando en el archivo NET$ACCT.DAT se almacena información de períodos largos de tiempo.
2- Que los usuarios se desconecten al finalizar el día.
3- Que se baje la red antes de apagar (o desconectar) el SERVER de la red.
4- Que la red se baje al menos una vez al día.

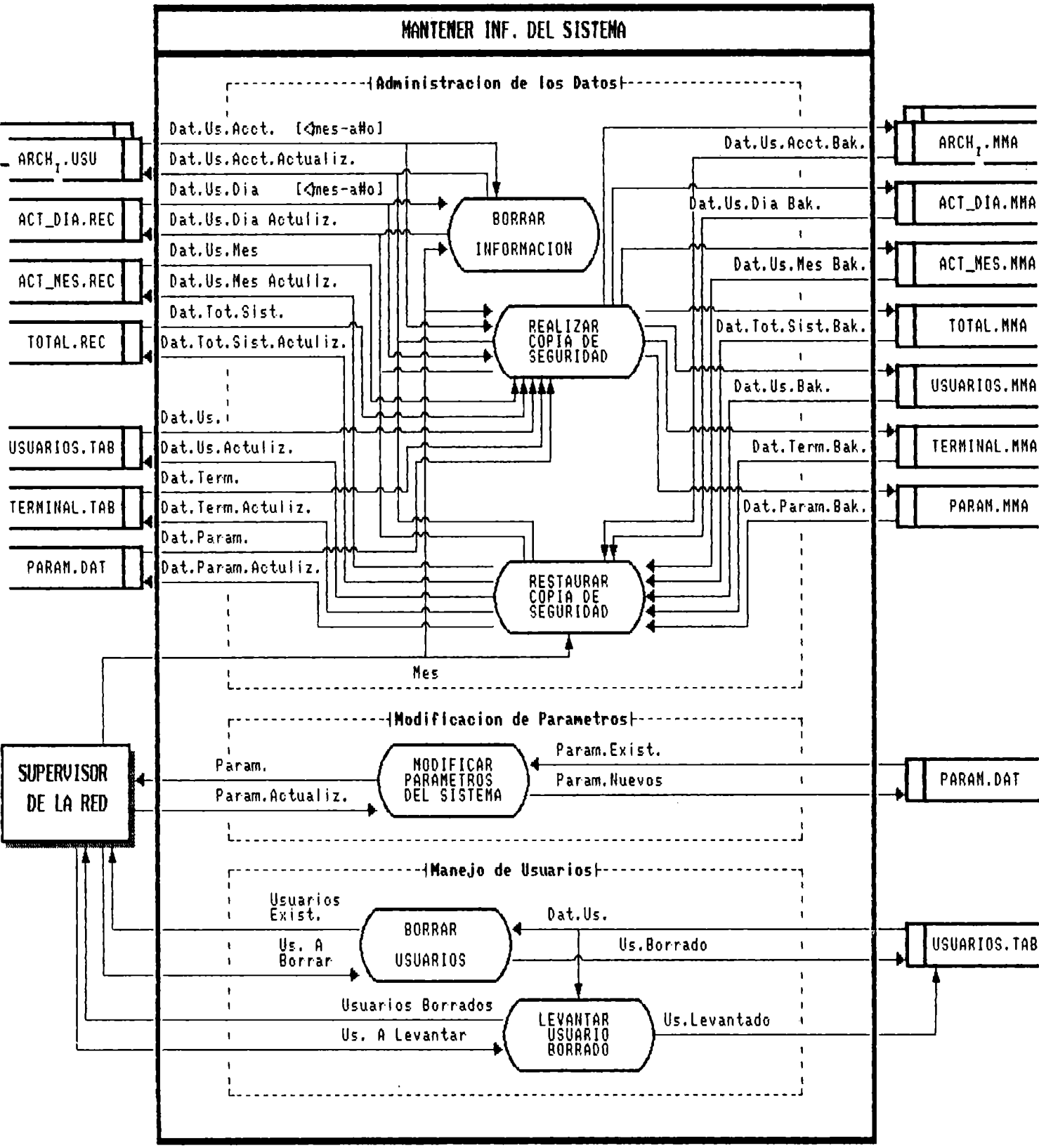
2 Diseño Estructurtado

2.1 Diagrama de Flujo de Datos




GENERAR ARCHIVOS RECORD
NET$ACCT.DAT
Carasteres
GENERAR ARCHIVO TEXTO
Texto Tot.
NET$TOT.TXT
Texto Acct.
NET$ACCT.TXT
Texto Tot.
GENERAR ACCT. TOTAL DEL SISTEMA
Consumo Req. xSistema
TOTAL.REC
PARAN.DAT
SubDir.Sist .
Consumo Req. XConeccion XUs.
ACCUS_I.USU
Us.Exist.
USUARIOS.TAB
Us.Nuevos
GENERAR ACCT. DETALLADO POR USUARIO
Consumo Req. en el Dia XUs.
ACT_DIA.REC
Consumo Req. XMes XUs.
ACT_MES.REC
Term. Exist.
Nuevas Term.
TERMINAL.TAB


RESPONDER CONSULTAS DEL SUPERVISOR
Rec.Tot.Consumidos
Mes-A#o
Total Rec.Cons. x el Sist.
DAR INF.GRAL. de la RED DIA a DIA
Consumo Req. xSistema [Mes-A#o▷]
TOTAL.REC
Consumo Req. xMes xUs.
Total Rec.Cons. xUs.
DAR INF.TOTAL de REC. xUS. al MES
[Mes-A#o▷]
ACT_MES.REC
Consumo Recursos xUs.
Usuarios
Dat. Us.
USUARIOS.TAB
Us.Elegido
SELECCIONAR USUARIO
Us.Selec.
Fecha
Cons. Disco Dia [Fecha▷]
Blq.Disco C/Media Hr.
DAR INF. CONS. DISCO xUS. para un DIA
ACT_DIA.REC
SUPERVISOR DE LA RED
Mes-A#o
Rec.Cons. xUs. xMes xTerm.
DAR INF.RECUR. CONSUMIDOS xUS. xMES xTERMINAL
Rec.Cons. xUs.Selc.
[Mes-A#o▷]
ARCH$_1$.USU
Seguridad de la Red
Usuarios
Dat.Us.
USUARIOS.TAB
Us.Elegido
SELECCIONAR USUARIO
Us. Selecc.
Hr.Conec/ Desc. xUs.Selc.
Mes-A#o
DAR INF. de CONECC. xDIA de un US.
Conecc. xDia
[Mes-A#o▷]
ARCH$_1$.USU
DETERMINAR DESCONECCIONES INCORRECTAS
Desconecc.Inc.
DETERMINAR US. DETECTADOS como INTRUSOS
Conecc.Inc.
Periodo Consulta
DAR INF. CONECC SIMULTANEAS xUS.
Conecc.Simult.xUs.
Hr.Conec/Desc.
Facturacion de Usuarios
Dat.Facturacion
FACTURAR USUARIOS
Carga [Fecha▷]
Us.Facturados
Dat.Us.
USUARIOS.TAB


MANTENER INF. DEL SISTEMA
Administracion de los Datos
Dat.Us.Acct. [<mes-a#o]
Dat.Us.Acct.Actualiz.
Dat.Us.Dia [<mes-a#o]
Dat.Us.Dia Actuliz.
Dat.Us.Mes
Dat.Us.Mes Actuliz.
Dat.Tot.Sist.
Dat.Tot.Sist.Actuliz.
Dat.Us.
Dat.Us.Actuliz.
Dat.Term.
Dat.Term.Actuliz.
Dat.Param.
Dat.Param.Actuliz.
Mes
BORRAR INFORMACION
REALIZAR COPIA DE SEGURIDAD
RESTAURAR COPIA DE SEGURIDAD
Dat.Us.Acct.Bak.
Dat.Us.Dia Bak.
Dat.Us.Mes Bak.
Dat.Tot.Sist.Bak.
Dat.Us.Bak.
Dat.Term.Bak.
Dat.Param.Bak.
ARCH_I.USU
ACT_DIA.REC
ACT_MES.REC
TOTAL.REC
USUARIOS.TAB
TERMINAL.TAB
PARAM.DAT
ARCH_I.MMA
ACT_DIA.MMA
ACT_MES.MMA
TOTAL.MMA
USUARIOS.MMA
TERMINAL.MMA
PARAM.MMA
Modificacion de Parametros
SUPERVISOR DE LA RED
Param.
Param.Actualiz.
MODIFICAR PARAMETROS DEL SISTEMA
Param.Exist.
Param.Nuevos
PARAM.DAT
Manejo de Usuarios
Usuarios Exist.
Us. A Borrar
BORRAR USUARIOS
Dat.Us.
Us.Borrado
USUARIOS.TAB
Usuarios Borrados
Us. A Levantar
LEVANTAR USUARIO BORRADO
Us.Levantado

2.2 Diccionario de Datos

Estructura de los Flujo de Datos:

NOMBRE	CONTENIDO
Blq. Disco c/Media Hr.:	fecha hora alm. en disco
Caracteres:	caracteres ASCII
Conex.Int	usuario fecha hr. conexión
Conex. xDia	usuario fecha terminal hr. conexión hr. desconexión
Conex.Simult. xUs.	usuario fecha terminal hr. conexión hr. desconexión
Consumo Disco Dia:	registro: ACCDIA
Consumo Req. en el dia:	registro: ACCDIA
Consumo Req. xConexión xUs.:	registro: ACCUS
Consumo Req. xSistema:	registro: TOTAL
Consumo Req. xMes xUs.:	registro: REG_MES
Dat.Facturación:	fecha desde fecha hasta pesos
Dat.Param.	registro: PARAMETROS
Dat.Param.Actualiz.	registro: PARAMETROS
Dat.Param.Bak.	registro: PARAMETROS
Dat.Term.	registro: REC_TERM
Dat.Term.Actualiz.	registro: REC_TERM
Dat.Term.Bak	registro: REC_TERM
Dat.Us.Acct.:	registro: ACCUS
Dat.Us.Acct.Actualiz.	registro: ACCUS
Dat.Us.Acct.Bak.	registro: ACCUS

NOMBRE	CONTENIDO
Dat.Us.	registro: REC_TAB
Dat.Us.Actualiz.	registro: REC_TAB
Dat.Us.Bak.	registro: REC_TAB
Dat.Us.Día	registro: ACCDIA
Dat.Us.Día Actualiz.	registro: ACCDIA
Dat.Us.Día Bak	registro: ACCDIA
Dat.Us.Mes	registro: REG_MES
Dat.Us.Mes.Actuliz	registro: REG_MES
Dat.Us.Mes Bak.	registro: REG_MES
Dat.Tot.Sist.	registro: TOTAL
Dat.Tot.Sist.Actuliz.	registro: TOTAL
Dat.Tot.Sist.Bak.	registro: TOTAL
Desconex.Inc.	usuario fecha terminal hr. conex.
Hr.Conex/Desc.	usuario fecha terminal hr. conexión hr. desconexión
Hr.Conex/Desc. xUs.Selec.	fecha terminal hr. conexión hr. desconexión
Mes-Año	mes año
Nueva Term.:	registro: REC_TERM
Param.:	subdir tipo placa video
Param.Actualiz.	Idem Param.
Param.Exist	subdir inic_mem_video long_mem_video
Param.Nuevos	Idem Param.Exist.
Período Consulta	fecha desde fecha hasta

NOMBRE	CONTENIDO
SubDir.Sist:	subdir
Req. Cons. xUs. Selec.:	registro: ACCUS
Req. Cons. xUs. xMes. xTerm.:	terminal mes día req. server tpo. consumido Blq. leidos blq. escritos
Term. Exist.:	registro: REC_TERM
Texto Acct.:	texto string(80)
Texto Tot.:	texto, string(80)
Total Rec. Cons. x el Sist.:	mes día tpo. consumido req. server blq. leidos blq. escritos blq. disco/día
Total Rec. Cons. x Us.:	mes usuario tpo. consumido req. server blq. leídos blq. escritos conex. c/intr.
Us. Facturado:	usuario carga pesos
Us. A Borra	usuarios activos
Us. A Levantar	usuarios inactivos
Us. Borrado	baja = *
Us. Elegido	usuario
Us.Exist.:	registro: REC_TAB
Us. Levantado	baja = ' '
Us.Nuevo:	registro: REC_TAB
Usuarios Borrados	usuario inactivos
Us.Selecc.	registro: REC_TAB

Estructura de los Archivos:

ARCHIVO	REGISTRO	DESCRIPCION
NET$ACCT.DAT		Generado por Novell, registrando ciertos acontecimiento que suceden en la Red.
NET$ACCT.TXT		Texto generado por el comando de Novell: " PAUDIT > NET$ACCT.TXT " Este archivo de texto contiene los suceso detallados de cada usuario a medida que estos se van produciendo a través del tiempo.
NET$TOT.TXT		Texto generado por el comando de Novell: " ATOTAL > NET$TOT.TXT " Este archivo de texto el contiene el total de los recursos consumidos por todos los usuarios a lo largo de un día.
ARCHi.USU	ACCUS	Contiene información del consuno de cada uno los recursos consumidos por el i-ésimo usuario a lo largo del día. Hay un archivo por usuario.
ACT_DIA.REC	ACCDIA	Contiene la carga total asignada a un usuario a lo largo del día y cada media hora.
ACT_MES.REC	REG_MES	Contiene el total de recursos consumidos por cada usuario a lo largo de un mes.
TOTAL.REC	TOTAL	Contiene el total por día de los recursos consumido por la totalidad de los usuarios.
USUARIOS.TAB	REC_TAB	Tabla donde se relaciona para cada usuario que archivo (ARCHi.USU) le corresponde.
TERMINAL.TAB	REC_TERM	Tabla donde se relaciona cada dirección de terminal con un número entero.

ARCHIVO	REGISTRO	DESCRIPCION
PARAM.DAT	PARAMETROS	Contiene parámetros del sistema.

Estructura de Datos de los Registros:

NOMBRE DEL REGISTRO	ESTRUCTURA DEL REGISTRO	DESCRIPCION
ACCUS:	primero	primero=TRUE(rØ) indica reg. cero
	(rØ)punt_día	Puntero al primer registro correspondiente al día en curso.
	(rØ)punt_mes	Idem al mes.
	(rØ)cant_día	Cantidad de días muestreados
	(rØ)f_act	Ultima fecha de actualización del archivo.
	(rØ)mcarga	Carga acumulada.
	(rØ)mtpo_conex	Idem tpo. conexión.
	(rØ)mreq	Idem. requerimientos al Server.
	(rØ)mblq_leídos	Idem blq. leídos.
	(rØ)mblq_esc	Idem blq. escritos.
	fecha	Fecha.
	hr_conec	Hora conexión.
	hr_desc	Hora desconexión.
	terminal	Nro. de terminal.
	intruso	Si fue detectado intruso.
	carga	Cantidad cargada.
	tpo_conex	Tiempo de conexión.
	req	Requerimiento al Server.
	blq_leídos	Bloques leídos.
	blq_esc	Bloques escritos.
ACCDIA:	primero	primero=TRUE(rØ) indica reg. cero
	(rØ)punt	Puntero.
	usu	Nombre del usuario.
	carga	Cantidad cargada
	fecha	Fecha
	vec	Vector conteniendo la carga del us. cada media hora.
REG_MES:	usuario	Nombre del usuario.
	mes	Mes de la inf.
	anio	Anio de la inf.
	cant_intr	Cant. veces detectado intruso
	carga	Cant. cargada en el mes.
	tpo_conex	Tpo. conectado en el mes.
	req.	Req. al Server en el mes
	blq_leídos	Blq. leídos en el mes.
	blq_esc	Blq. escritos en el mes.

NOMBRE DEL REGISTRO	ESTRUCTURA DEL REGISTRO	DESCRIPCION
TOTAL:	primero	primero=TRUE(rØ) indica reg. cero
	(rØ)punt	Puntero.
	fecha	Fecha de la inf.
	tpo_conex	Tpo. conexión del sistema.
	req	Requerimientos al Server.
	blq_leídos	Blq. leídos consumidos.
	blq_esc	Blq. escritos consumidos.
	blq_disc	Blq. disco consumidos.
REC_TAB:	primero	primero=TRUE(rØ) indica reg. cero
	(rØ)cant_us	Cantidad de usuarios.
	usuario	Nombre del usuario.
	nom_arch	Nombre del archivo.
	baja	baja=* Marc de borado.
REC_TERM:	dir_term	Dirección de la terminal.
	nro_term	Número de la terminal.
PARAMETROS:	subdir	Sub-Directorio donde se encuentra el sistema. Por defecto el corriente
	inic_mem_video	Puntero donde inicia la memoria de video. Por defecto HERCULES.
	long_mem_video	Longitud de la memoria de video. Por defecto HERCULES.
	calidad_print	Calidad de la impresión gráfica. Por defecto baja.
	mes	Mes correspondiente a la última actualización del ACT_MES.REC.
	anio	Idem año.

Datos Elementales de los Registros:

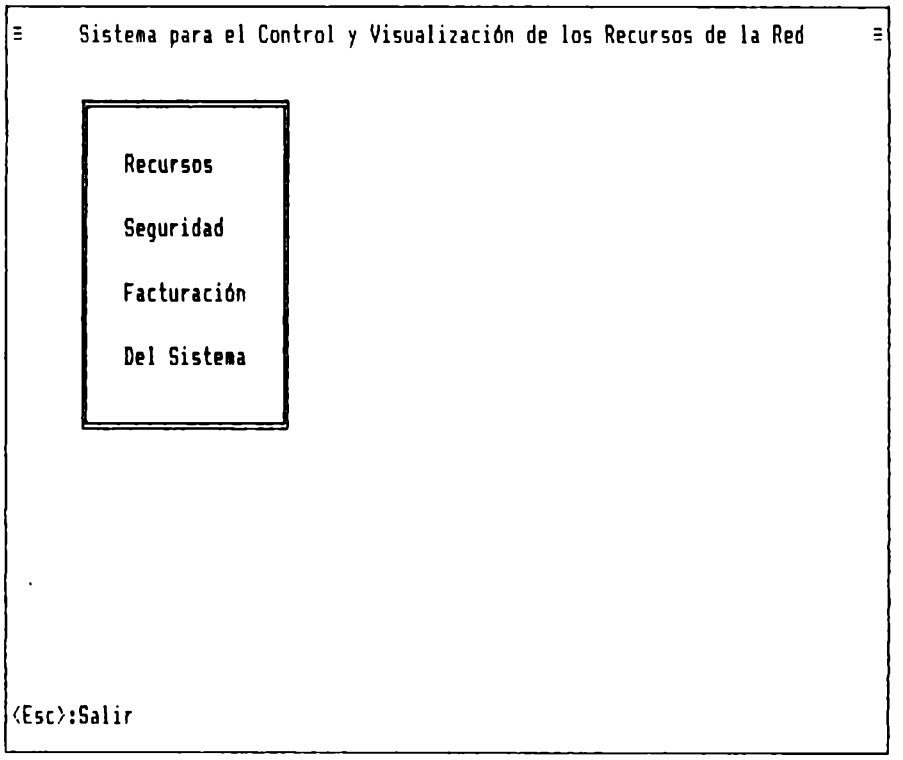
NOMBRE DEL CAMPO	TIPO
anio	byte
baja	char
blq_esc	longint
blq_leídos	longint
cant_día	integer
cant_intr	byte
cant_us	integer
calidad_print	boolean. TRUE=Alta Calidad
carga	longint
dir_term	str12
f_act	str8
fecha	str8
hr_conec	str8
hr_desc	str8
inic_mem_video	word
intruso	boolean
long_mem_video	word
nblq_esc	longint
nblq_leídos	longint
ncarga	longint
nes	meses
neses	(0..12)
ntpo_conec	longint
nreq	longint
nro_term	integer
pesos	real
primero	boolean
punt	word
punt_día	word
punt_mes	word
req	longint
subdir	string
terminal	integer
tpo_conec	longint
usu	str12
usuario	str12
vec	array(1..48) of longint

2.3 Diseño de Pantallas

El sistema responderá a sus objetivos separando a estos en cuatro módulos principales: *Recursos* consumidos por los usuarios y el sistema en general; información inherente a *Seguridad*; consultar costo($) consumido por cada usuario en *Facturación* y un opción *Del Sistema* para realizar el mantenimiento del mismo.

Pantalla inicial:

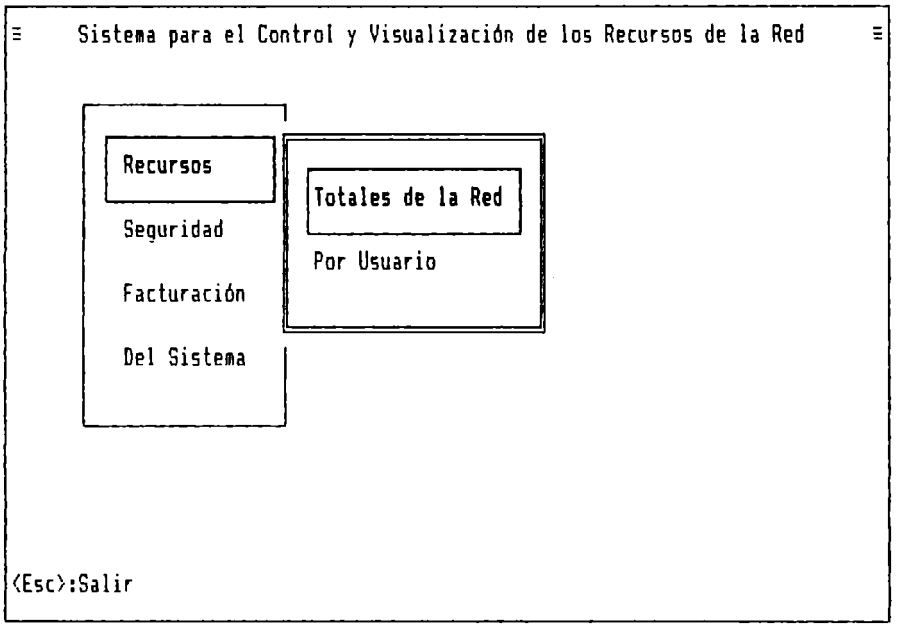
PANTALLA 1



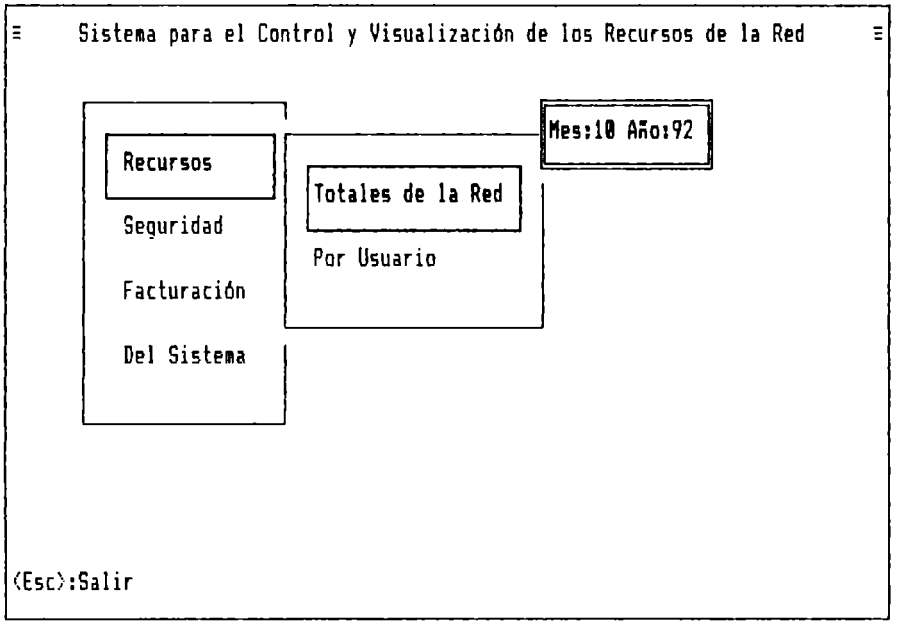
Módulo 1: ***RECURSOS.*** Provee información respecto de los recursos consumidos de la red, tales como, tiempo (minutos), requerimientos al server (cantidad), bloques (4 Kb): leídos, escritos y consumidos por día.

Modulo 1.1: Totales de recursos de la Red consumidos de la red para un mes y año dado (dado por el usuario), discriminados por día.

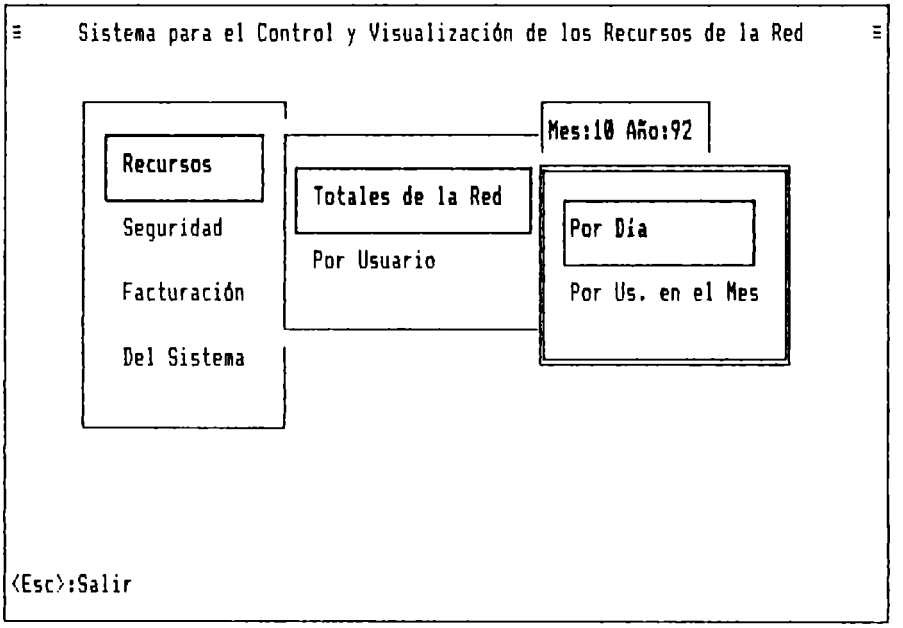
PANTALLA 1.1.1



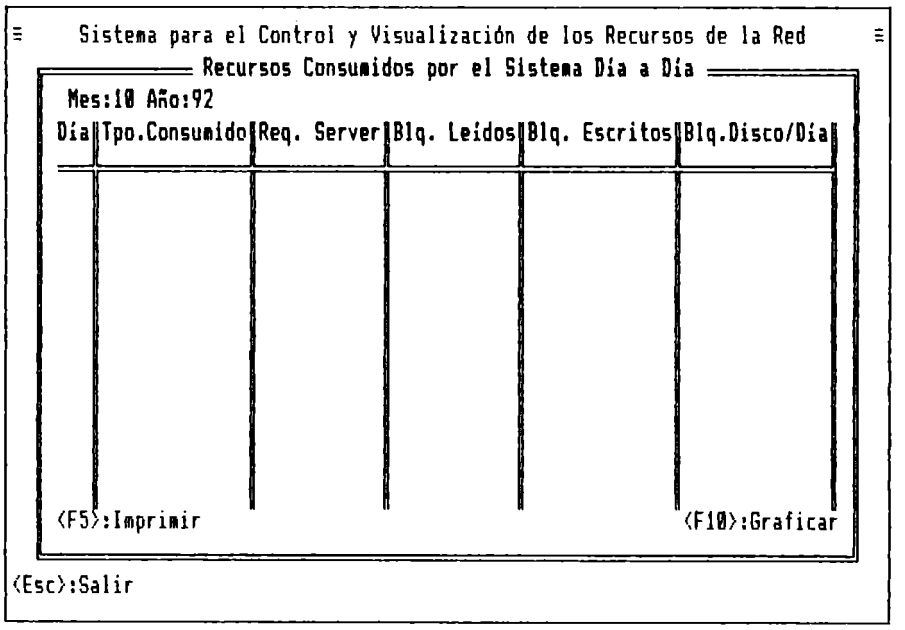
PANTALLA 1.1.2



PANTALLA 1.1.3

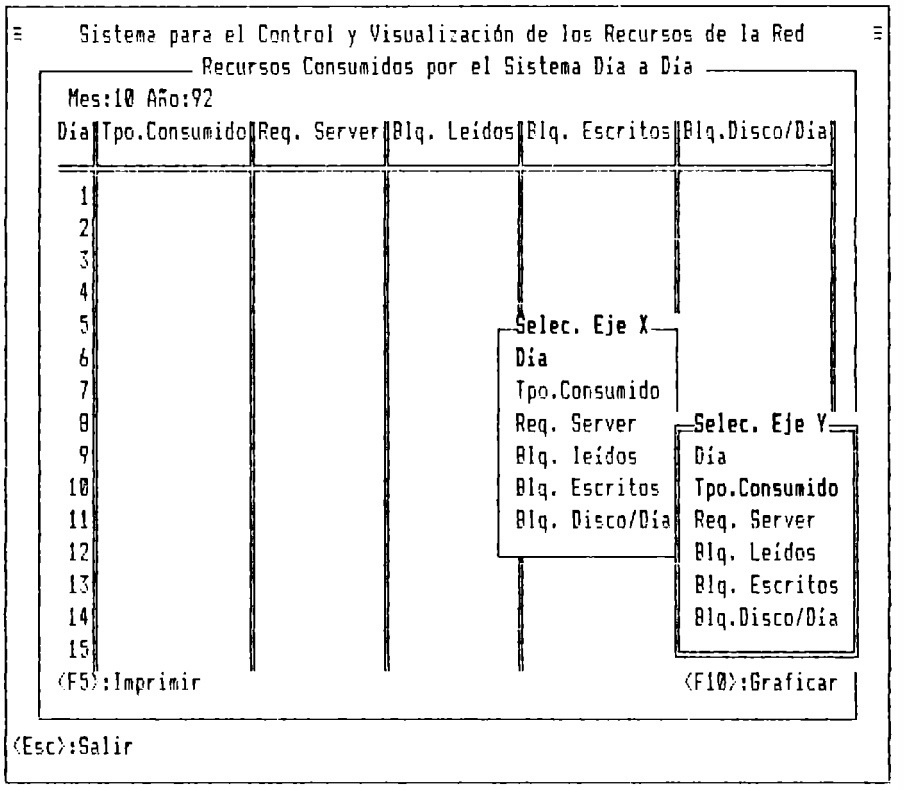


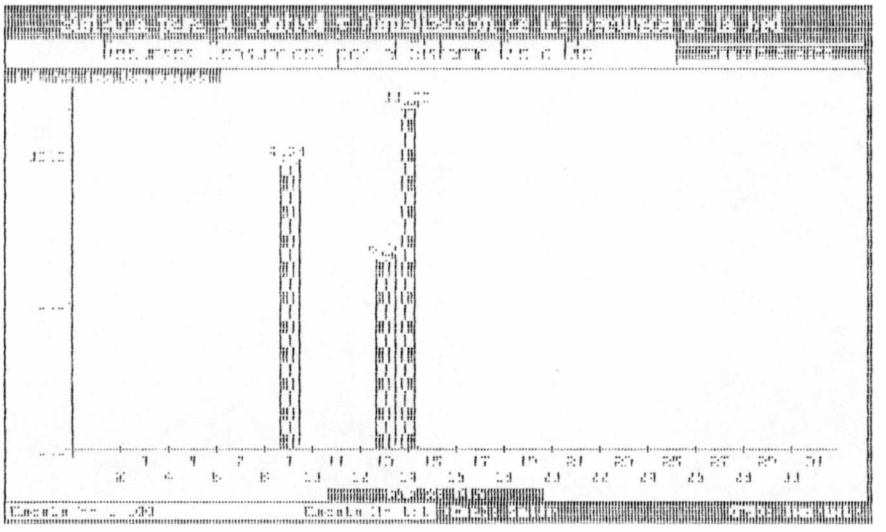
PANTALLA 1.1.4



Para obtener una salida gráfica el usuario selecciona que columna o información quiere representar en cada eje. En las siguientes dos pantallas.

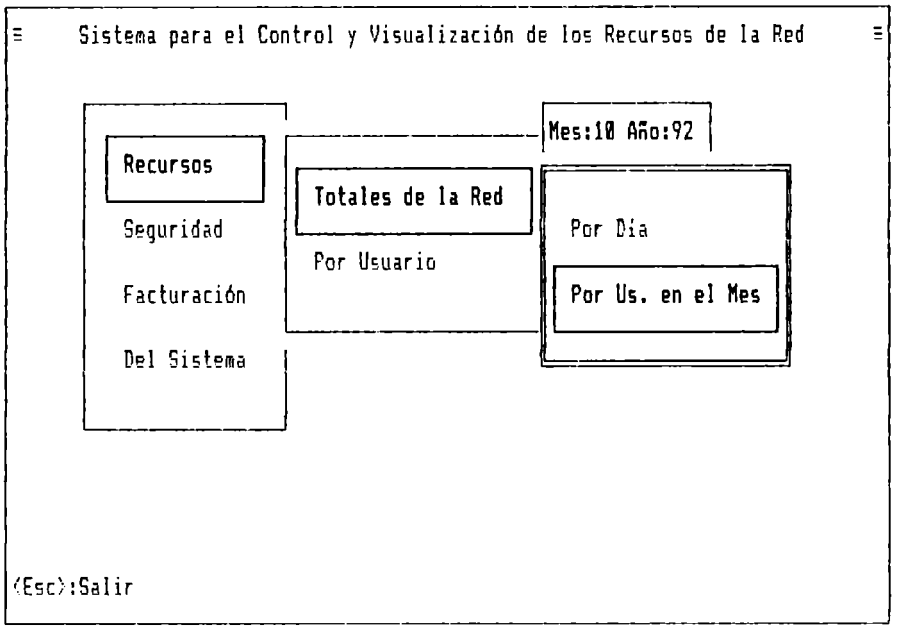
PANTALLA 1.1.5



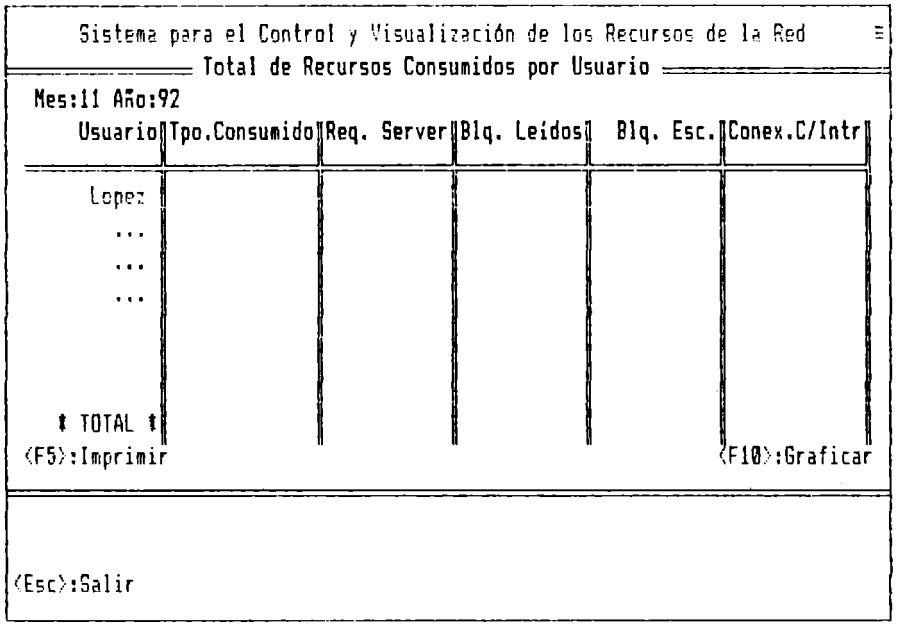


Modulo 1.2: Información gral. de recursos consumidos en la red en un mes dado, discriminado por usuario. Hasta la PANTALLA 1.2, cambiando en la PANTALLA 1.3 por:

PANTALLA 1.2.3



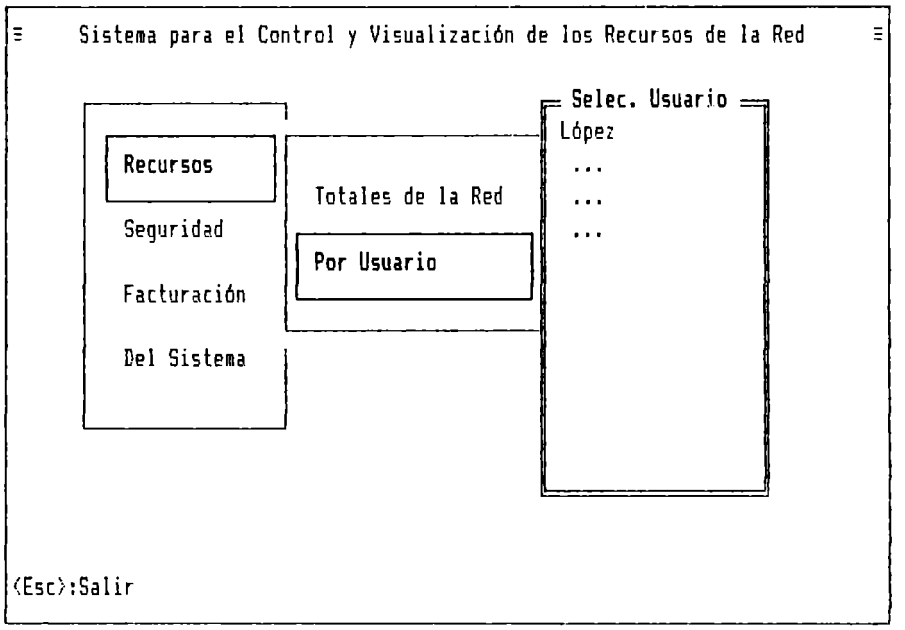
PANTALLA 1.2.4



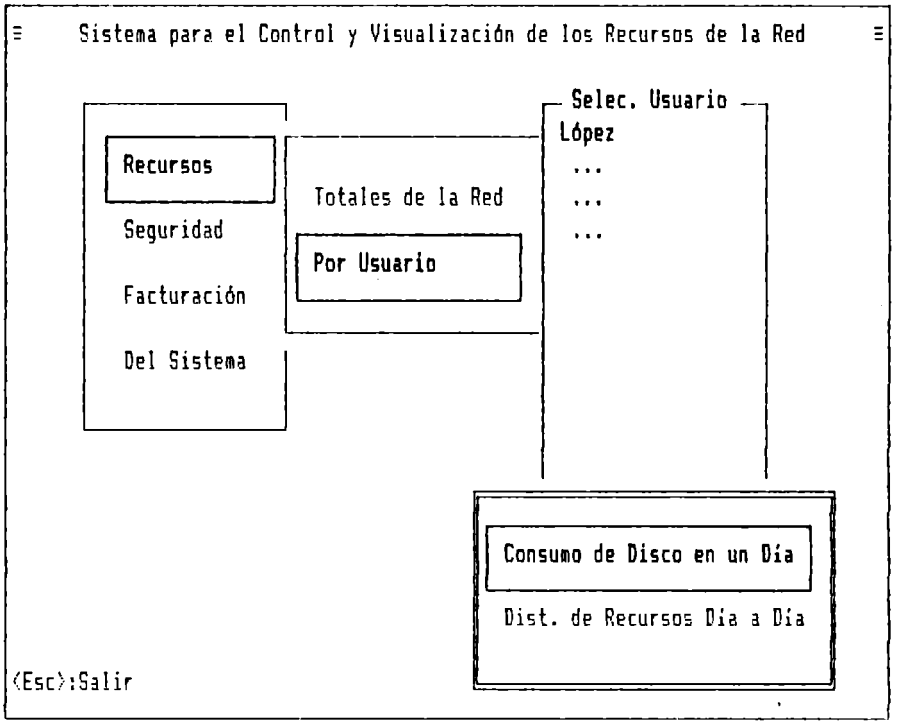
La salida gráfica es equivalente a la anterior.

Modulo 1.3: Bloques almacenados en disco para un usuario dado en un dia. Esta información se muestra actualizada cada media hora.

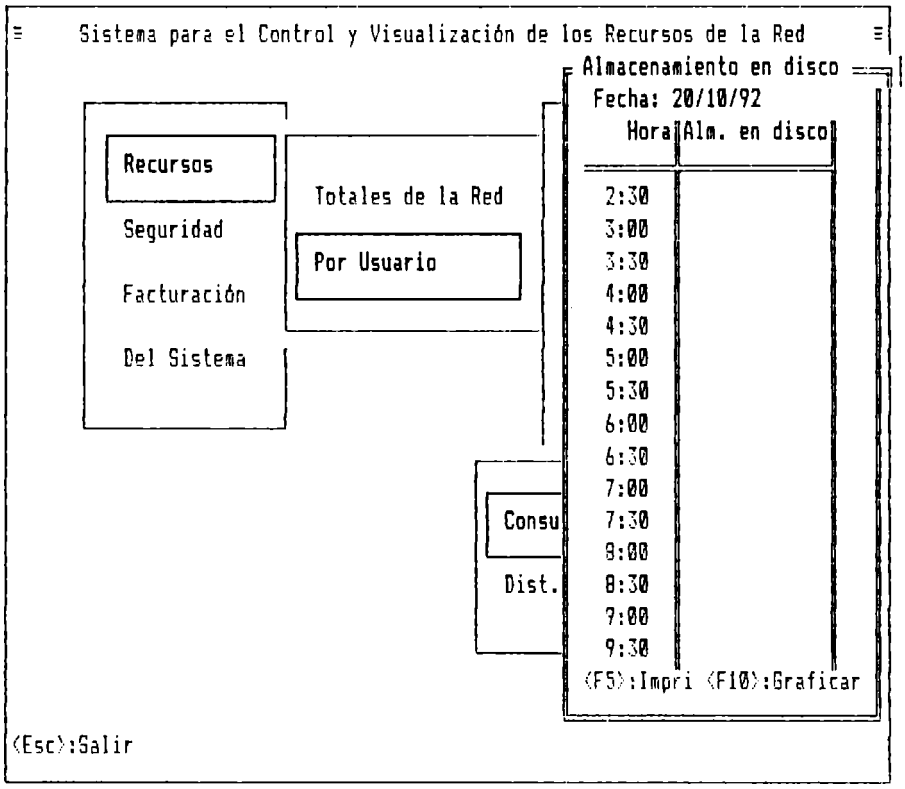
PANTALLA 1.3.1



PANTALLA 1.3.2



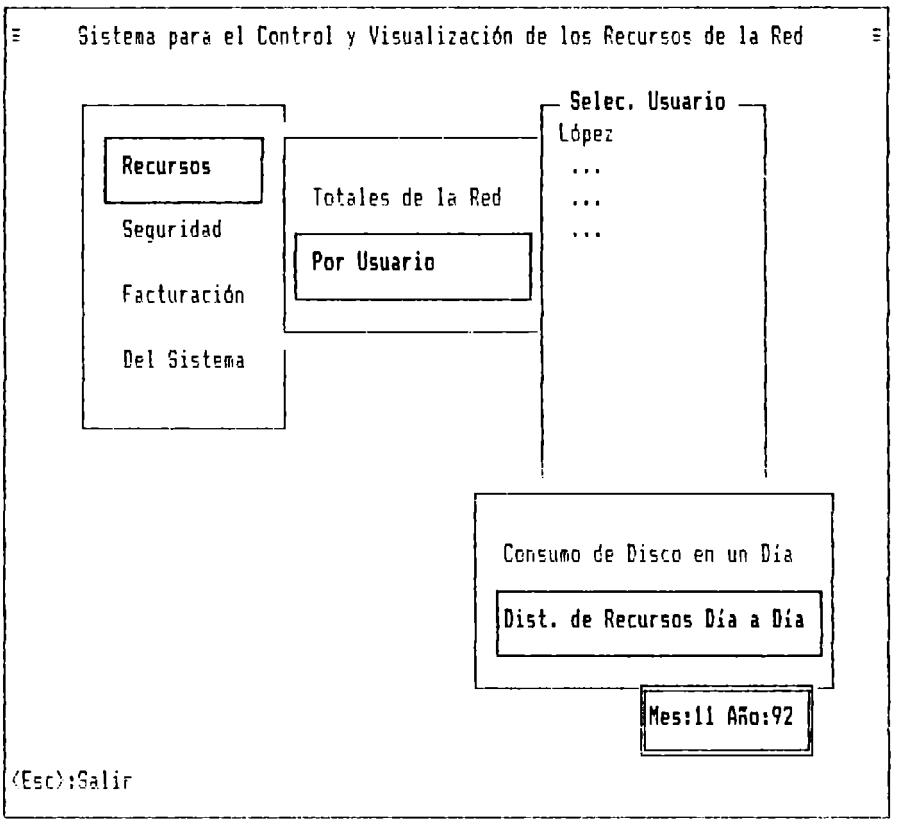
PANTALLA 1.3.3



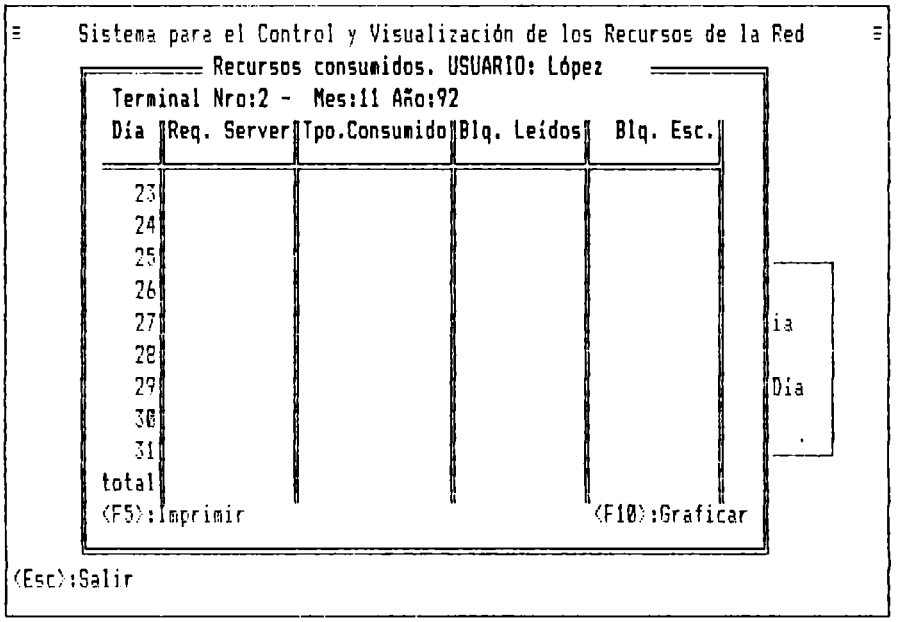
La salida gráfica es equivalente a las anteriores

Modulo 1.4: Recursos consumidos de la red por usuario a lo largo de un mes para cada una de las terminales en que se conectó; discriminado por día. Hasta la PANTALLA 1.3.1 es igual.

PANTALLA 1.4.2



PANTALLA 1.4.3



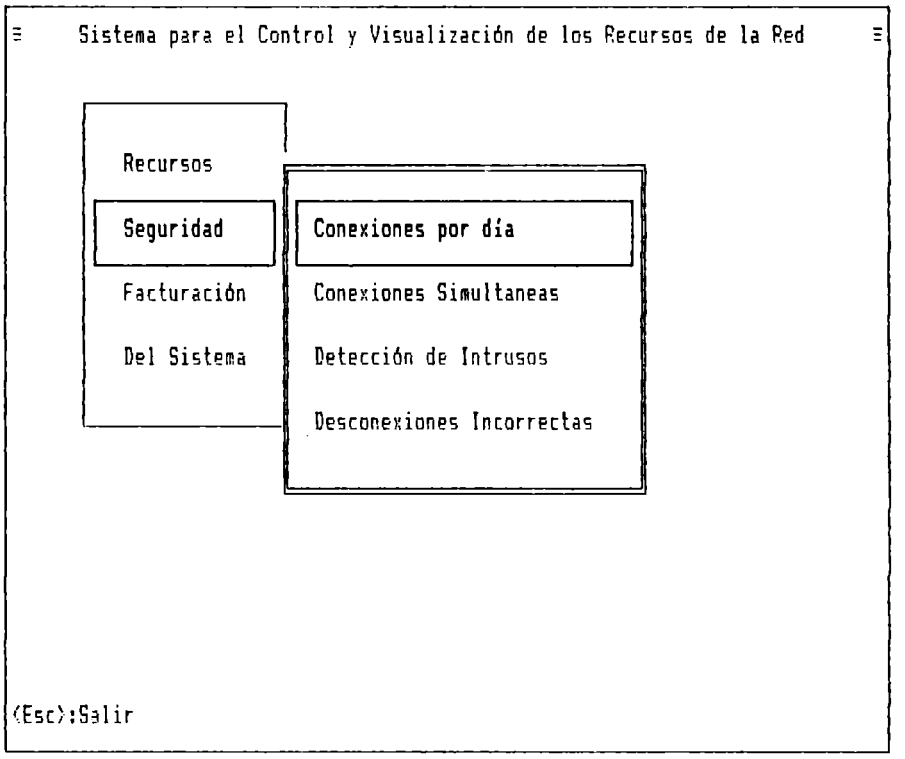
La salida gráfica es equivalente a las anteriores.

Modulo 2: SEGURIDAD. Brinda información inherente a la seguridad de la red:

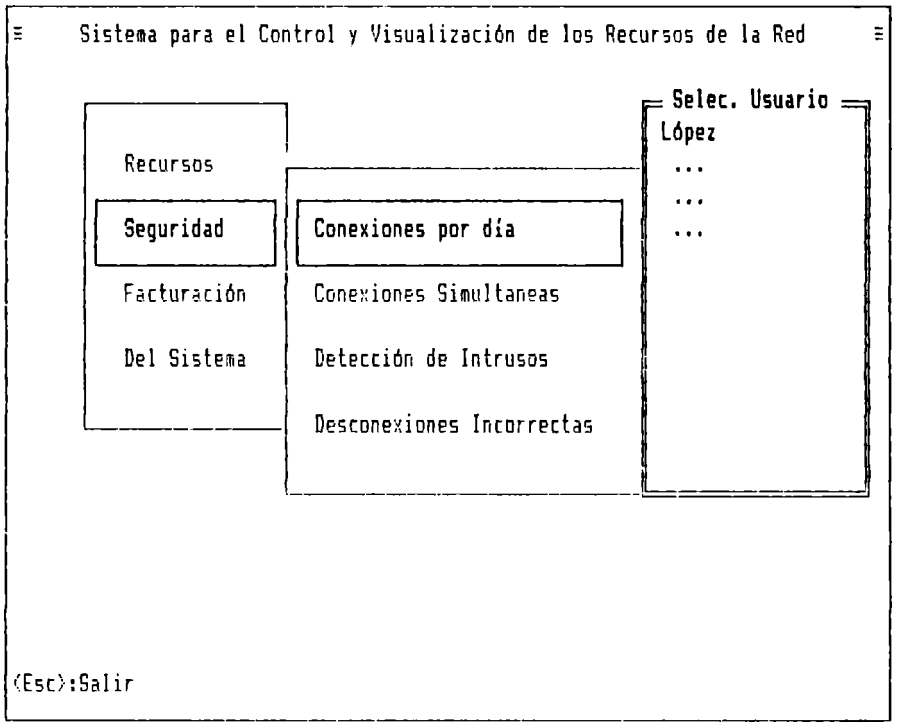
Modulo 2.1: Brinda información de la hora de conexión y desconexión en cada terminal para cada usuario de la red a lo largo de un mes.

Una vez esplayada la información antes mencionada, da la posibilidad de una salida gráfica con el objetivo de visualizar las conexiones simultáneas realizadas por el usuario en distintas terminales para un día dado.

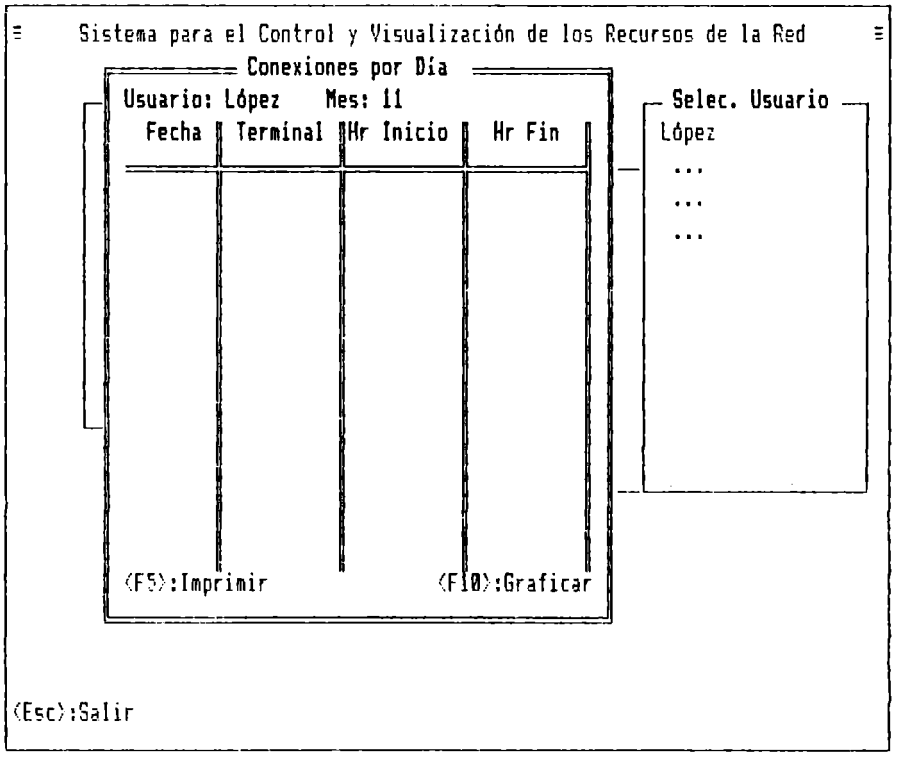
PANTALLA 2.1.1



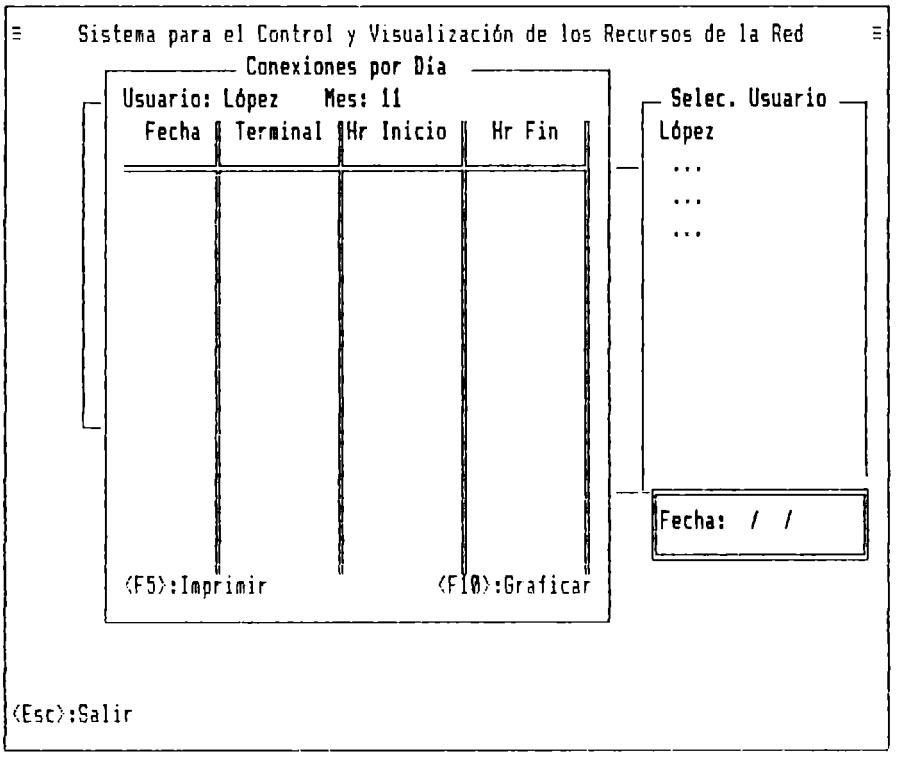
PANTALLA 2.1.2



PANTALLA 2.1.3



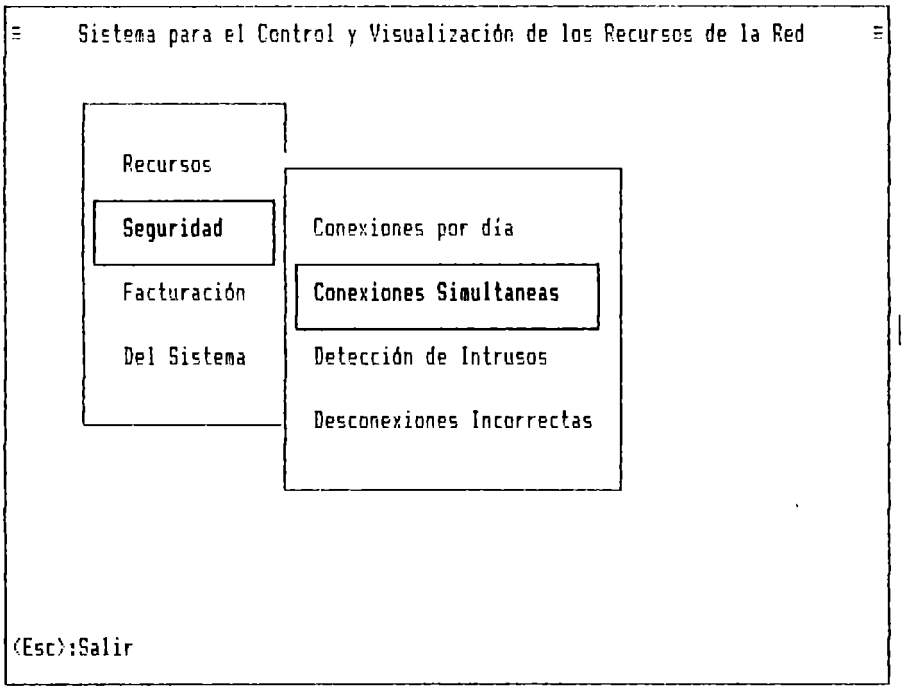
PANTALLA 2.1.4



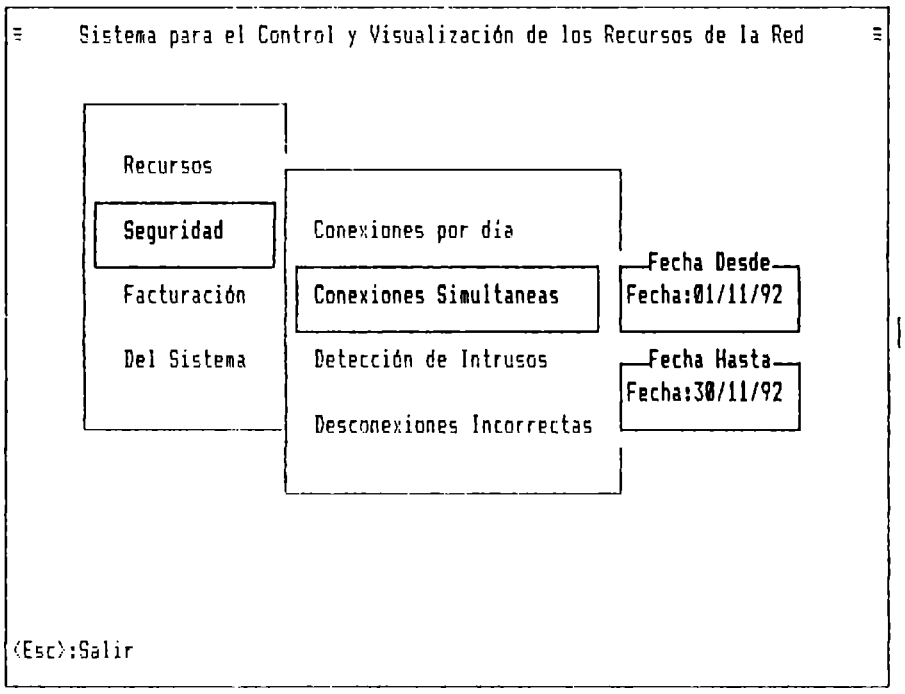
La salida gráfica es equivalente a las anteriores.

Modulo 2.2: Proporciona las conexiones simultáneas realizadas por todos los usuarios de la red entre dos fechas determinadas.

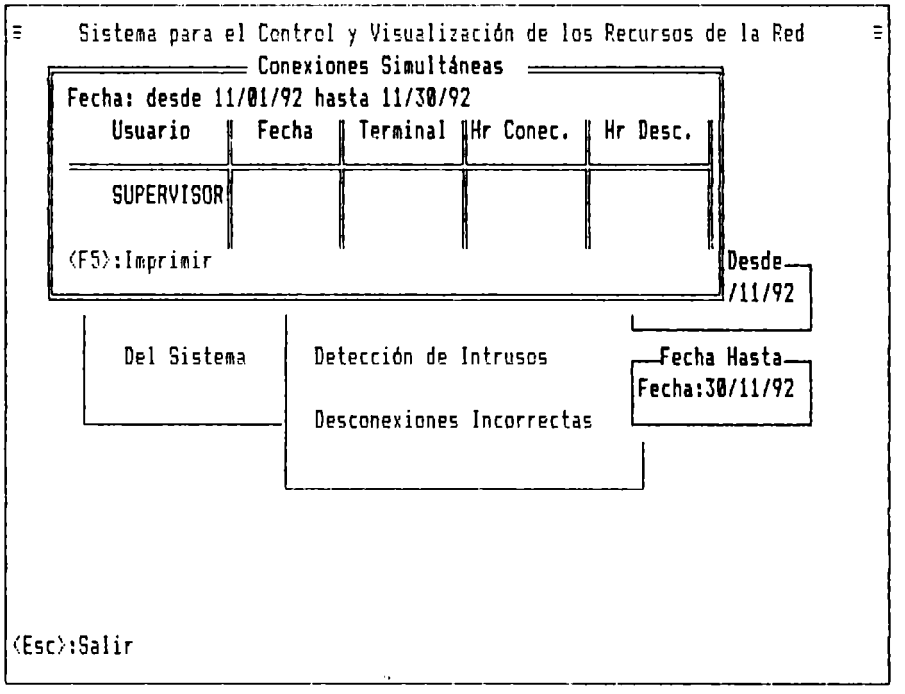
PANTALLA 2.2.1



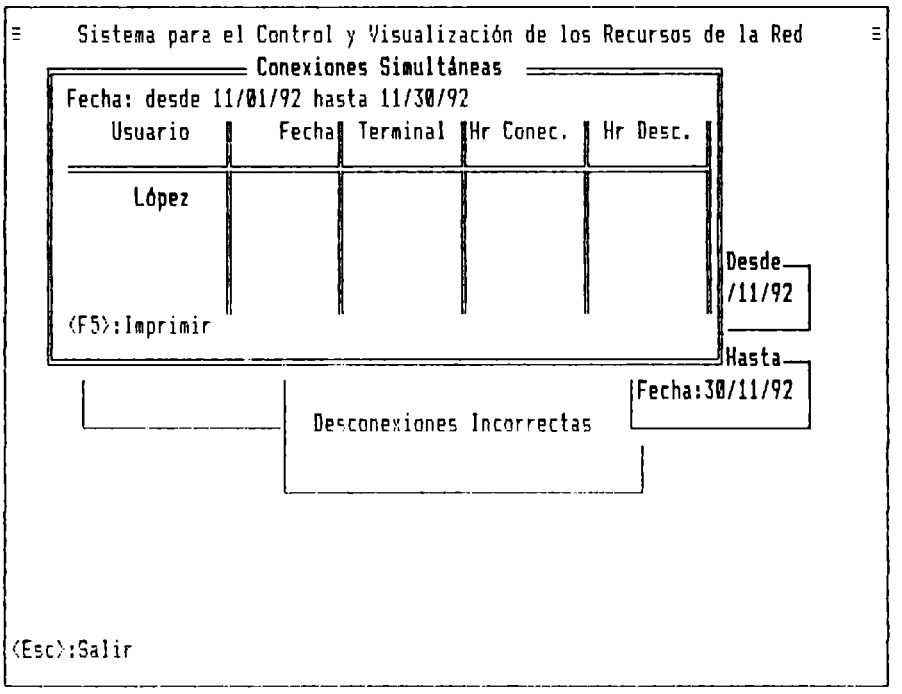
PANTALLA 2.2.2



PANTALLA 2.2.3

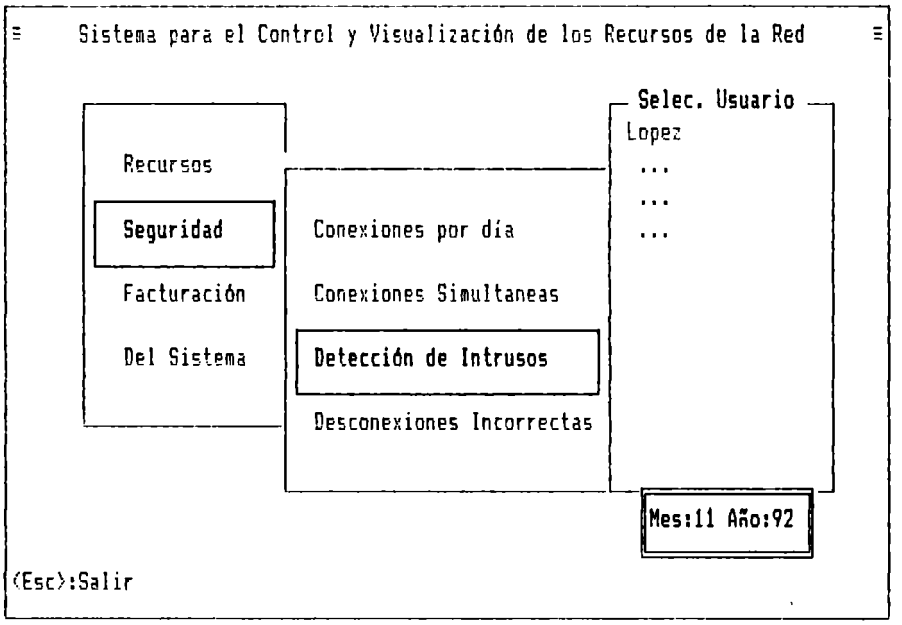


PANTALLA 2.2.4

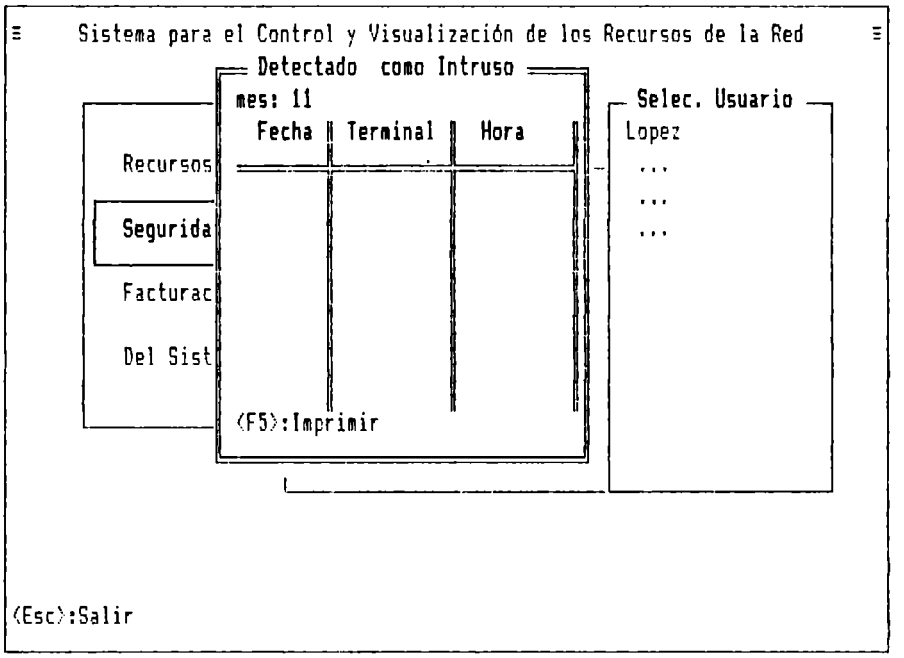


Módulo 2.3: Proporciona la fecha, terminal y hora en que un usuario en particular fue detectado como intruso, para un mes dado.

PANTALLA 2.3.3

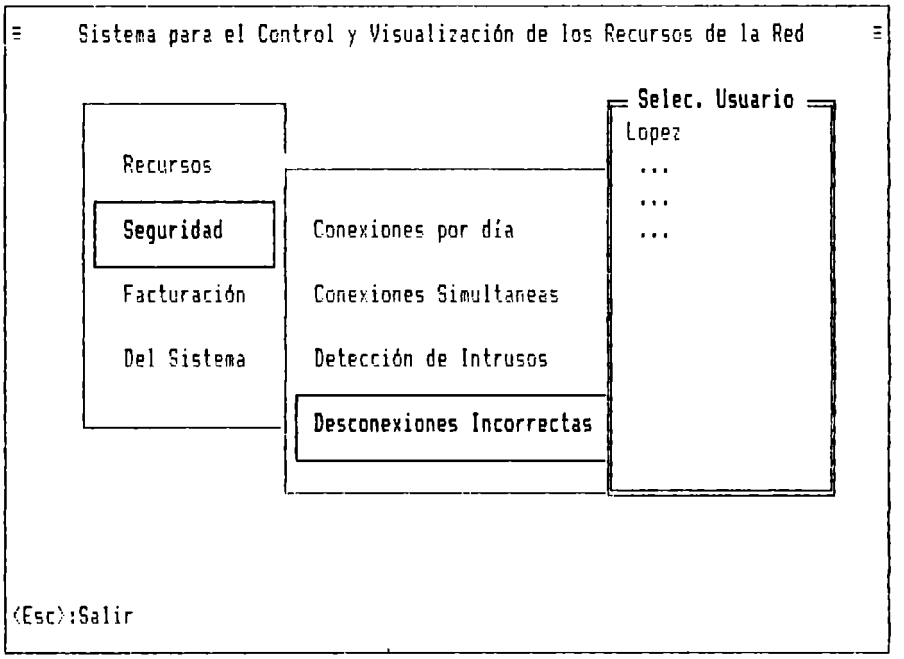


PANTALLA 2.3.4

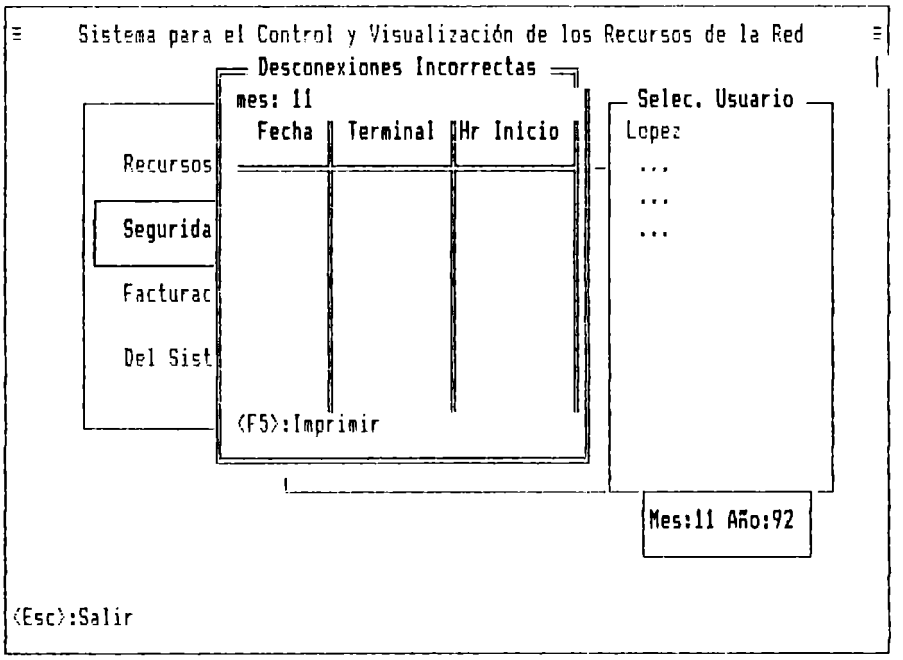


Modulo 2.4: Proporciona información de los usuarios que NO se desconectan explícitamente o en forma correcta; para un mes dado.

PANTALLA 2.4.3

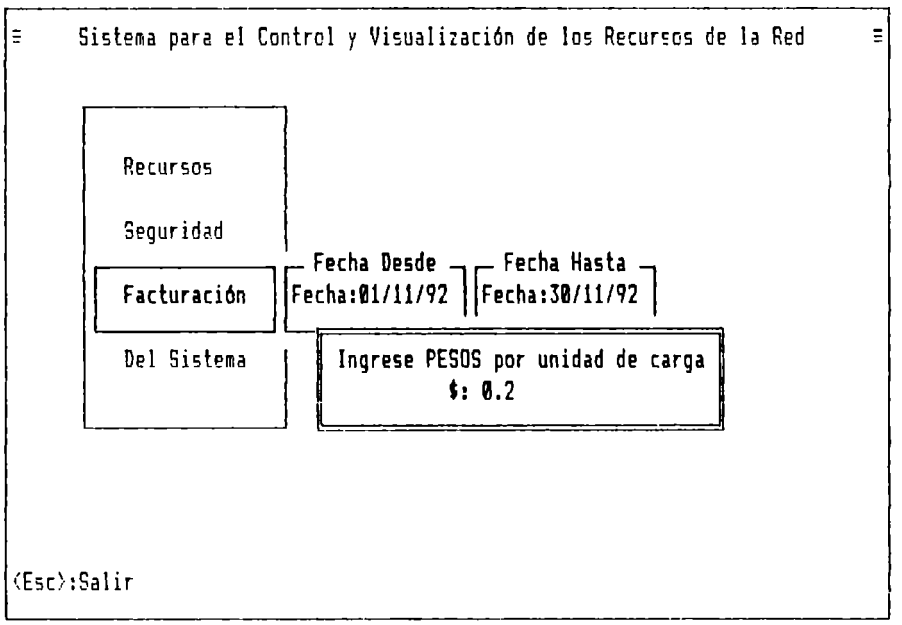


PANTALLA 2.4.4

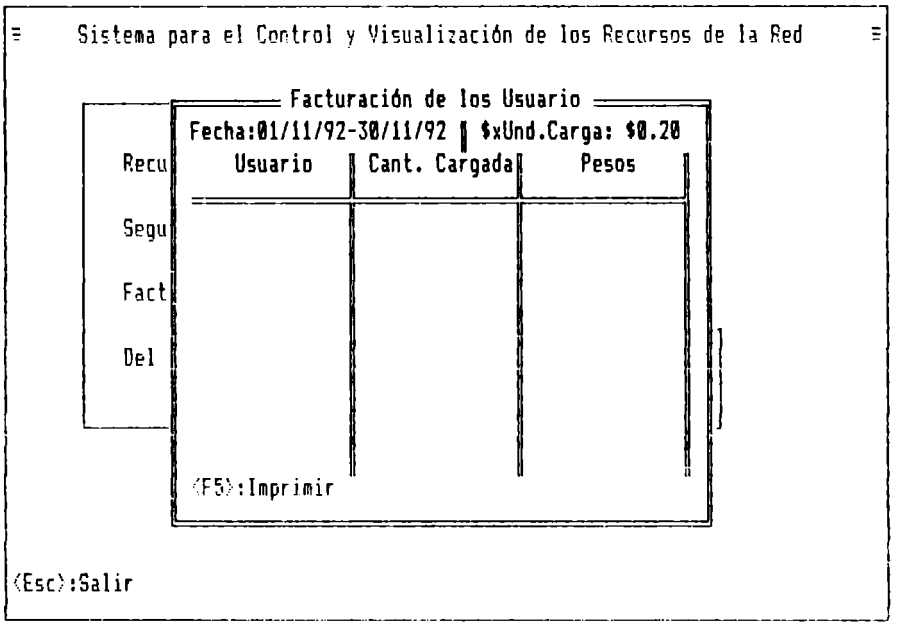


Módulo 3: ***FACTURACION.*** Proporciona la carga acumulada por cada usuario entre dos fechas y su costo facturado, según los pesos($) por unidad de carga a erogar.

PANTALLA 3.1.1



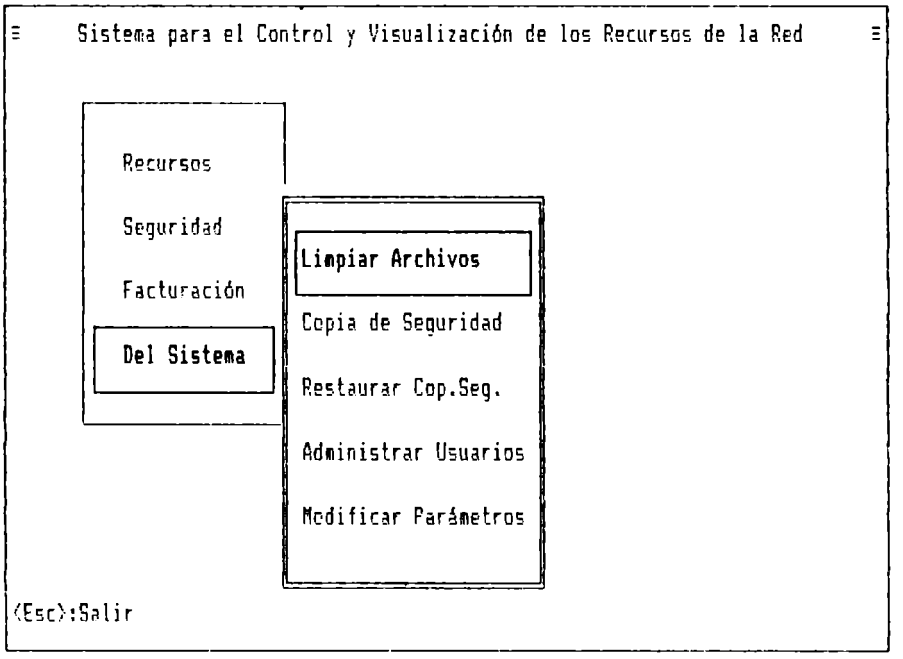
PANTALLA 3.1.2



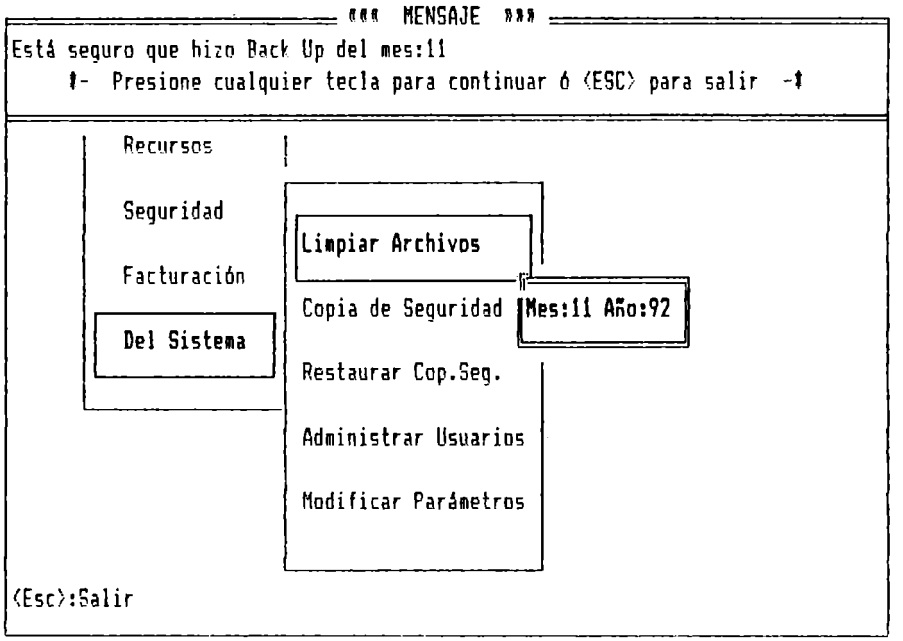
Modulo 4: DEL SISTEMA. En este módulo consta del mantenimiento de la información del sistema.

Modulo 4.1: Limpia información respecto de la actividad de los usuarios para un mes dado.

PLANILLA 4.1.1

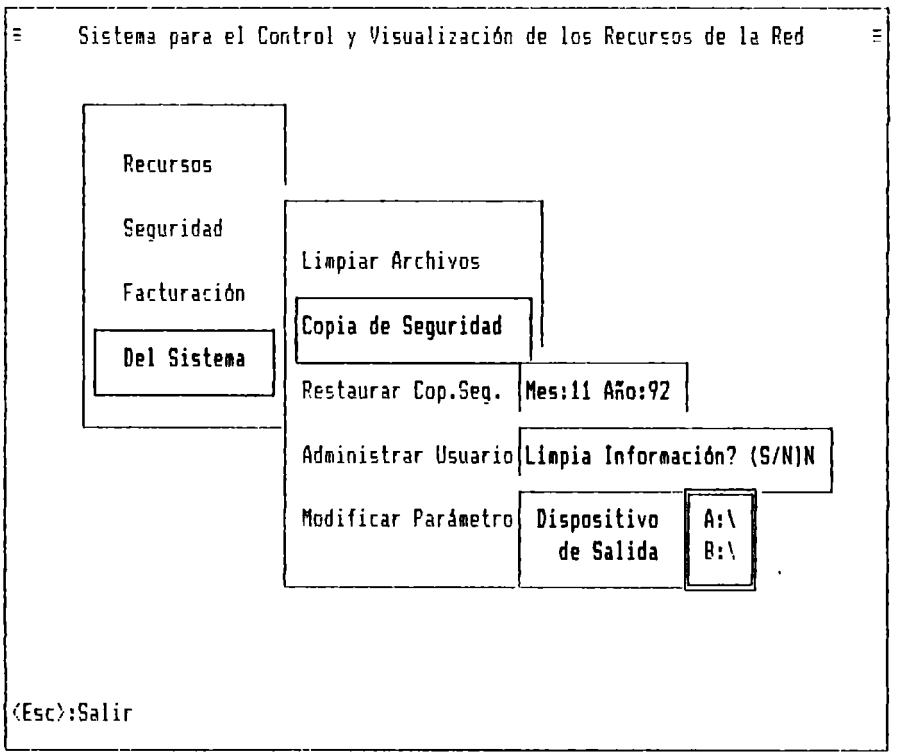


PANTALLA 4.1.2



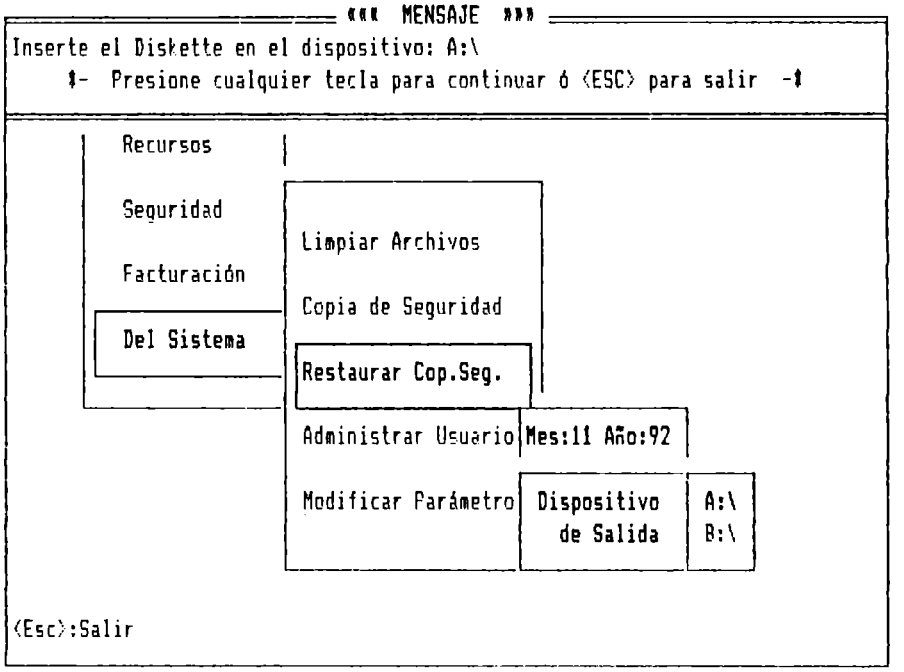
Modulo 4.2: Realiza las copias de seguridad en forma selectiva por mes de la información correspondiente al consumo de la red por parte de los usuarios; bajo al opción de borrar o no la información, así mantener optimizado el espacio en disco. Para los datos generales del sistema, la copia de la información se hace en forma total.

PANTALLA 4.2.1



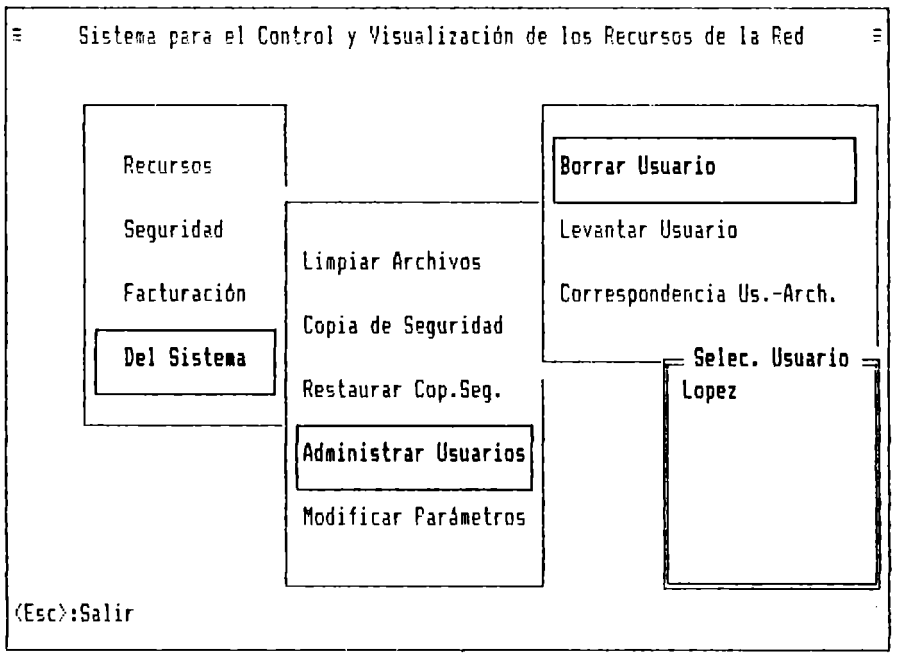
Modulo 4.3: Restaura la copia de seguridad en forma acorde a como se realizó la copia de seguridad.

PANTALLA 4.3.1

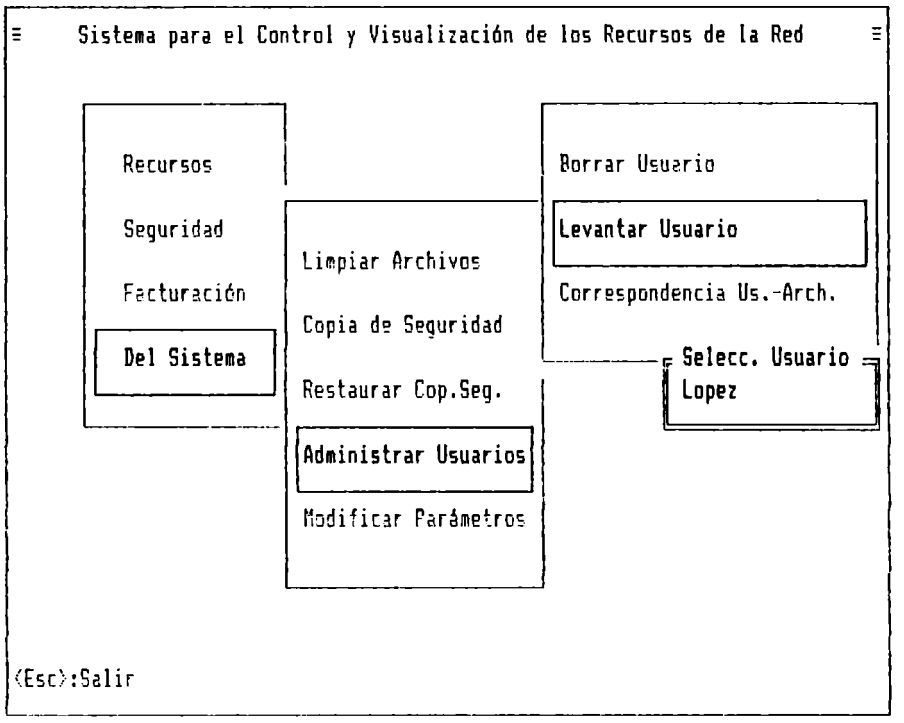


Modulo 4.4: Se realiza toda la parte de administración de los usuarios, permitiendo desactivarlo ó bajas lógicas, activarlo ó darle de alta, y visualizar que arch. le corresponde a cada usuario. Se esplaya en las siguientes 3 pantallas:

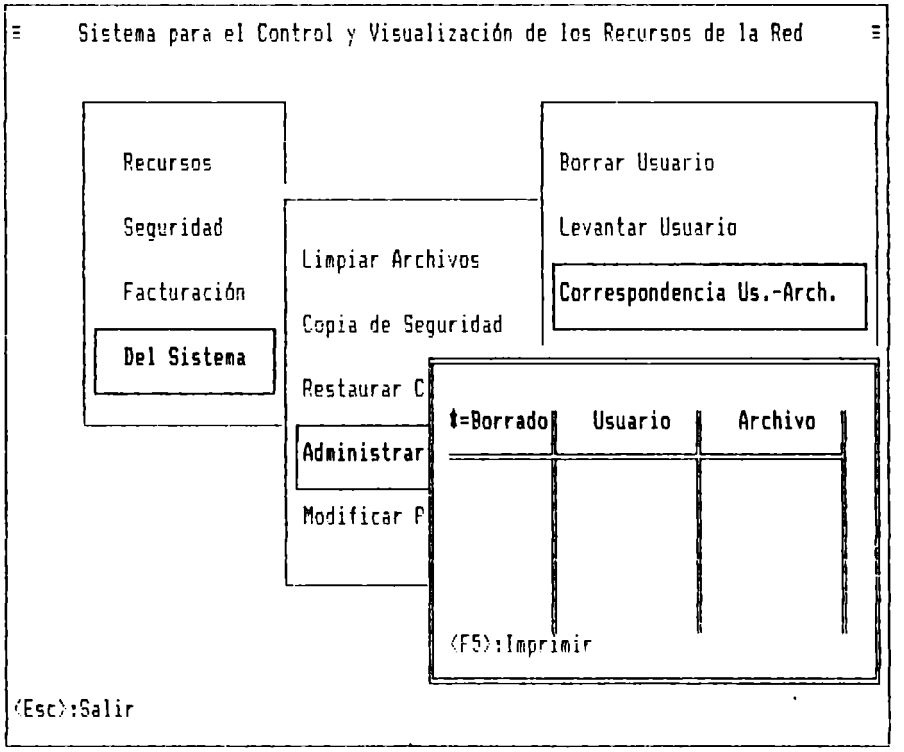
PANTALLA 4.4.1



PANTALLA 4.4.2



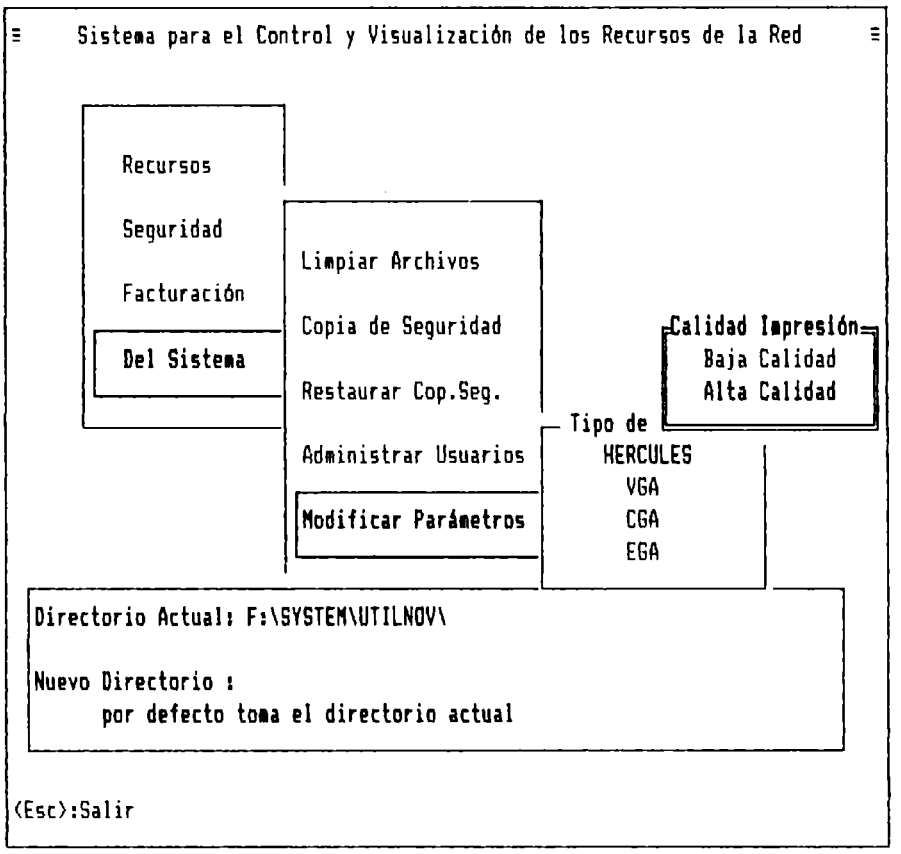
PANTALLA 4.4.3



Modulo 4.5: Permite la reinstalación del sistema corrigiendo sus parámetros: subdirectorio donde se encuentra instalado, placa de video y calidad de las impresiones gráficas; en la siguiente

pantalla:

PANTALLA 4.5.1



3 Analisis Detallado

PROCEDIMIENTO: *Generar Archivo Texto*

Definición :

Este proceso genera dos archivos de texto a través de los siguientes comandos de Novell:

"PAUDIT > net$acct.txt"

"ATOTAL > net$tot.txt"

Observación:

Estos comandos de Novell son incorporados en un archivo .BAT.

Datos Entrada:

- NET$ACDT.DAT; archivo generado por Novell, que contiene los distintos acontecimientos sucedidos en la red.

Datos Salida :

- NET$ACCT.TXT; archivo de texto el cual contiene los sucesos detallados de cada usuario, a medida que estos se van produciendo.
- NET$TOT.TXT; archivo de texto el cual contiene el total de los recursos consumidos por todos los usuarios a lo largo de un día.

Proceso:

PROCEDIMIENTO: *Generar Acct. Detallado por Usuario*

Definición :

Convierte un archivo de texto (NET$ACCT.TXT) procesándolo línea a línea en tres archivos de registros los cuales serán fuente de datos que se usarán para obtener información detallada de los usuarios; por otro lado se genera la tabla de usuarios y la tabla de terminales.

Observación:

1-El contenido del ACCOUNTING depende de como esté definido por el administrador de la red (Cantidad cargada, Tiempo de conexión, Requerimientos, Bloques leídos, Bloques escritos). Por defecto solo se registra la Fecha, Hora, Nombre del usuario, el mensaje de "Log-In" ó "Log-Out" y la dirección de la terminal. Se observo que mensajes como "Server booted" y "Server down" no figuran en el ACCOUNTING de las instalaciones con Novell V2.11.

Nota: basta con solo asignar alguna carga a algún recurso para que en el ACCOUNTING figure toda la información inherente al consumo de recursos ya descrita.

2-Conexiones Consecutivas. Un usuario puede conectarse (Log-In) más de una vez en la misma terminal sin que medie una desconexión (Log-Out). Se observó que en ciertas instalaciones al desconectarse (Log-Out) el usuario en dicha terminal, se registra los recursos consumidos por este respecto de la 2da. conexión(Log-In); siendo que en otras instalaciones genera la desconexión (Log-out) automáticamente.

Solución: antes de registrar una conexión verificar que no haya una conexión sin cerrar en dicha terminal. Si la hay, registrar una desconexión (Log-Out) asignándole los recursos consumidos en forma proporcional respecto al tiempo que dista entre ambas conexiones (Log-In).

3-No Registrarse Desconexiónes (Log-Out). Esto puede suceder por: un corte de luz; por olvido del usuario de desconectarse de la red al finalizar su sesión ó día de trabajo, entonces si se apagada del SERVER sin haber bajado la red no se registrará la desconexión (Log-Out) para dichos usuarios; igualmente si se apaga -quitar el suministro de corriente eléctrica- el SERVER habiendo usuarios conectados; etc.

Solución: Por un lado, generar una salida para el supervisor de modo tal que pueda consultar cuando se apagó el server sin haber bajado la red. Por otro lado, asignar al/los usuario/s una cantidad de recursos consumidos igual al valor medio del consumo por este/os.

Datos Entrada:
- NET$ACCT.TXT
- Sub-directorio donde está instalado el sistema.
- Term. Exist.; terminales existentes.
- Us. Exist.; usuarios existentes

Datos Salida :
- $ARCH_i$.USU
- ACT_DIA.REC
- ACT_MES
- Nueva Term; nuevas terminales incorporadas a la red.
- Us Nuevos; nuevos usuarios incorporados a la red.

Proceso:

```
<Abrir archivos: NET$ACCT.TXT, ARCHi.USU, ACT_DIA.REC y
  ACT_MES>
<Leer sub-directorio>
<Leer 1ra línea de un bloque de información del
  NET$ACCT.TXT. Retornar:fecha y hora>
   WHILE NOT EOF(NET$ACCT.TXT) DO
   BEGIN
     <Leer 2da línea de NET$ACCT.TXT>
     <Procesar 2da línea. Retorna:tipo_línea, carga y usuario>
     IF tipo_línea ='CHARGE'
       THEN
       BEGIN
         <Leer 3da línea de NET$ACCT.TXT>
         <Procesar 3da línea. Retorna: tpo_conexión,
         req_server, blq_leídos, dlq_escritos, blq_disco,
         es_log_out>
         IF es_log_out
           THEN
           BEGIN
             <Leer 4 líneas de NET$ACCT.TXT. Retorna:Obtener
              usuario, terminal>
             <Almacenar en ARCHi.USU i-esimo us. los datos de
              desconexión: carga, tpo_conexión, req_server,
            blq_leídos, blq_escritos y terminal.
```

```
              Correspondientes a la: fecha, hora, usuario>
            END
            ELSE
              <Almacenar el consumo de disco realizado en la
               última media hora: hora, carga, blq_disco.
               Correspondiente al: usuario y fecha>
        END
        ELSE  (note por log_in o intruso)
        BEGIN
          <Leer 3ª línea de NET$ACCT.TXT>
          <Procesar 3ª línea. Retorna: tipo_inf y terminal>
          CASE tipo_inf OF
            <Almacenar Log-In ó Intruso correspondiente a la:
             fecha, hora, usuario, terminal, log-In ó intruso>
            <Almacenar Log-Out en ARCHi.USU i-esimo us. con los
             datos de desconexión: carga=0, tpo_conexión=0,
           req_server=0, blq_leídos=0, blq_escritos=0 y
           terminal. Correspondientes a la: fecha, hora,
           usuario>
            <Cambio de día, inicializando todos los ARCHi.USU>
          END
        END
      <Leer 1ra línea de un bloque de información del
       NET$ACCT.TXT. Retornar:fecha y hora>
    END;
    <Cerrar archivos: NET$ACCT.TXT, ARCHi.USU, ACT_DIA.REC y
     ACT_MES>>
END;
```

PROCEDIMIENTO: *Generara Acct. Total del Sistema*

Definición :

Convierte un archivo de texto (NET$TOT.TXT) procesandolo línea a línea en obteniendo un archivos de registros (TOTAL.REC) el cual será la fuente de datos que se usará para obtener información del total de recursos consumidos por los usuarios a lo largo del día.

Observación:

El NET$TOT.TXT también proporciona un total semanal de todos los recursos consumidos por todos los usuarios, el cual no será tenido en cuenta.

Datos Entrada:

- NET$TOT.TXT
- Sub-directorio donde está instalado el sistema.

Datos Salida :

- TOTAL.REC

Proceso:

```
BEGIN
  <Abrir archivos: NET$TOT.TXT y TOTAL.REC>
     <Leer 1ra línea de un bloque de información del
      NET$TOT.TXT. Retornar:fecha>
  WHILE (NOT EOF(NET$TOT.TXT)) DO
```

```
    BEGIN
      <Leer 2da línea de NET$TOT.TXT. Retornar: tpo_conec y
       req_server>
        <Leer 3ra línea de NET$TOT.TXT. Retornar: blq_leido,
         blq_escrito>
        <Leer 4ta línea de NET$TOT.TXT. Retornar: blq_disco>
        <Grabar: fecha, tpo_conec, req_server, blq_leido,
         blq_escrito y blq_disco en TOTAL.REC>
        <Leer 1ra línea de un bloque de información del
         NET$TOT.TXT. Retornar:fecha>
      END
    <Cerrar: NET$TOT.TXT y TOTAL.REC>
  END
```

PROCEDIMIENTO: *Dar Inf. Gral. de la Red Día a Día*

Definición :

Obtiene el total de recursos consumidos por todos los usuarios de la red día a día en un mes dado.

Observación:

Datos Entrada:

- mes y año
- TOTAL.REC

Datos Salida :

- mes y año
- día
- tpo. consumido
- req. server
- blq. leídos
- blq. escritos
- blq. disco

Proceso:

```
  BEGIN
      <Armar una matriz donde las filas son los días del mes y
       cinco columnas, una por cada uno de los recursos
       consumidos por la totalidad de los usuarios>
      WHILE NOT(fin_consulta) DO
      BEGIN
        <Mostrar matriz en pantalla. Retornar: un
         carater_retorno>
        IF carater_retornoret = inprimir
          THEN <Imprimir Matriz>
        IF carater_retornoret = graficar
          THEN <Graficar dos columnas cualesquiera de la
                matriz>
      END
  END
```

PROCEDIMIENTO: *Dar Inf. Total de Rec. x Us. al Mes*

Definición :

Lee de la tabla de usuarios a aquellos que están activos y busca el total de recursos consumidos por éste y la cantidad de veces que el sistema lo reconoce como intruso en el mes y año indicado.

Observación:

Datos Entrada:

- mes y año
- USUARIOS.TAB
- ACT_MES.REC

Datos Salida :

- mes y año;
- usuario
- tpo. consumido
- req. server
- blq. leídos
- blq. escritos
- conexiones como intruso

Proceso:

```
BEGIN
     <Armar matriz donde las filas son los usuarios más una
      última fila que es el total y en las cinco columnas sus
      correspondientes recursos consumidos más las conexiones
     incorrectas>
     WHILE (NOT(fin_consulta)) DO
     BEGIN
       <Mostrar matriz en pantalla. Retornar: un
        carater_retorno>
       IF carater_retornoret = inprimir
         THEN <Imprimir Matriz>
       IF carater_retornoret = graficar
         THEN <Graficar dos columnas cualesquiera de la
               matriz>
     END
END
```

PROCEDIMIENTO: *Dar Inf. Cons. Disco x Us. para un Día*

Definición :

Mostrar un tabla con la carga consumida cada media horas desde la conexión y hasta la desconexión del usuario seleccionado en una fecha dada.

Observación:

Datos Entrada:

- usuario
- fecha
- ACT_DIA.REC

Datos Salida :

- fecha
- carga total del día

```
   - hora
   - carga para dicha hora
Proceso:
   BEGIN
        <Abrir archivo ACT_DIA.REC>
     <Buscar registro en ACT_DIA.REC que corresponda al usuario
       en la fecha deseada>
        IF existe_inf
          THEN
          BEGIN
            <Generar matiz donde las filas corresponde una cada
             media hora y la columna es la correspondiente carga>
            WHILE (NOT (fin_consulta)) DO
            BEGIN
              <Mostrar matriz en pantalla. Retornar: un
               carater_retorno>
              IF carater_retornoret = inprimir
                THEN <Imprimir Matriz>
              IF carater_retornoret = graficar
                THEN <Graficar dos columnas cualesquiera de la
                      matriz>
            END
          END
        <Cerrar ACT_DIA.REC>
      END
```

PROCEDIMIENTO: *Dar Inf. Recur. Consumidos xUs. xMes xTerminal*

Definición :

Proporciona los recursos consumidos de un usuario seleccionado día a día discriminados por terminal, para un mes dado.

Observación:

Datos Entrada:

- usuario
- mes y año
- $ARCH_i$.USU; correspondiente al usuario seleccionado.

Datos Salida :

- usuario
- mes
- día
- req. server
- tpo. consumido
- blq. leídos
- blq. escritos

```
Proceso:
   BEGIN
        <Abrir ARCHi.USU>
        <Leer de ARCHi.USU hasta encontrar inf. del mes deseado>
        IF hay_inf.
```

```
          THEN
          BEGIN
            <Pasar inf. del mes a un arch. auxiliar>
            <Obtener la cantidad de terminales en que se conectó
              el usuario>
            FOR para_cada_terminal DO
            BEGIN
              <Generar matriz donde las filas son los días y las
               columnas los recursos en dicha terminal>
              WHILE (NOT (fin_consulta)) DO
              BEGIN
                <Mostrar matriz en pantalla. Retornar: un
                 carater_retorno>
                IF carater_retornoret = inprimir
                  THEN <Imprimir Matriz>
                IF carater_retornoret = graficar
                  THEN <Graficar dos columnas cualesquiera de la
                        matriz>
              END
            END
          END
        <Cerrar ARCHi.USU>>
    END
```

PROCEDIMIENTO: *Dar Inf. de Conex. x Día de un Us.*

Definición :

Mostrar una tabla con las conexiones realizadas por un usuario en un mes dado.

Observación:

Este proceso tiene una salida que permite visualizar en forma gráfica las conexiones simultaneas de un usuario en distintas terminales para un día determinado.

Datos Entrada:

- usuario selecc.
- mes y año
- terminal
- hr. conexión
- hr. desconexión

Datos Salida :

- usuario
- fecha
- terminal
- hr. conexión
- hr. desconexión

Proceso:

```
    BEGIN
        <Abrir el archivo ARCHi.USU del correspondiente usuario>
        <Buscar registros que cumplan con mes>
        IF  no hay registro para ese mes
          THEN <enviar mensaje de que no hay registros para ese
                mes>
```

```
        ELSE
       <Mostrar tabla de conexiones para ese mes>
      <Cerrar archivo ARCHi.USU>
    END
```

PROCEDIMIENTO: *Determinar Desconexiones Incorrectas*

Definición :

Mostrar una tabla con todas las veces que un usuario se halla desconectado incorrectamente para un mes dado.

Observación:

La hora de desconexión está marcada como 99:99:99

Datos Entrada:

- usuario selecc.
- mes y año
- fecha
- termimnal
- hr. conexión
- hr. desconexión

Datos Salida :

- usuario
- fecha
- termnal
- hr. conex.

Proceso:

```
BEGIN
     <Abrir archivo ARCHi.USU del correspondiente usuario>;
     <Buscar registros que cumplan con mes y hayan sido
      detectado como desconectado incorrectamente>
     IF  no hay registro
       THEN <enviar mensaje de que no hay registros para ese
             mes>
       ELSE <Mostrar tabla de Desconexiones Incorrectas>
  <Cerrar archivo ARCHi.USU>
  END;
```

PROCEDIMIENTO: *Determinar Usuarios Detectados como Intrusos*

Definición :

Mostrar una tabla con todos los usuarios detectados como intrusos para un mes dado.

Observación:

Datos Entrada:

- usuario selec.
- mes y año
- terminal
- hr. conexión
- intruso

Datos Salida :

- mes y año

```
   - usuario
   - terminal
   - hr. conexión
   - intruso
Proceso:
   BEGIN
        <Abrir archivo ARCHi.USU del correspondiente usuario>
        <Buscar registros que cumplan con mes y haya sido
         detectado como intruso>;
        IF  No hay registro
          THEN <enviar mensaje de que no hay registros para ese
                mes>
          ELSE <Mostrar tabla de Detecciones de intruso>
        <Cerrar archivo ARCHi.USU>;
     END;
```

PROCEDIMIENTO: *Dar Inf. Conex. Simultaneas x Us.*

```
Definición :
      Muestra una tabla por cada usuario con cada una de las
   conexiones simultaneas en distintas terminales realizadas por
   estos.
Observación:
Datos Entrada:
   - fecha desde
   - fecha hasta
   - usuarios
   - terminal
   - hr. conexión
   - hr. desconexión
Datos Salida :
   - usuario
   - fecha
   - terminal
   - hr. conexión
   - hr. desconexión
Proceso:
   BEGIN
        <Abrir archivos USUARIOS.TAB y ARCHi.USU>
        WHILE (haya usuarios en USUARIOS.TAB) DO
        BEGIN
          <Leer usuario de USUARIOS.TAB>
          <Armar matriz de conex. simultaneas para el usuario
           dado; donde cada fila es una conexión simultanea con
           otra y columnas para la terminal y las hr. de conex.
           y desc. respectivamente>
          <Cerrar archivo ARCHi.USU>
          <Mostrar matriz>
          IF imprime_matriz
            THEN <Imprimir matriz>
        END
   END
```

PROCEDIMIENTO: *Facturar Usuarios*

Definición :
Para un período de dos fechas, lee los pesos($) que se desea cobrar por cada unidad de carga.
Observación:
Datos Entrada:
- usuarios
- carga
- fecha desde
- fecha hasta
- pesos($)

Datos Salida :
- usuarios
- carga
- pesos($) facturados al us.

Proceso:

```
BEGIN
      <Abrir archivos USUARIOS.TAB y ARCH_i.USU>
      WHILE (haya usuarios en USUARIOS.TAB) DO
      BEGIN
        <Ubicar datos del us. ARCH_i.USU correspondientes a la
         fecha>
        <Mientras haya datos correspondan a la fecha acumular
         en una fila de la matriz>
        <Multiplicar carga_acumulada x pesos($)>
      END
      <Mostrar matriz en pantalla>
END
```

PROCEDIMIENTO: *Borrar Información*

Definición :
Limpia información correspondiente al mes seleccionado por el supervisor.
Observación:
Datos Entrada:
- mes
- registros de cada uno de los archivos

Datos Salida :
- registros de cada uno de los archivos

Proceso:

```
BEGIN
      <Limpiar inf. de los arch. ARCH_i.USU      *-LimpAcctUs-*>
      <Limpiar inf. del arch. ACT_DIA.REC        *-LimpAcctDia-*>
END
```

PROCEDIMIENTO: *LimpAcctUs.*

Definición :
Limpiar inf. de los arch. $ARCH_i$.USU correspondientes al mes pasado por parámetro.
Observación:
Datos Entrada:
- mes: mes a limpiar.
- registro del archivo $ARCH_i$.USU

Datos Salida :
- cont_proc: si continua el proceso.

Proceso:

```
BEGIN
     <Leer un usuario de USUARIOS.TAB>
     WHILE (haya usuario) AND (continuar)DO
     BEGIN
       <Abrir arch. ARCHi.USU correspondiente al us. leído>
       <Abrir arch. con que contendrá inf. depurada o
        resultante ACCTCOMP.USU>
       <Grabar reg. cero de ARCHi.USU en ACCTCOMP.USU>
       <Grabar inf correspondiente a meses anteriores al
        seleccionado de ARCHi.USU a ACCTCOMP.USU>
       IF (hay inf. a borrar en ARCHi.USU)
         THEN
         BEGIN
           <Saltear inf. a borrar de ARCHi.USU>
           IF (no queda inf. de meses posteriores al selec.)
          THEN
             ELSE <Pasar el resto de la inf. ARCHi.USU a
                   ACCTCOMP.USU>
           <Cerrar ACCTCOMP.USU y ARCHi.USU>
           <Borrar ARCHi.USU>
           <Convertir ACCTCOMP.USU en ARCHi.USU
                                          *-ActualizarAcctUs-*>
         END
         ELSE
         BEGIN
           <Cerrar ACCTCOMP.USU y ARCHi.USU>
           <Error>
         END
     END
END
```

PROCEDIMIENTO: *ActualizarAcctUs*

Definición:
Convertir ACCTCOMP.USU actualizado en $ARCH_i$.USU; actualizando el puntero de inicio de día del reg. cero.
Observaciones:
Datos Entrada:
- acct_comp: referencia al arch. ACCTCOMP.USU.

```
   - nom_arch: nombre del arch. con que se renombrará al
     ACCTCOMP.USU.
   - rec_cop: número de registros del arch. ACCTCOMP.USU.
Datos Salida:
Proceso:
   BEGIN
     <Abrir el arch. ACCTCOMP.USU>
     IF (puntero de inicio del día <> Ø)
       THEN <Actualiza reg. cero>
     <Renombrar ACCTCOMP.USU por nom_arch>
     <Cerrar arch.>
   END
```

PROCEDIMIENTO: *LimpAcctDia*

```
Definición:
      Limpiar inf. del arch. ACT_DIA.REC correspondientes al  mes
   pasado por parámetro.
Observación:
Datos Entrada:
   - dir: directorio donde esta instalado el sistema.
   - mes: mes a limpiar.
Datos Salida:
Proceso:
   BEGIN
     <Abrir el arch. ACT_DIA.REC>
     <Abrir arch. con  que contendrá inf. depurada  o resultante
      ACCTCOMP.USU>
     <Grabar inf correspondiente a meses  anteriores al
      seleccionado de ACT_DIA.REC en ACCTCOMP.USU>
     IF (hay inf. a borrar en ACT_DIA.REC)
       THEN
       BEGIN
         <Saltear inf. a borrar de ACT_DIA.REC>
         IF (no queda inf. de meses posteriores al selec.)
           THEN
           ELSE
           BEGIN
             <Grabar inf. correspondiente a meses posteriores
               al seleccionado de ACT_DIA.REC en ACCTCOMP.USU>
             <Cerrar ACCTCOMP y ACT_DIA.REC>
             <Borrar ACT_DIA.REC>
             <Convertir ACCTCOMP.USU en ACT_DIA.REC
                         *-ActualizarAcctDia(acct_comp,rec_cop-*>
          END
       ELSE
       BEGIN
         <Cerrar ACCTCOMP y ACT_DIA.REC>
         <Error>
       END
   END
```

PROCEDIMIENTO: *ActualizarAcctDia*

Definición:
Convertir ACCTCOMP.USU actualizado en ACT_DIA.REC actualizando el puntero de inicio de día del reg. cero.
Observación:
Datos Entrada:
- acct_comp: referencia al arch. ACCTCOMP.USU.
- rec_cop: número de registros del arch. ACCTCOMP.USU.

Datos Salida:
Proceso:

```
BEGIN
  <Abrir el arch. ACCTCOMP.USU>
  IF (puntero de inicio del día <> Ø)
    THEN <Actualizar reg. cero>
  <Renombrar el arch. ACCTCOMP.USU por ACT_DIA.REC>
  <Cerrar arch.>
END
```

PROCEDIMIENTO: *Realizar Copia de Seguridad*

Definición:
Realiza el Back-Up en forma selectiva por mes para los arch. ARCHi.USU y ACT_DIA.REC; bajo las opciones de limpiar o eliminar la inf. bajada o no. Optimizando dichos archs.
Observación:
El resto de los arch. realiza una copia de seguridad de la totalidad de la información.
Datos Entrada:
- mes y año
- registros de los archivos

Datos Salida:
- registros copiados

Proceso:

```
BEGIN {ProcesarBackUp}
  <Leer si desea eliminar inf. bajada      *-LeerTipoBackUp-*>
  IF (mes escogido  <> Ø) AND (no es mes corriente) AND
      (continua el proceso)
    THEN
    BEGIN
      <Leer drive donde se bajará la inf.       *-LeerDrive-*>
      WHILE (mes no actual) AND (continue proceso) DO
      BEGIN
        IF (cont. proc.)
          THEN <Bajar los arch. ARCHi.USU     *-BajarAcctUs-*>
        IF (cont. proc.)
          THEN <Bajar el rch. ACT_DIA.REC     *-BajarAcctDia-*>
        <Incrementa el mes>
        IF (no es mes actual)
       THEN <Continua co meses subsiguientes?>
      END
```

```
        <Continua con el resto de los arch.?>
        IF (cont. proc.)
          THEN <Bajar arch. ACT_MES.REC          *-BajarAcctMes-*>
        IF (cont. proc.)
          THEN <Bajar arch. TOTAL.REC              *-BajarTotal-*>
        IF (cont. proc.)
          THEN <Bajar arch. USUARIOS.TAB           *-BajarTabUs-*>
        IF (cont. proc.)
          THEN <Bajar arch. TERMINAL.TAB         *-BajarTabTerm-*>
        IF (cont. proc.)
          THEN <Bajar arch. PARAM.DAT              *-BajarParam-*>
    END
  END
```

PROCEDIMIENTO: *BajarAcctUs*

Definición:

Realiza el Back-Up del arch. ARCHi.USU bajo las opciones de limpiar o eliminar la inf. bajada o no. Optimizando el arch.

Observación:

El resto de los arch. realiza una copia de seguridad de la totalidad de la información.

Datos Entrada:

- drive: dirección del dispositivo a realizar el Back_Up.
- nro_drive: número del dispositivo.
- dir: directorio donde está instalado el sistema.
- mes: mes correspondiente a bajar información.
- borrar_inf: si el Back-Up es optimizando el arch.

Datos Salida:

- cont_proc: si continua el proceso.

Proceso:

```
  BEGIN
    <Abrir arch. tabla de usuarios USUARIOS.TAB>
    WHILE (haya usuarios) DO
    BEGIN
      <Leer usuario>
      <Abrir arch. ARCHi.USU>
      IF (modo optimizar arch.)
        THEN
        BEGIN
          <Abrir arch. con que contendrá inf. compactada o
          optimizada ACCTCOMP.USU>
          <Grabar reg. cero>
        END;
      fin:=FALSE;
      { ubicar inf. del mes interesado }
      WHILE (sea dist. de EOF ARCHi.USU) AND (no finalice) DO
      BEGIN
        <Leer de ARCHi.USU>
        IF (mes de la inf. leída es dist. al mes de bach-up)
          THEN <Finaliza>
          ELSE IF (modo optimizar arch.)
```

```
                THEN <Grabar reg. leido en ACCTCOMP.USU>
      END;
      IF (finalizó hallando inf. para hacer back-up)
        THEN
        BEGIN
          <Abrir arch. que contendrá el back-up; ACCTAUX.USU>
          <Grabar reg. cero>
          WHILE (se dist. EOF ARCHi.USU) AND
                (mes de inf. leída igual a la de back-up) DO
            <Grabar inf. del ARCHi.USU al ACCTAUX.USU>
          IF (EOF del ARCHi.USU)
            THEN
              <Grabo último reg. de ARCHi.USU al ACCTAUX.USU>
            ELSE
              IF (modo optimizar arch.)
                THEN
              <Termino de pasar la inf de ARCHi.USU a
              ACCTCOMP.USU para meses subsiguientes>
              <Verificar  la  existencia  de  espacio  en  el
              diskette>
          IF (cont. proc.)
            THEN
            BEGIN
              <Bajar inf. del ACCTAUX.USU a diskett*-BackUpUs-*>
              IF (modo optimizar arch.)
                THEN
                  <Actualizar ARCHi.USU con la inf. optimizada
                    de ACCTCOMP.USU        *-ActualizarAcctUs-*>
            END
        END
        ELSE
          <Error>
    END
  END
```

PROCEDIMIENTO: *BackUpUs*

Definición:
Realiza la copia física de la inf. en disco a diskette.
Observación:
Datos Entrada:
- drive : dirección donde se encuentra el diskette.
- dir: directorio del sistema.
- nom_arch: nombre del arch. ARCHi.USU.
- mes: mes del back-up.

Datos Salida:
Proceso:

```
  BEGIN
    <Genero nombre del arch. con que residirá en diskette
                                                *-NomArchDisk-*>
    <Genero arch. en diskette>
```

```
    WHILE (sea dist. EOF ACCTAUX.USU) DO
    BEGIN
      <Leo del ACCTAUX.USU>
      <Grabo en arch. de diskette>
    END;
    <Cerrar ambos arch.>
  END;
```

PROCEDIMIENTO: *BajarAcctDia*

Definición:
Realiza el Back-Up de un mes específico del arch. ACT_DIA.REC bajo las opciones de limpiar o eliminar la inf. bajada o no.

Observación:

Datos Entrada:
- drive: dirección del dispositivo a realizar el Back_Up.
- nro_drive: número del dispositivo.
- dir: directorio donde está instalado el sistema.
- mes: mes correspondiente a bajar información.
- borrar_inf: si el Back-Up es optimizando el arch.

Datos Salida:
- cont_proc: si continua el proceso.

Proceso:

```
  BEGIN
    <Es simétrico al proceso: BajarAcctUs>
  END
```

PROCEDIMIENTO: *BajarAcctMes*

Definición:
Realiza el Back-Up del arch. ACT_MES.REC a diskette.

Observación:

Datos Entrada:
- drive: dirección del dispositivo a realizar el Back_Up.
- nro_drive: número del dispositivo.
- dir: directorio donde está instalado el sistema.
- mes: mes correspondiente a bajar información.
- borrar_inf: si el Back-Up es optimizando el arch.

Datos Salida:
- cont_proc: si continua el proceso.

Proceso:

```
  BEGIN
    <Abrir arch. ACT_MES.REC>
    <Verificar la existencia de diskette y espacio libre>
    IF (cont. proc.)
      THEN <Hacer la copia en diskette          *-BackUpMes-*>
    <Cerrar arch. ACT_MES.REC>
  END
```

PROCEDIMIENTO: *BackUpMes*

Definición:
Realiza la copia en diskette del arch. ACT_MES.REC
Observación:
Datos Entrada:
- drive: dirección del dispositivo seleccionado.
Datos Salida:
Proceso:

```
BEGIN
  <Generar nombre del arch. con que se grabará en diskette>
  WHILE (no sea EOF ACT_MES) DO
  BEGIN
    <Leer reg. de ACT_MES.REC>
    <Grabar en reg. en diskette>
  END
  <Cerrar arch. del diskette>
END
```

PROCEDIMIENTO: *BajarTotal*

Definición:
Realiza el Back-Up del arch. TOTAL.REC a diskette.
Observación:
Datos Entrada:
- drive: dirección del dispositivo a realizar el Back_Up.
- nro_drive: número del dispositivo.
- dir: directorio donde está instalado el sistema.

Datos Salida:
- cont_proc: si continua el proceso.

Proceso:

```
BEGIN
  <Es simétrico a BackUpDia>
END
```

PROCEDIMIENTO: *BajarTabUs*

Definición:
Realiza el Back-Up del arch. USUARIOS.TAB a diskette.
Observación:
Datos Entrada:
- drive: dirección del dispositivo a realizar el Back_Up.
- nro_drive: número del dispositivo.
- dir: directorio donde está instalado el sistema.

Datos Salida:
- cont_proc: si continua el proceso.

Proceso:

```
BEGIN
  <Es simétrico a BackUpDia>
END
```

PROCEDIMIENTO: *BajarTabTerm*

Definición:
Realiza el Back-Up del arch. TERMINAL.TAB a diskette.
Observación:
Datos Entrada:
- drive: dirección del dispositivo a realizar el Back_Up.
- nro_drive: número del dispositivo.
- dir: directorio donde está instalado el sistema.

Datos Salida:
- cont_proc: si continua el proceso.

Proceso:
```
BEGIN
  <Es simétrico a BackUpDia>
END
```

PROCEDIMIENTO: *BajarParam*

Definición:
Realiza el Back-Up del arch. PARAM.DAT a diskette.
Observación:
Datos Entrada:
- drive: dirección del dispositivo a realizar el Back_Up.
- nro_drive: número del dispositivo.
- dir: directorio donde está instalado el sistema.

Datos Salida:
- cont_proc: si continua el proceso.

Proceso:
```
BEGIN
  <Es simétrico a BackUpDia>
END
```

PROCEDIMIENTO: *Restaurar Copia de Seguridad*

Definición:
Restaura la inf. almacenada en Back-Up.
Observación:
Datos Entrada:
- mes y año
- registros de archivos con copia de seguridad

Datos Salida:
- registros restaurados

Proceso:
```
BEGIN {LevantarBackUp}
     <Leer dispositivo de donde se restarará     *-LeerDrive-*>
     IF (cont. proc.)
       THEN <Restaurar ARCHi.USU              *-LevantarAcctUs-*>
     IF (cont. proc.)
       THEN <Reastaurar ACT_DIA.REC          *-LevantarAcctDia-*>
```

```
        IF ExisArchInfGral(dir)
          THEN
          BEGIN
            IF (cont. proc.) THEN <Restaurar ACctMes-*>
            IF (cont. proc.) THEN <Restaurar TOTALarTotal-*>
            IF (cont. proc.) THEN <Restaurar USUARIOS.TAB-*>
            IF (cont. proc.) THEN <Restaurar TERMINAL.TAB-*>
            IF (cont. proc.) THEN <Restaurar PARAMETROS.TAB-*>
          END
      END
```

PROCEDIMIENTO: *LevantarAcctUs*

Definición:
Restaura la inf. de los arch. ACRHi.USU

Observación:
- drive: dirección del dispositivo de donde se restaura.
- dir: directorio del sistema.
- m_bak: mes que se resturará.

Datos Entrada:
- cont_proc: si continúa el proceso.

Datos Salida:

Proceso:

```
BEGIN
  { levantar los accounting de cada usuario }
  <Abrir arch.de usuarios de USUARIOS.TAB>
  WHILE (haya usuarios) AND (cont. proc.) DO
  BEGIN
    <Leer usuario>
    <Abrir ARCHi.USU>
    arch_dk:=NomArchDisk(r_tab.nom_arch,m_bak)
   IF (existe el arch. a restaurar en diskette) AND
      (exite datos para restaurar)
      THEN IF (no existen datos en al ARCHi.USU)
             THEN <Restaura inf. de diskette
                                          *-LevParcialAcctUs-*>
             ELSE <Error>
      ELSE <Error>
  END
END
```

PROCEDIMIENTO: *LevParcialAcctUs*

Definición:
Restaura la inf. del arch. en diskette de modo tal que en el ARCHi.USU se mantenga ordenado por fecha.
Observación:
Datos Entrada:
- drive: dirección del dispositivo de donde se restaurará.
- dir: directorio del sistema.
- arch_dk: nombre del arch. en diskette.
- arch_dc: nombre del arch. en disco.

Datos Salida:
Proceso:

```
BEGIN
  <Abrir arch. en diskete y en disco>
  <abrir arch. auxiliar ACCTAUX.USU>
  <Salteo la inf. del disco que corresponda a meses menores>
  WHILE (haya inf.arch.disco) AND (corresp.meses menores) DO
    <Salteo inf. del disco>
  WHILE (haya inf en diskette) DO
    <Leer reg. del diskette y grabo en disco>
  <Pasar el resto de la inf que haya quedado en el disco>
  <Cerrar archs.>
END
```

PROCEDIMIENTO: *Modificar Parámetros del Sistema*

Definición:
Lee los parámetros actuales con que se instaló el sistema y permite su modificación.
Observación:
Por defecto asume el directorio corriente y una placa HERCULES o VGA que tienen idénticas características de memoria
Datos Entrada:
- parámetros existentes
- parámetros nuevos

Datos Salida:
- parámetros nuevos

Proceso:

```
BEGIN
  <Abrir archivo PARAM.DAT>
  <Muestra el directorio actual>
  REPEAT
    <Lee nuevo directorio>
     UNTIL (el nuevo directorio sea correcto)
  <selecciona placa de video>
  <asigno diecc. de memoria y tamaño de la misma según la
    placa>
  <Grabo nuevos parámetros en PARAM.DAT>
  <Cierro PARAM.DAT>
END
```

PROCEDIMIENTO: *Borrar Usuarios*

Definición:

Muestra la lista de usuarios activos del archivo USUARIOS.TAB, permitiendo la selección de alguno de estos y marcarlo como borrado.

Observación:

La bajas son lógicas.

Datos Entrada:

- usuarios existentes
- usuario a borrar

Datos Salida:

- usuario borrado

Proceso:

```
BEGIN
  <Abrir archivo USUARIOS.TAB>
  <Carga tabla con usuarios activos>
  <Selecciona usuario a desactivar>
  <Marca y graba al usuario como borrado o inactivo>
  <Cierra archivo USUARIOS.TAB>
END
```

PROCEDIMIENTO: *Levantar Usuarios Borrados*

Definición:

Muestra la lista de usuarios inactivos del archivo USUARIOS.TAB, permitiendo la selección de alguno de estos eliminado la marca de borrado.

Observación:

- usuarios borrados o inactivos
- usuario a levantar

Datos Salida:

- usuario levantado

Proceso:

```
BEGIN
  <Abrir archivo USUARIOS.TAB>
  <Carga tabla con usuarios inactivos>
  <Selecciona usuario a levantar>
  <Desmarca y graba al usuario como borrado o inactivo>
  <Cierra archivo USUARIOS.TAB>
END
```

UNIDAD: *Def*

Definición:
Conrtiene la definición de constantes, los archivos, tipos y variables generales que se utilizan en el sistema.

```
UNIT Def;

INTERFACE

CONST
  TIT_SIST=' ≡     Siatema para el Control y Visualización de los
           Recursos de la Red      ≡';

  MAX_MODOS=100;
  MAX_US=100;

  BLINK=128;

  NULO=#0;
  ARRIBA = #72;
  ABAJO = #80;
  BACKSPACE = #8;
  DEL = #83;
  ENTER = #13;
  ESC = #27;
  IZQ = #75;

  F5  = #63;
  F10 = #68;

TYPE
  meses=0..13;

  str2 =string[2];
  str3 =string[3];
  str8 =string[8];
  str10=string[10];
  str12=string[12];
  str20=string[20];
  str21=string[21];
  str79=string[79];

  accus=RECORD
           CASE primero:boolean OF
           TRUE: (punt_día,Punt_mes:word;
                  cant_día:integer;
                  mcarga,mtpo_conec,mreq,mblq_leídos,
                  mblq_esc:LONGINT);
           FALSE:(fecha,hr_conec,hr_desc:str8;
                  terminal:integer;
                  intruso:boolean;
                  carga,tpo_conec,req,blq_leídos,
```

```
                    blq_esc:LONGINT);
        END;
  f_acct_us=FILE OF accus;

  accdia=RECORD
           CASE Primero:boolean OF
           TRUE: (punt:integer);
           FALSE:(usu:str12;
                  carga:integer;
                  fecha:str8;
                  vec: ARRAY[1..48] OF longint);
         END;
  f_acct_dia=FILE OF accdia;

  rec_tab=RECORD
           CASE Primero:boolean OF
           TRUE :(cant_us:integer);
           FALSE:(usuario:str12;
                  nom_arch:str12;
                  baja:char);
         END;

  reg_mes=RECORD
           usuario:str12;
           mes:MESES;
           anio,cant_intr:byte;
           carga,tpo_conec,blq_esc,blq_leidos,req:longint;
         END;

  total=RECORD
         CASE primero:boolean OF
         TRUE: (punt:word);
         FALSE:(fecha:str8;
         tpo_conec:word;
         blq_esc,blq_leidos,blq_disc,req:longint);
       END;

  parámetros=RECORD
               subdir:string;
               mes,anio:str2;
             END;

  rec_term=RECORD
             dir_term:str21;
             nro_term:integer;
           END;

  vec_eje=ARRAY[1..48] OF str20;

VAR
  buffer:pointer;
IMPLEMENTATION
END. {Unit Def}
```

UNIDAD: *Util*

Definición:
Esta unidad contiene todos aquellos procesos y funciones que son comunes a distintos módulos del sistema.

```
UNIT Util;

INTERFACE

{$S-}

USES Crt,Def,Help,Printer,Vent;

TYPE
  vec_op=array[1..24] of str79;
  vec_ter=array[1..20] of byte;
  mat_mat=array[1..32,1..5]  of longint;
  rec_mat=array[1..48,1..10] of str20;
  tit_mat=array[1..10] of str79;
  long_mat=array[1..10] of byte;
  opcion=byte;

  PROCEDURE CursorOff;
  PROCEDURE CursorOn;
  PROCEDURE Beep;
  PROCEDURE Sacar_bco(VAR linea:string);
  PROCEDURE Seleccion(cant:byte; vec:vec_op; VAR op:opcion);
  PROCEDURE Mostrar_tit(col:byte; tit:tit_mat; long:long_mat);
  PROCEDURE ScrollMat(x1,y1,x2,y2:byte; mat:rec_mat;
                      fil,col:byte; tit:tit_mat; long:long_mat;
                      s_tit:st79; graf:boolean; VAR return:byte);
  PROCEDURE SelecEjes(tit:tit_mat; long:long_mat; fil,col:byte;
                      rec:rec_mat; VAR titX,titY:str79;
                      VAR vecX,vecY:vec_eje; VAR salir:boolean);
  PROCEDURE ImprimirTab(m_izq,cant_fil,cant_col:byte;
                        w_tit,s_tit:str79; rec_m:rec_mat;
                        tit_m:tit_mat; long_col:long_mat);
  PROCEDURE LeerFecha(x1,y1,x2,y2:integer; t:string;
                      VAR dia,mes,anio:str2);
  PROCEDURE InicMat(VAR mat:rec_mat; col,fila:byte);
  PROCEDURE InicMatEnt(VAR mat:mat_mat; col,fila:byte);
  PROCEDURE InicMatXDia(VAR mat:rec_mat; col,fila:byte);
  PROCEDURE InicVec(VAR v:vec_ter;fila:byte);
  FUNCTION LeerSubDir:string;
  FUNCTION LeerMes(x1,y1,x2,y2:integer; t:string):meses;
  FUNCTION DistMes(mes:meses; fecha:str12):boolean;
  FUNCTION Dia(fecha:str8):integer;
  FUNCTION Mes(fecha:str8):integer;
  FUNCTION IgualFecha(dia,mes,anio:str2;fecha:str8):boolean;
```

IMPLEMENTATION

PROCEDURE CursorOff

Definición:
Elimina el cursor de la pantalla.
Datos Entrada:
Datos Salida:

PROCEDURE CursorOn

Definición:
Restaura el cursor en la pantalla.
Datos Entrada:
Datos Salida:

PROCEDURE Beep

Definición:
Efectua un pitido o sonido.
Datos Entrada:
Datos Salida:

PROCEDURE Sacar_bco(VAR línea:string)

Definición:
Elimina blancos a derecha.
Datos Entrada:
- línea: string de entrada.

Datos Salida:
- línea: string sin blancos a derecha.

PROCEDURE Seleccion(cant:byte; vec:vec_op; VAR op:opcion)

Definición:
Selecciona un ítem de un vector de opciones dado.
Datos Entrada:
- cant: cantida o longuitud del vector de ítems.
- vec: vector de opciones.

Datos Salida:
- op: opción seleccionada; si esta es cero(Ø) significa que no hubo selección alguna.

PROCEDURE Mostrar_tit(col:byte; tit:tit_mat; long:long_mat)

Definición:
Coloca los títulos de las columnas en la tablas.

Datos Entrada:
- col: cantidad de columnas.
- tit: vector con título o rótulo que llevaran la columnas.
- long: vector con las longitudes de cada columna.

Datos Salida:

```
PROCEDURE ScrollMat(x1,y1,x2,y2:byte; mat:rec_mat; fil,col:byte;
                    tit:tit_mat; long:long_mat; s_tit:st79;
                    VAR return:byte)
```

Definición:
Muestra en pantalla una matriz de valores y permite realizar scroll.

Datos Entrada:
- x1,y1: esquina superior derecha.
- x2,y2: esquina inferior izquerda.
- mat: matriz a poner en pantalla.
- fil: cantidad de filas de la matriz.
- col: cantidad de columnas de la matriz.
- tit: vector de titulos de la matriz.
- long: vector de longitudes de cada columna de la matriz.
- s_tit: subtítulo puesto en pantalla.

Datos Salida:
- return: devuelbe la condición de salida: 2=graficar; 1=imprimir; 0=salir.

```
PROCEDURE SelecEjes(tit:tit_mat; long:long_mat; fil,col:byte;
                    rec:rec_mat; VAR titX,titY:str79;
                    VAR vecX,vecY:vec_eje; VAR salir:boolean)
```

Definición:
Seleciona los ejes X e Y para ser graficados; retornando los títulos de dichas columnas y los valores a graficar.

Datos Entrada:
- tit: vector de títulos de las columnas de la matriz visualizada.
- long: vector de longitudes de cada columna de la matriz.
- fil: cantidad defilas de la matriz.
- col: cantidad de columnas de la matriz.
- rec: matriz de valores visualizados.

Datos Salida:
- titX: rótulo del eje X.
- titY: rótulo del eje Y.
- vecX: vector con los valoras del eje X.
- vecY: vector con los valoras del eje Y.
- salir: si sale abortando la selección.

```
PROCEDURE ImprimirTab(m_izq,cant_fil,cant_col:byte;
                      w_tit,s_tit:str79; rec_m:rec_mat;
                      tit_m:tit_mat; long_col:long_mat)
```

Definición:
Imprime la tabla puesta en pantalla.
Datos Entrada:
- m_izq: para asignar margen izquiedo a la impresión.
- cant_fil: cantidad de filas a imprimir.
- cant_col: cantidad de columnas a imprimir.
- w_tit: titulo de la impresion.
- s_tit: subtitulo de la impresion.
- rec_m: matriz a imprimir.
- tit_m: vector de títulos de las columnas de la matriz.
- long_col: vector de lonitudes de las columnas de la matriz.

Datos Salida:

```
PROCEDURE LeerFecha(x1,y1,x2,y2:integer; t:string;
                    VAR día,mes,anio:str2)
```

Definición:
Lee una fecha realizando las validaciones correspondientes, generando a su vez la ventana respectiva.
Datos Entrada:
- x1,y1: esquina superior derecha.
- x2,y2: esquina inferior izquierda.
- t: título de la ventana.

Datos Salida:
- día: día leido.
- mes: mes leido.
- anio: año leido.

```
ROCEDURE InicMat(VAR mat:rec_mat; col,fila:byte)
```

Definición:
Inicializa la matriz con caracteres='Ø'.
Datos Entrada:
- fil: cantidad defilas.
- col:cantidad de columnas.

Datos Salida:
- mat: matriz inicializada.

```
PROCEDURE InicMatEnt(VAR mat:mat_mat; col,fila:byte)
```

Definición:
Idem, inicializando con valores enteros=Ø.
Datos Entrada:
Datos Salida:

```
PROCEDURE InicMatXDia(VAR mat:rec_mat; col,fila:byte)
```

Definición:
Idem, inicializando la primer columna con valores correspondientes a lois días del 1 al 31

Datos Entrada:
Datos Salida:

```
PROCEDURE InicVec(VAR v:vec_ter; fila:byte)
```

Definición:
Inicializa vector.
Datos Entrada:
- fila: cantidad de elementos del vector.

Datos Salida:
- v: vector inicializado.

```
FUNCTION LeerMes(x1,y1,x2,y2:integer; t:string):meses
```

Definición:
Función que leer mes de pantalla generando la respectiva ventana; si este es cero(0) significa que se aborto de la lectura..

Datos Entrada:
- x1,y1: esquina superior derecha.
- x2,y2: esquina inferior izquierda.
- t:texto para la ventana.

Datos Salida:
- mes

```
FUNCTION DistMes(mes:meses; fecha:str12):boolean
```

Definición:
Compara un mes con el mes de la fechas y retorna TRUE si son distintas ó FALSE si son iguales.

Datos Entrada:
- mes: mes.
- fecha: fecha completa mm/dd/aa.

Datos Salida:

```
FUNCTION Dia(fecha:str8):integer
```

Definición:
Función que devuelve el día como valor entero.

Datos Entrada:
- facha: mm/dd/aa como caracter.

```
FUNCTION Mes(fecha:str8):integer
```

Definición:
Idem a la anterior con el mes.

Datos Entrada:
Datos Salida:

```
FUNCTION IgualFecha(día,mes,anio:str2; fecha:str8):boolean

Definición:
      Función que compara dos  fechas completas; devuelve TRUE si
   son iguales ó FALSE si son distintas.
Datos Entrada:
   - día: día.
   - mes: mes.
   - anio: año.
   - fecha: fecha completa mm/dd/aa.
```

```
END. {Unit Util}
```

UNIDAD: *Graf*

Observación:
Esta unidad contiene los procesos para graficar.

```
UNIT Graf;

INTERFACE

USES Graph,Crt,Dos,Def;

CONST
  SIZE=1;

TYPE
  pto_pant=^list;
  list=RECORD
         pto:byte;
         prox:pto_pant;
       END;

PROCEDURE SalvarImagen(VAR celda:pto_pant);
PROCEDURE RestaurarImagen(VAR celda:pto_pant);
PROCEDURE Graficar(w_tit,s_tit,titX,titY:str79;
                   vecX,vecY:vec_eje; tope:integer);

IMPLEMENTATION
```

```
PROCEDURE SalvarImagen(VAR celda:pto_pant)
```

Definición:
Salva la imagen de pantalla en memoria, dado que cuando se pasa a modo gráfico esta se pierde.
Datos Entrada:
Datos Salida:
- celda: puntero a la cabeza de la lista que contiene la imgen.

```
PROCEDURE RestaurarImagen(VAR celda:pto_pant)
```

Definición:
Restaura la imagen almacenada en memoria.
Datos Entrada:
- celda: puntero a la cabeza de la lista que contiene la imgen.

Datos Salida:

```
PROCEDURE Graficar(w_tit,s_tit,titX,titY:str79;
                   vecX,vecY:vec_eje; tope:integer);
```

Definición:
Grafica los vectores pasados por parámetro.

Datos Entrada:
- w_tit: titulo de gráfico.
- s_tit: subtitulo de gráfico.
- titX: rótulo del eje X.
- titY: rótulo del eje Y.
- vecX: vector correspondiene a los valores del eje X.
- vecY: vector correspondiene a los valores del eje Y.
- tope: cantidad de valores a reprecentar.

Datos Salida:

```
END. {Unit Graf}
```

UNIDAD: *Help*

Observación:
Esta unidad maneja los mensajes especiales para la comunicación con el usuario y los de error.

```
UNIT Help;

INTERFACE


USES Crt,Vent;

  FUNCTION Error(tipo:integer; st:string):boolean;
  FUNCTION Mensaje(tipo:integer; st:string):boolean;

IMPLEMENTATION
```

FUNCTION Error(tipo:integer; st:string):boolean

Definición:
Envía los mensaje e error al usuario, previamente definido en un vector intermo a la función; dando la posibilidad de adicionar texto.
Esta función devueve TRUE ó FALSE si desea abortar el proceso al haberse producido el error.

Datos Entrada:
- tipo: número que indica el tipo de error.
- st: texto de adición al mensaje.

Datos Salida:
- true ó false

FUNCTION Mensaje(tipo:integer; st:string):boolean

Definición:
Idem, no son mensajes para comunicar o prevenir al usuario antes de sierta acción.

Datos Entrada:

Datos Salida:

```
END.  {Unit Help}
```

CONCLUSIONES

» A lo largo del proyecto hemos estudiado diversos temas que constituyeron un background imprescindible para plantear el desarrollo del ambiente de control y evaluación del trabajo de una red bajo Novell.

» El ambiente planteado se basó en la situación de contexto del momento inicial del proyecto, analizando las posibilidades y restricciones de Novell V 2.11. Posteriormente evolucionamos algunos resultados considerando Novell 3.11.

» Obviamente la evolución de las versiones de Novell obliga a replantear algunos aspectos de la implementación, pero el aporte más importante ha sido el estudio, análisis y definición conceptual de un sistema de apoyo al gerenciamiento de una red bajo Novell.

» Estrictamente el ambiente aporta una interacción con el supervisor de red, le permite:
- Detectar violaciones a normas elementales, tales como la desconección incorrecta de los usuarios o del server de la red.
- Distinción de una conexión incorrecta de un usuario al accionar de un intruso.
- Estudiar la evolución de la utilización de los recursos de la red, dicho consumo se puede verse reflejado por datos generales o detallados por usuario; a lo largo de un período de tiempo o de un día determinado.
- Relacionar el nivel de actividad de un usuario en la red, durante el tiempo que este estuvo conectado.

» Las pruebas realizadas (limitadas por la disponibilidad de acceso a redes con un número considerable de usuarios) fueron satisfactorias y dejan abiertas una serie de inquietudes que permitiría perfeccionar el ambiente en versiones posteriores.

ANEXO A

HERRAMIENTAS DE MONITOREO

Hay varias herramientas de monitoreo disponibles. Algunas son propias de NetWare; otras, como las Frye utilities son de terceros fabricantes.

Se analizaran MONITOR NLM y Frye utilities.

1 MONITOR NLM

La utilidad MONITOR es un de las más potentes, para monitorear la performance del servidor. Esta utilidad (un módulo cargable NetWare que puede ser corrido desde el servidor), puede ser utilizada para ver información sobre:

- Utilización de la CPU y actividad total del servidor .
- Estado de la memoria cache.
- Unidades de disco.
- Manejadores de LAN.
- Módulos NLM cargados.
- Estado de bloqueo de los archivos.
- Utilización de memoria.
- Volúmenes montados.

1.1 Pantalla Principal

La pantalla principal muestra importante información, mucha de la cual describe la utilización del servidor y caching.

La utilización es el porcentaje de tiempo que el servidor esta ocupado.

Cuando el servidor se inicia, varios file cache buffers están disponibles, el nro esta indicado por *original cache buffers*.

El servidor satisface los requerimientos de memoria de nuevos NLMs de estos cache pool, y por un período de tiempo el número de buffers cache disponibles puede disminuir. El nro corriente de files cache buffers es indicado por *Total Cache Buffers*.

Dado que las actividades de I/O del disco del servidor aumenta, no todos los requerimientos de disco pueden ser procesados simultáneamente y algunos tiene que esperar en cola para ser atendidos. El nro de requerimiento de disco en cola de espera es indicado por *Current Disk Requests*.

Un cierto nro de buffers, indicado por *Paquet Receive Buffers*, debe estar disponible para manejar los requerimientos de la estación.

El nro total de buffers asignados para acelerar el acceso al disco es indicados por *Directory Cache Buffers*.

El nro de procesos dedicados a manejar los requerimiento de servicios de archivo está indicado por *Service Processes*. Este nro crecer con la demanda, y nunca disminuye a menos que el servidor sea reiniciado.

El nro total de estaciones corrientemente conectadas al servidor está dado por *Conection In Use*, y el nro total de archivos abiertos en el servidor de archivos está indicado por *Open File*.

1.2 Información de Conexiones

El MONITOR información de conexión, muestra la estadística siguiente:

- Tiempo de conexión. Medido en días, horas y minutos.
- Dirección de la red. Que consiste de tres partes: nro de red(nro de segmento de cable), dirección del nodo, y dirección del zócalo. La dirección del zócalo para la estación shell es 4003(hex); esta es la dirección del software del proceso worstation shell.
- Requerimientos. El nro total de requerimientos de NCP(netware control programa) generados por la conexión de la estación .
- Kilobytes leídos. Mide la cantidad de información leída. Si se permite contar los servicios, se pueden cargar al usuario dichos servicios.
- Kilobytes escritos. Mide la cantidad de información escrita. Si se permite contar los servicios, se pueden cargar al usuario dichos servicios.
- Estado. El estado de un usuario puede ser *Normal*, cuando el usuario esta conectado; *Esperando*, cuando se esta esperando por un archivo bloqueado; *No conectado*, no esta reconocido en la red.

■ Semáforo. Muestra el nro de semáforos utilizados por la estación . Los semáforos son utilizados para arbitrar el acceso a los recursos(NICs[1], áreas de RAM, bus), y también el nro de estación que pueden compartir una aplicación.

■ Bloqueo lógico de registros. Muestra el nro de registros con bloqueo lógico usados por una conexión. Un *bloqueo lógico* se implementa asignando un nombre lógico a un recurso y bloqueando ese nombre. Antes de acceder al recurso se realiza un chequeo del nombre bloqueado. Un bloqueo lógico de registro es forzado por una aplicaciones. Se diferencia al bloqueo físico, ya que son forzados por el operador de la red y pueden bloquear un rango de byte de un archivo. Si otro usuario desea acceder a un rango de bytes que están físicamente bloqueados, dicho usuario recibe un *Access denied* como mensaje de error.

Tipo de bloqueos físicos

Bloqueo físico	Descripción
Bloqueo exclusivo	Bloqueo para que otra persona no pueda leer ni escribir sobre el rango de bytes especificado.
Bloqueo compartido	Son permitidas las lecturas simultaneas, pero solamente puede escribir una estación a la vez.
Bloqueo	Logged para futuros bloqueos
TTS Bloqueo de contención	Desbloqueo para aplicaciones todavía bloqueadas por TTS por que las transacción no se completó.

1.3 Información de Disco

La información de la controladora de disco incluye lo siguiente:

■ Drive. Nombre del controlador de disco que opera el hard

[1]NertWare Interface Card

disk.
- Tamaño del disco. Tamaño en megabytes de todas las particiones del hard disk.
- Particiones. Númerode particiones definidas sobre el hard disk.
- Mirror estado. Los valores son *Mirrored*, el cual indica que los datos están almacenados en más de un hard disk; *Not Mirrored*, indica que los datos están sobre un solo discos; *Remirroring*, el cual indica que los datos están comenzando a transferirse a otro disco para tener en ambos disco los mismos datos (mirrored).
- Estado del Hot fix. *Normal* indica que hot fix esta habilitado. *Not-hot-fix* indica que hot fix esta deshabilitado o falla.
- Bloques de particiones. Espacio total de bloques de disco sobre el disco del servidor .
- Bloque de datos. El espacio del hard disk (en bloques) disponible para datos.
- Bloques de redirección. Número de bloque reservados para el área de redirección del Hot Fix.
- Bloques redireccionados. Número de bloques con fallas y redireccionados por el hot fix.
- Bloque reservado. Bloques reservados para las tablas del hot fix.

Los campos del estado del Drive son los siguientes:
- Segmento de volumen sobre el Drive.
- Verificación de lectura después de la escritura.
- Estado de la luz del drive.
- Estado operativo del drive. Usado para manualmente desactivar/activar el disco.
- Estado de los dispositivos removibles montados para desmontar o montar manualmente medios removibles.
- Estado de bloqueo de los dispositivos removibles. Utilizado para bloquear un medio removible y prevenir su remoción física. Si el estado es desbloqueado, el medio removible puede ser desmontado y físicamente removido.

1.4 MONITOR Información de la LAN

La información del driver de la LAN indica el nro de versión, la dirección del nodos del NIC en el file servidor , limitaciones del protocolo del driver, y la dirección de la red del segmento de cable sobre el cual el driver esta trabajando.

Se pueden obtener dos tipos de estadísticas: Genéricas y Personalizadas.

Estadística genérica para Drivers de LAN

Parámetro	Descripción
Total de paquetes enviados	Indica cual driver esta manejando la mayor cantidad de tráfico , y el valor del mismo.
Total de paquetes recibidos	Indica cual driver esta manejando la mayor cantidad de tráfico, y el valor del mismo.
No ECB contador disponible	Un contador que se incrementa cuando no hay buffers disponibles y se recibe un nuevo paquete.
Cuenta de paquetes demasiado grandes enviados	Un contador que se incrementa cuando el servidor transmite un paquete demasiado grande como para que el NIC lo pueda manejar; indica un seteo incorrecto de MAXIMUNM PHYSICAL RECEIVE PACKET SIZE o un problema con el software del servidor o del driver NIC.
Cuenta de paquetes demasiado pequeños enviados	Un contador que se incrementa cuando el servidor transmite un paquete demasiado pequeño como para que el NIC lo pueda manejar; indica un problema con el software del servidor o del driver NIC.
Contador de desbordamiento (Overflow) en los paquetes recibidos	Un contador que se incrementa cuando el servidor recibe un paquete demasiado grande como para almacenarlo en un buffer cache; indica un problema con el software del estación en la negociación de una medida adecuada del paquete; puede también ser un problema con el driver del NIC o la tarjeta adaptadora en el transmisor
Cuenta de paquetes demasiado grandes recibidos	Un contador que se incrementa cuando el servidor recibe un paquete demasiado grande como para que el NIC lo pueda manejar; indica un seteo incorrecto de MAXIMUNM PHYSICAL RECEIVE PACKET SIZE o un problema con el software del transmisor o del driver NIC.
Cuenta de paquetes demasiado pequeños recibidos	Un contador que se incrementa cuando el servidor recibe un paquete demasiado pequeño como para que el NIC lo pueda manejar; indica un problema con el software del transmisor o del driver NIC.

Estadística genérica para Drivers de LAN
(continuación)

Parámetro	Descripción
Misceláneos de errores de paquetes enviados	Un contador de todos los errores; que se incrementa cuando un error ocurre durante la transmisión de un paquete que no esta justo en alguna otra categoría. Un valor muy grande puede indicar problemas con el hardware del la red.
Misceláneos de errores de paquetes recibidos	Un contador de todos los errores que se incrementa cuando un error ocurre durante la recepción de un paquete que no esta justo en alguna categoría. Un valor muy grande puede indicar problemas con el hardware del la red.
Contador de paquetes retransmitidos	Un contador que se incrementa cuando el servidor retransmite un paquete por que hay un error de hardware; indica un problema con el cableado o el hardware de la NIC, o un problema con el tiempo de retardo a través de la WAN (Wide Area Network). Pruebe incrementando el contador de reintentos usando el comando LOAD.
Error en la suma de chequeo (checksum errors)	Un contador que se incrementa cuando un error en los datos es detectado por el CRC[2]en el fin del paquete; indica un problema con el hardware de la NIC, ruido, o el cableado.
Contador de desigualdades que recibe el hardware	Un contador que se incrementa cuando la longitud del paquete recibido por el NIC no coincide con la del campo longitud del paquete; esto indica un problema con el NIC o el driver del NIC.

[2]CRC: (Cyclical Redundancy Checking) verificación cíclica de la redundancia
Una técnica de verificación de errores utilizada para asegurar la precisión de la transmisión de código digital a través de un canal de comunicaciones.

1.5 Actividad de los archivos abiertos/bloqueados

La información de la actividad de los archivos abiertos/bloqueados es la siguiente:

- Use Count. Es el nro de conexiones que tiene un archivo abierto o bloqueado.
- Open Count. Define el nro de conexiones del corriente archivo abierto.
- Open Read. Indica el nro de conexiones de lectura del archivo.
- Open Write. Indica el nro de conexiones de escritura para el archivo. La siguiente relación se cumple en todo momento:

  Open Count = Open Read + Open Write
- Los campos Deny Read y Deny Write indican si el bloqueo de apertura del archivo es Exclusivo o Compartido. Deny Read indica el nro de conexiones que tiene un archivo abierto, pero denegado para la operación de lectura (Archivo con bloqueo exclusivo). Deny Write indica el nro de conexiones que tiene un archivo abierto, pero denegado para la operación de escritura por otra estación (Archivo con bloqueo compartido de lectura).
- Status. Indica si el archivo esta bloqueado o desbloqueado.

1.6 Utilización de los recursos

El MONITOR de utilización de los recursos muestra los siguientes campos:

- Permanent Memory Pool. Memoria usada por los procesos cuando ellos no intentan retornarla por algún tiempo. Cuando se necesitada más memoria, los buffers cache son transferidos a esos pool; esos buffer cache no serán retornados al cache buffer pool. El buffer cache de directorio así como los buffers de paquetes recibidos están contenidos en este pool.

- Alloc Memory Pool. Memoria utilizada por cargar módulos, cuando ellos no intentan utilizarla por grandes períodos de tiempo. Cuando se necesita más memoria para este pool, la misma es transferida desde le pool de memoria permanente; esta memoria no será retornada al pool de memoria permanente.

- Cache Buffer. Es la cantidad de memoria en el cache buffer pool. Esta memoria corrientemente esta siendo utilizada por el file caching. La cantidad de memoria en esta área debe ser mayor que en cualquier otra área.

- Cache movable memory. Memoria asignada directamente en el cache buffer pool, cuando es liberada se retorna al cache buffer pool. Difiere del cache non-movable memory en que el manejador de memoria puede mover la ubicación de esos bloques de memoria para optimizar su uso. Tablas de volumen y tablas de Hash utilizan tipicamente esta memoria.

- Cache Non-Movable Memory. Memoria asignada directamente en el cache buffer pool, cuando se liberada esta memoria es retornada al cache buffer pool. Se utiliza cuando grandes partes de memoria son buscadas.

- Total Server Work Memory. Es la suma de todos los pool de memoria.

2 Frye Tools

Esta herramienta (Frye Utilities for NetWare Manegement, de Frye Computer Systems) puede monitorear performance sobre NetWare 2.x y 3.x . Además del monitoreo de performance la utilidades proveen diagnósticos, documentación del servidor y reportes semanales de mantenimiento. La Frye utilities para NetWare incluye un sistema de aviso automático de errores al alcanzar ciertos umbrales.

Los notificaciones de alerta pueden ser enviadas a usuario de la red a través de coreo electrónico, fax, etc.

Algunas de las condiciones de alerta son:

- El servidor no responde.
- Umbral para pico de conexiones.
- Umbral para las conexiones corrientes.
- Driver utilizado por encima del umbral.
- Utilización media del servidor por arriba de un umbral.
- Valor medio de I/O de disco pendiente por arriba de un umbral.
- Memoria cache disponible por debajo de determinado umbral.

3 Nu-Mega Technology tools

Cuando una nueva aplicación o sistema NLM es instalado, debe compartir la CPU con el sistema operativo NetWare. Dependiendo de como están escritos los NLM, pueden causar un serio impacto en la performace del servidor (y en toda la red).

Nu-Mega Technology provee Net-Check y el NLM-Profile Tools, los cuales son utilizados para detectar y monitorear a los NLMs con comportamiento deficiente. El Net-Check Tools puede ser utilizado para monitorear NLMs y evitar que se escriba sobre otro código NLM o área de datos. Net-Check es capaz de detectar a NLM que hacen una escritura de memoria fuera de sus limites. Puede ser configurado para prevenir esas sobrescrituras o para permitirlas e informar que han ocurrido.

Net-Check provee esta protección corriendo sobre un nivel superior al NetWare y los NLMs y accediendo a los mecanismos de protección de hardware de la CPU 80386.

Antes de instalar un nuevo NLM se debe correr Net-Check, Nu-Mega afirma que Net-Check puede ser corrido todas las veces que se desee sin afectar la performance del servidor .

El Net-Profile Tools da un perfil del uso de la CPU por los procesos y los NLMs corriendo en le servidor. Aunque el mecanismo despachador de NetWare es eficiente un NLM pobremente escrito puede abusar del uso de la CPU y degradar la performance del servidor. Net-Profile puede detectar este tipo de NLMs.

El Net-Profile Tools utiliza una gran cantidad de tiempo de procesamiento en la operación del servidor, por lo que debe ser corrido en la instalación de una nueva aplicación NLM y cuando se está seguro que todos los NLMs se comportan correctamente Net-Profile puede ser descargado de la memoria del servidor .

BIBLIOGRAFIA

1- TANENBAUM, ANDREW S.: "Redes de Ordenadores", Segunda edición 1991.

2- NOVELL: NetWare v 2.11, 1988

3- NETWARE CONNECTION: "NetWare Optimizacion", pág. 9-13, Noviembre 1992.

4- NETWARE CONNECTION: "NetWare Optimizacion, File Server / Operating System Optimization", pág. 18-24, Marzo 1993.

5- COMPU MAGAZINE: "EISA vs MCA", pág 28-38, Marzo 1990.

6- PC MAGAZINE: "Software para Analizar su Servidor", pág. 91-110, Marzo 1993.

7- KARANJIT, SIYAM: "NetWare de Professional Reference", 1992.

8- NOVELL: NetWare v 3.11 "System Administration", 1991.

9- NOVELL: NetWare v 3.11 "Utilities Reference", 1991.

10- NEWTON A. DE CASTILHO LAGES
JOSE MARCOS SILVANOGUEIRA : "Introducao a los Sistemas Distribuidos", 1986.

11- JENNIFER SEBERRY
JOSEF PIEPRZYK: "CRYPTOGRAPHY, An Introduction to Computer Security" , 1989.

12- DANIEL ALONZO: "Introducción a las Redes Locales" ,(curso de la sociedad de computación, IEEE Arg.), Mayo 1993.