UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

INTERFAZ DE ADMINISTRACIÓN DEL SISTEMA *SECUREFTP*

INFORME DE ACTIVIDADES PROFESIONALES

QUE PARA OBTENER EL GRADO DE:
INGENIERO EN COMPUTACIÓN

PRESENTA:

EDUARDO PALMA AVILA

CON LA ASESORÍA DE:

M. EN C. MA. JAQUELINA LÓPEZ BARRIENTOS

MÉXICO, D. F. MAYO DE 2010

Agradecimientos

Agradezco a Dios por haberme dado la oportunidad de pertenecer a la mejor familia del Mundo. Mis padres y hermanos han estado conmigo en todo momento y siempre me han brindado todo su amor, respeto y cariño necesarios para concluir mis estudios universitarios.

A mis padres me han dado el mejor ejemplo que un hijo puede recibir. No me alcanzará la vida para pagarles todo lo que han hecho por mis hermanos y por mí.

A mis hermanos me han dado la alegría y energía necesarias para luchar día a día por ser el mejor en todo y superar todos los obstáculos.

A Adrianita por brindarme su amor, amistad, comprensión y apoyo. Por compartir conmigo su vida durante mi última etapa en la Universidad y en este inicio de mi vida profesional.

A mis amigos con los que compartí el privilegio de pisar las aulas de la Facultad de Ingeniería de la Universidad Nacional Autónoma de México; gracias a su amistad y lealtad logré muchas cosas.

A todos mis maestros que compartieron su conocimiento, experiencia y profesionalismo conmigo, especialmente a la profesora Ma. Jaquelina López Barrientos por instruirme y ayudarme a dar el último paso para concluir con mis estudios universitarios.

A mis compañeros del Departamento de Seguridad en Cómputo de DGSCA que me mostraron la riqueza e importancia de esta disciplina.

A Pilar Revilla, Alejandro Villanueva, Karen Reyes y Jesús Velazco por haberme dado la oportunidad de desarrollarme profesionalmente y crecer dentro de un proyecto, por la confianza que me brindaron y por el apoyo incondicional.

Índice

Objetivo

El objetivo de este proyecto es desarrollar una interfaz de administración para el sistema *SecureFTP* que cumpla con las características de ser eficiente y amigable para los usuarios.

A través de un análisis minucioso y debido, principalmente, a la dificultad en la administración del sistema, se desea desarrollar una interfaz efectiva y de fácil uso con el objetivo de reducir la tasa de errores y los tiempos de respuesta en los administradores del sistema *SecureFTP* en la Ciudad de México, México, y Budapest, Hungría, sedes mundiales de los Centros de Excelencia para la Administración de Seguridad de la Información de Citigroup.

Introducción

Hasta hace un par de décadas, las únicas computadoras eran mainframes. Eran muy pocas y se utilizaban para tareas específicas, generalmente para ejecutar grandes trabajos por lotes (batch jobs), uno a la vez, y llevando a cabo cálculos muy complejos. Si los usuarios estaban conectados a los mainframes era a través de terminales que tenían una funcionalidad limitada y eran totalmente dependientes del mainframe para sus operaciones y ambiente de procesamiento. Esto representaba un ambiente cerrado con pequeñas amenazas y pocas brechas de seguridad y vulnerabilidades que pudieran ser explotadas. Esto no significaba que las cosas fueran perfectas, las vulnerabilidades de seguridad no existían, y la gente vivía en una utopía informática. En lugar de ello, eran pocas personas que trabajaban en una "caja de cristal" y sabían como operar el mainframe. Ellos decidían quién podía acceder a los sistemas y cuándo. Esto propiciaba un ambiente mucho más seguro, debido a su simplicidad, que lo que vemos en el mundo distribuido e interconectado de hoy.

En aquellos días, los sitios web que describen los pasos necesarios para vulnerar una aplicación o sistema operativo no existían. Los protocolos de red solo eran comprendidos por pocas personas, en comparación del vasto número de individuos que hoy lo hacen. Herramientas para saturar buffers y escanear puertos no existían. Esto representaba un ambiente realmente hermético que solo poca gente comprendía.

Si las redes se interconectaban era solo para realizar tareas muy específicas y las corporaciones no dependían totalmente del procesamiento de datos como lo hacen hoy. Los sistemas operativos de aquel tiempo tenían problemas, *bugs* en el software, y vulnerabilidades, pero no mucha gente estaba interesada en tomar ventaja de estas situaciones. Si los operadores de mainframe se encontraban con algún problema de software, lo que hacían era modificar el código para solucionarlo. Todo esto no ocurrió hace mucho tiempo, tomando en cuenta dónde nos encontramos hoy.

Debido a que las compañías se volvieron más dependientes del poder computacional de los mainframes, la funcionalidad de los sistemas creció y varias aplicaciones fueron desarrolladas. Fue claro que al dar a los empleados pequeños periodos de tiempo de acceso a los mainframes no era tan productivo como se pensaba. El procesamiento y poder de cómputo le fue otorgado a los empleados, habilitándolos a ejecutar pequeños procesos en sus computadoras personales mientras que los procesos grandes seguían

ejecutándose en la “caja de cristal”. Esta tendencia continuó y las computadoras personales se volvieron más independientes y autónomas, y solo necesitaban acceso al mainframe para realizar ciertas tareas.

Las computadoras personales se volvieron más eficientes y continuaron realizando más tareas y tomando más responsabilidades. Esto mostró que varios usuarios accediendo al mainframe era un modelo ineficiente; era necesario dar acceso a los empleados a ciertos componentes para realizar sus tareas de una manera eficiente y efectiva. Este enfoque condujo al nacimiento del modelo cliente/servidor. Aunque muchas computadoras personales tenían la capacidad para procesar sus propios datos, cálculos y operaciones lógicas, no tenía mucho sentido que cada computadora almacenara información necesaria para otras computadoras. De esta manera, los programas y datos fueron centralizados en servidores, con PC’s accediendo a ellos cuando era necesario y utilizando menos frecuentemente los mainframes.

Con el aumento en la exposición al cómputo y procesamiento, los usuarios que utilizaban computadoras aprendieron más sobre cómo usar la tecnología y cómo sacar el mayor provecho posible de ella. Sin embargo, las cosas buenas frecuentemente tienen su lado oscuro. Al quitarle la tecnología al mainframe y dársela a muchos individuos trajo muchos problemas que no se habían visto antes. Ahora, eran miles de usuarios inexpertos quienes tenían mucho más acceso a datos y procesos importantes. Las barreras y mecanismos de protección no estaban allí para proteger a los empleados y sistemas de los errores, de manera que muchos datos se corrompieron accidentalmente, y este tipo de fallas afectaron muchos otros sistemas en lugar de solo uno.

Debido a que mucha gente utilizaba los sistemas, el software tuvo que desarrollarse utilizando interfaces amigables para que todos pudieran usar la misma plataforma. Los operadores de mainframe comprendían lo que los sistemas esperaban, como era el formato de entrada, y como leer la salidas. Cuando este poder fue puesto en las computadoras de los usuarios, cada imaginable (e inimaginable) entraba fue utilizada, lo cual corrompía la información y dañaba los sistemas operativos.

Importancia de la Seguridad de la Información

Pronto, las compañías se dieron cuenta de que los empleados tenían que ser protegidos de ellos mismos y que los datos tenían que protegerse de accidentes y errores. Los empleados necesitaron capas de software entre ellos y los componentes del sistema operativo y los datos que pudieran destruir. Implementado estas capas no solo mejoró la seguridad – al separar a los usuarios de los componentes y archivos del sistema operativo – sino que también incrementó la productividad ya que la funcionalidad continuó implementándose haciendo las computadoras más útiles para los negocios y las personas.

El mundo computacional evolucionó y crecieron las relaciones simbióticas entre las tecnologías de hardware, poder de procesamiento y software. Una vez que esto sucedió, se les agregó más memoria y espacio de almacenamiento a las computadoras para soportar el nuevo software. Cuando el software irrumpió, no se tomaron en cuenta los registros necesarios ni las unidades de control, la industria estuvo en el lugar y en el momento adecuado para desarrollar las piezas faltantes. El hardware creció y proporcionó una plataforma estable y rica para el software, los programadores desarrollaron aplicaciones que proporcionaron funcionalidad y posibilidades nunca antes vistas en los años anteriores. Esto representa un maravilloso juego de evolución que no parece tener fin.

Hasta este punto todo parece perfecto, pero, ¿en dónde está la seguridad de la información?

En un inicio, los problemas asociados con acercar el cómputo a los usuarios trajeron consigo muchos errores, vallas tecnológicas y problemas operacionales que no se habían visto antes. Las computadoras son herramientas. Como un cuchillo puede ser utilizado como una herramienta para cortar carne y vegetales, también puede utilizarse como una herramienta peligrosa si cae en manos de alguien con malas intenciones. Las computadoras han traído a la sociedad vastas capacidades y funcionalidad pero también han traído métodos complejos de destrucción, fraude, abuso y robo de identidad.

Debido a que las computadoras están construidas en capas (plataformas de hardware, chips, sistemas operativos, kernels, interfaces de red, servicios y aplicaciones), estos problemas complejos han estado entretejidos a través de los entornos computacionales.

La solución a estos inconvenientes ha sido cubriendo los agujeros y escribiendo mejor software, proporcionando un mejor perímetro de seguridad a nivel físico (cámaras de seguridad, guardias, sistemas biométricos, etc.) y a nivel lógico (firewalls, sistemas detectores de intrusos, etc.); se dice fácil pero no lo es debido a la densidad de funcionalidad dentro de una infraestructura, problemas de interoperación y la disponibilidad de los requerimientos necesarios.

Durante un corto periodo de tiempo, la gente y los negocios se han vuelto muy dependientes de la tecnología computacional y la automatización, en varios aspectos de sus vidas. Las computadoras ejecutan utilidades públicas, sistemas militares de defensa, instituciones financieras y equipo médico, y son muy utilizadas en cada sector de los negocios. Casi cada compañía depende del procesamiento de los datos por una u otra razón. Este nivel de dependencia y la extensión en la integración que la tecnología ha alcanzado en nuestras vidas han hecho a la seguridad una disciplina mucho más necesaria y esencial.

Las computadoras y las redes están presentes en cada faceta de la vida moderna. Somos altamente dependientes de estas tecnologías para comunicación, transferencia de fondos, administración de utilidades, servicios gubernamentales, acciones militares, y mantenimiento de información confidencial. Utilizamos la tecnología para proporcionar energía, suministro de agua, servicios de emergencia, sistemas de defensa, banca electrónica y servicios de salud pública. Al mismo tiempo, esta tecnología se está utilizando para llevar a cabo acciones ilegales y maliciosas, tales como el robo de información con fines de lucro, uso fraudulento de sistemas telefónicos, trasmisión ilegal de secretos comerciales y propiedad intelectual, modificación no autorizada de sitios web por razones políticas, interrupción de comunicaciones, revelación de secretos críticos y estrategias nacionales e incluso terrorismo [1].

Para que las computadoras y redes de datos cumplan con las funciones para las que fueron creadas, es necesario que la información resguardada y procesada en estos sistemas cuente con los siguientes servicios de seguridad:

a) Confidencialidad: capacidad de asegurar que sólo las personas autorizadas tienen acceso a la información o recurso en cuestión.

b) Autenticación: acción de verificar la identidad del usuario o proceso que desea acceder al recurso o la información, esto es, “validar que efectivamente sea quien dice ser”.
c) Integridad: capacidad de evitar que los datos sean modificados por usuarios o procesos no autorizados para ello.
d) No repudio: protección que se ofrece para prevenir que los emisores o los receptores de negar un mensaje transmitido.
e) Control de acceso: habilitar para limitar y determinar qué usuario o proceso está autorizado para acceder a un recurso o información; así, este servicio se ejecuta con el fin de que un usuario sea identificado y autenticado de manera exitosa para que entonces le sea permitido el acceso.
f) Disponibilidad: posibilidad de acceder a los recursos o información requeridos cuando sea oportuno hacerlo, a la hora que sea necesario y tantas veces como se pretenda, siempre y cuando se esté dentro del rango de servicio otorgado por la entidad responsable de los recursos [2].

Hackers* e *Insiders

Hubo un tiempo en el que las actividades de los hackers, los virus y los incidentes de *malware* eran relativamente benignos. Muchos hackers llevaban a cabo sus actividades para impresionar a sus amigos y mostrar lo inteligentes que eran al interrumpir algunos negocios, pero en general su intención no era causar daños masivos.

La amenaza del *hacking* por “diversión” está desapareciendo y está siendo reemplazada rápidamente por razones lucrativas.

Hoy en día, las brechas de seguridad de la información provocan que el malware y los hackers tengan víctimas y metas específicas. Los hackers trabajan para robar datos y utilizarlos para obtener beneficios de ello, desvían fondos de las cuentas en línea y llevan a cabo extorsión cuando descubren agujeros de seguridad en el programa de seguridad de una compañía. Algunos individuos son incluso contratados por organizaciones criminales para llevar a cabo tales actividades.

Sin embargo, en una encuesta[1] realizada por la revista CSO, en conjunto con el Servicio Secreto de los Estados Unidos, el CERT de la Universidad de Carnegie Mellon y la empresa Deloitte, muestran que en el periodo de Agosto de 2008 a Julio de 2009, más de un tercio (37%) de más de 500 encuestados - entre los que se encuentran empresas de todo el Mundo, gobiernos, profesionales y consultores de Seguridad de la Información – han sufrido un incremento en los *cibercrímenes* en comparación al año anterior. Mientras que los *hackers* (aquellos que accedieron a sus redes de datos sin autorización) son los principales culpables de los delitos informáticos en general, los ataques más dañinos y costosos son los causados por *insiders* (empleados o contratistas con acceso autorizado). Una cuarta parte de todos los ataques fueron cometidos por fuentes desconocidas. El 51% de los encuestados fueron víctimas de algún ataque por parte de *insiders* mientras que el 67% indica que fueron mucho más costosos que los ataques externos. La mayoría de estos ataques fueron cometidos por error y sin intención por parte de los empleados. La tecnología avanza pero también evolucionan los métodos para cometer crímenes cibernéticos. Los hackers invaden las organizaciones utilizando virus, gusanos u otros códigos maliciosos, *phishing* y spyware, mientras que los *insiders* revelan información personal o sensible de manera intencional o por error, tienen acceso a sistemas de información o redes a los que no deberían, roban información de propiedad intelectual, etc.

Las instituciones que se preocupan por sí mismas y por sus clientes procuran contar con las soluciones tecnológicas más sofisticadas para proteger uno de sus recursos más importantes, la información que se almacena y procesa dentro de sus redes de datos. De esta manera se protege su infraestructura de ataques, principalmente externos.

Sin embargo, de poco o nada sirve esta tecnología cuando dentro de las organizaciones existen huecos de seguridad informática. La falta de aplicación del principio del menor privilegio, es decir, proporcionar a los usuarios las herramientas necesarias para llevar a cabo su trabajo y no más; el uso de sistemas no seguros, los cuales no verifican entradas y salidas de datos válidos, la falta de menús para restringir la funcionalidad en los sistemas, provocan que no se cumplan de manera adecuada algunos de los servicios de seguridad mencionados anteriormente.

[1] http://www.cert.org/archive/pdf/ecrimesummary10.pdf

Quizá no sea intencionalmente pero los usuarios pueden insertar datos inválidos a los sistemas, borrar o alterar información existente, aún así los resultados son los mismos que si un atacante externo lo hiciera. Al contener datos inválidos, el sistema puede sufrir de un comportamiento inesperado, pudiendo hasta corromper la información, afectando la integridad y disponibilidad de la misma.

Al realizar modificaciones por error se afectan, de manera directa o indirecta, los servicios de seguridad de la información.

Estos errores humanos ocasionan pérdidas económicas, daño a la reputación de las organizaciones, incumplimiento de las leyes para la protección de los datos, incumplimiento de acuerdos corporativos, etc. Los resultados pueden ser catastróficos: penas de cárcel, multas e incluso pueden llevar a la quiebra a las compañías.

El desafío de proteger la información

El reto de la seguridad de la información es muy complicado debido a que los datos se almacenan en muchos lugares: dentro de archivos y carpetas en equipos personales y portátiles, servidores de la empresa u otros dispositivos electrónicos. Además, los datos se replican periódicamente y se trasladan de un punto a otro dentro de una organización, así como intercambio de archivos con clientes, socios y organismos reguladores. Para entender realmente las vulnerabilidades de los datos, una organización necesita entender el ciclo de vida global de sus activos de datos importantes, desde el momento de crearlos hasta que se destruyen, incluyendo todas las etapas intermedias.

En última instancia, las empresas tienen que lidiar con el equilibrio entre los riesgos asociados con la pérdida, la corrupción y la divulgación de datos sensibles, y el valor adquirido al almacenar, analizar y compartir esa información. Si las ganancias potenciales no compensan los riesgos para un determinado conjunto de datos, la organización no debe asumir los riesgos [4].

Existen mecanismos de seguridad que pueden ser utilizados para proveer los servicios de seguridad citados anteriormente:

- Intercambio de autenticación: corrobora que una entidad, ya sea origen o destino de la información, es la deseada; por ejemplo, A envía un número aleatorio cifrado con la clave pública de B, B lo descifra con su clave privada y se lo reenvía a A, demostrando así que es quién pretende ser. Por supuesto, hay que ser cuidadosos a la hora de diseñar e implementar estos protocolos, ya que existen ataques para desbaratarlos.
- Cifrado: garantiza que la información no es inteligible para individuos, entidades o procesos no autorizados a través de lo cual proporciona confidencialidad a los datos. Consiste en transformar un texto en claro mediante un proceso de cifrado en un texto cifrado, gracias a una información secreta o clave de cifrado.
- Integridad de datos: este mecanismo implica el cifrado de una manera comprimida de datos a transmitir, llamada generalmente valor de comprobación de integridad (Integrity Check Value ICV).
- Firma digital: este mecanismo implica el cifrado, por medio de una clave secreta del emisor, de una cadena comprimida de datos que se va a transferir. La firma digital se envía junto con los datos ordinarios, de manera que el mensaje se procesa en el receptor, para verificar su integridad.
- Control de acceso: esfuerzo para sólo aquellos usuarios autorizados accedan a los recursos del sistema o la red, como por ejemplo mediante las contraseñas de acceso.
- Tráfico de relleno: consiste en enviar tráfico espurio junto con los datos válidos para que el atacante no sepa si se está enviando información, ni qué cantidad de datos útiles se están transmitiendo.
- Control de encaminamiento: permite enviar determinada información por determinadas zonas consideradas clasificadas. Asimismo, posibilita solicitar otras rutas, en caso que se detecten persistentes violaciones de integridad en una ruta determinada.
- Unicidad: consiste en añadir a los datos un número de secuencia, la fecha y hora, un número aleatorio, o alguna combinación de las anteriores, que se incluyen en la firma digital o integridad de datos. De esta forma se evitan amenazas como la reactuación o resecuenciación de mensajes.

Los mecanismos básicos pueden agruparse de varias formas para proporcionar los servicios previamente mencionados, pero independientemente de cuántos sean y cómo se integren, los mecanismos de seguridad siempre deberán contener tres componentes principales:

- Una información secreta, como claves y contraseñas, conocida por las entidades autorizadas.
- Un conjunto de algoritmos, para llevar a cabo el cifrado y descifrado, y generación de números aleatorios.
- Un conjunto de procedimientos, que definen cómo se usarán los algoritmos, quién envía qué, a quién y cuándo [2].

La implementación de estos mecanismos de seguridad depende de la importancia de los datos y de un análisis de riesgos sobre las vulnerabilidades que pueden comprometer la información.

Sea cual sea el diseño de seguridad de la información de una empresa, es esencial el desarrollo regular de campañas de concientización sobre la importancia de proteger los datos**.** Lo más importante para proteger la información de una empresa es mostrar a sus empleados y socios los riesgos que implican el dar a conocer información sensible y la importancia de proteger el activo más valioso del negocio. En esta situación, la cultura es lo más importante. Sin ella, de muy poco o nada sirven los mecanismos de seguridad mencionados.

Capítulo 1

Marco teórico

El sistema *SecureFTP* se basa en el protocolo de transferencia de archivos FTP, cifrado de datos utilizando cifrado por intercambio de llaves GPG e Infraestructura de llave pública *Entrust*.
A continuación se definen estos conceptos y se muestra su importancia para proteger el intercambio seguro de información.

1.1 FTP (Protocolo de Transferencia de Archivos)

Es un protocolo de red para la transferencia de archivos entre sistemas conectados a una red TCP (Transmission Control Protocol), basado en la arquitectura cliente-servidor. Desde un equipo cliente se puede conectar a un servidor para descargar o enviar archivos, independientemente del sistema operativo utilizado en cada equipo.

El Servicio FTP es ofrecido por la capa de Aplicación del modelo de capas de red TCP/IP al usuario, utilizando normalmente el puerto de red 20 y el 21. Un problema básico de FTP es que está pensado para ofrecer la máxima velocidad en la conexión, pero no la máxima seguridad, ya que todo el intercambio de información, desde el login y password del usuario en el servidor hasta la transferencia de cualquier archivo, se realiza en texto claro sin ningún tipo de cifrado, con lo que un posible atacante puede capturar este tráfico, acceder al servidor, o apropiarse de los archivos transferidos.

En 1969, nació ARPANET como una pequeña red de pocos sistemas que transmitían información de unos a otros mediante paquetes conmutados (lo que sería en el futuro Internet), y tres años más tarde un grupo de investigadores del MIT presentó la propuesta del primer "Protocolo para la transmisión de archivos en Internet". Era un protocolo muy sencillo basado en el sistema de correo electrónico pero sentó las bases para el futuro protocolo de transmisión de archivos (FTP).

En 1985 se termina el desarrollo del aún vigente protocolo para la transmisión de archivos en Internet (FTP), basado en la filosofía de cliente-servidor.

La expansión en el uso de este protocolo se produce en 1995, año en el que puede considerarse el nacimiento del Internet comercial. Desde ese momento su crecimiento ha superado todas las expectativas. En este año la World Wide Web

supera a FTP transformándose en el servicio preferido de la red, después de que el año anterior superase en popularidad a Telnet.

Con la llegada del World Wide Web y de los navegadores, ya no fue necesario conocer los comandos de FTP, este protocolo se puede utilizar escribiendo la URL del servidor al que queramos conectar en el navegador web, indicando con ftp:// que vamos a contactar con un servidor ftp y no con un servidor web (que sería http://).

La versión original del File Transfer Protocol fue publicado como RFC [] 114 el 16 de abril de 1971, y más adelante reemplazado por el RFC 765 (junio de 1980) y el RFC 959 (octubre de 1985), la versión que se usa actualmente. Muchos han propuesto alternativas a la versión de 1985, como por ejemplo el RFC 2228 (junio de 1997) que propone extensiones de seguridad y la RFC 2428 (septiembre de1998) que añade soporte para IPv6 y define un nuevo tipo de modo pasivo.

1.2 Modelo FTP

En el modelo, el intérprete de protocolo (PI) de usuario, inicia la conexión de control en el puerto 21. Las órdenes FTP estándar las genera el PI de usuario y se transmiten al proceso servidor a través de la conexión de control. Las respuestas estándar se envían desde el PI del servidor al PI de usuario por la conexión de control como respuesta a las órdenes.

Estas órdenes FTP especifican parámetros para la conexión de datos (puerto de datos, modo de transferencia, tipo de representación y estructura) y la naturaleza de la operación sobre el sistema de archivos (almacenar, recuperar, añadir, borrar, etc.). El proceso de transferencia de datos (DTP) de usuario u otro proceso en su lugar, debe esperar a que el servidor inicie la conexión al puerto de datos especificado (puerto 20 en modo activo o estándar) y transferir los datos en función de los parámetros que se hayan especificado.

Vemos también en el diagrama que la comunicación entre cliente y servidor es independiente del sistema de archivos utilizado en cada computadora, de manera que

no importa que sus sistemas operativos sean distintos, porque las entidades que se comunican entre sí son los PI y los DTP, que usan el mismo protocolo estandarizado: el FTP.

También hay que destacar que la conexión de datos es bidireccional, es decir, se puede usar simultáneamente para enviar y para recibir, y no tiene por qué existir todo el tiempo que dura la conexión FTP.

1.3 Servidor FTP

Un servidor FTP es un sistema que se ejecuta en un equipo servidor normalmente conectado a Internet como el que se muestra en la figura 1.1 (aunque puede estar conectado a otros tipos de redes, LAN, MAN, etc.). Su función es permitir el intercambio de datos entre diferentes servidores/computadoras.

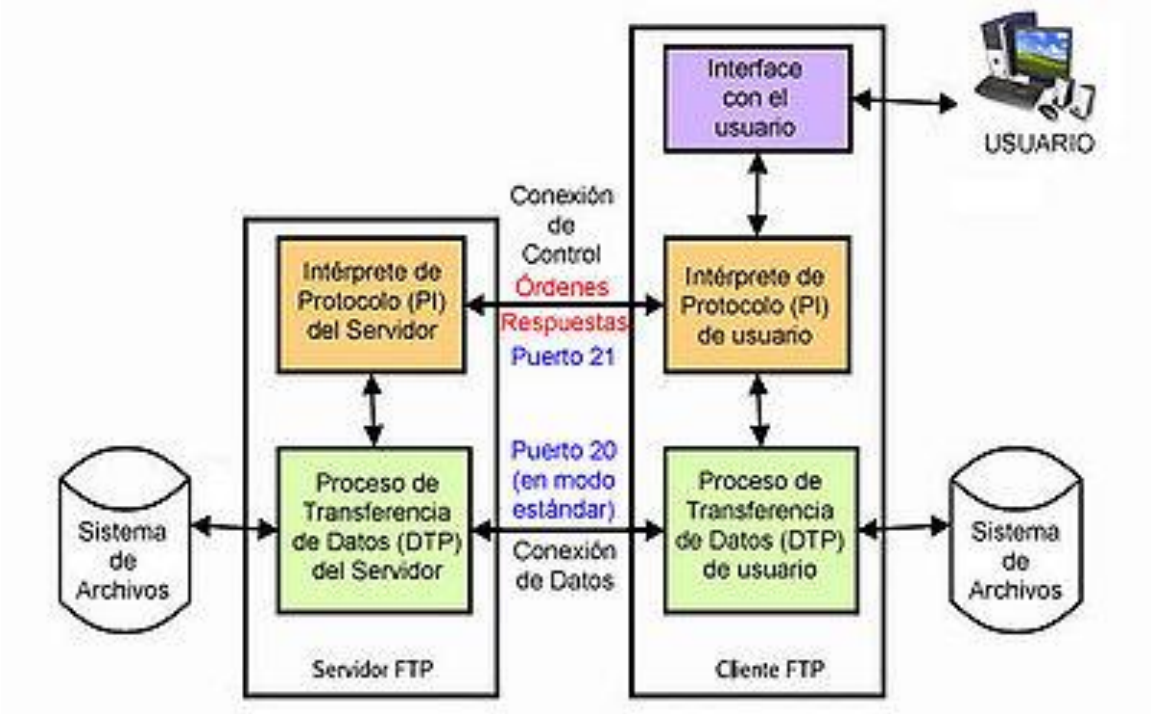


Figura 1.1 Diagrama de servidor ftp

Por lo general, los programas servidores FTP no se encuentran en las computadoras personales, por lo que un usuario normalmente utilizará el FTP para conectarse remotamente a uno y así intercambiar información con él.

Las aplicaciones más comunes de los servidores FTP suelen ser el alojamiento web, en el que sus clientes utilizan el servicio para subir sus páginas de Internet y sus archivos correspondientes; o como servidor de respaldo (copia de seguridad) de los archivos importantes que pueda tener una empresa. Para ello, existen protocolos de comunicación FTP para que los datos se transmitan cifrados, como el SFTP (Secure File Transfer Protocol).

1.4 Cliente FTP

Cuando un navegador no está equipado con la función FTP, o si se quiere cargar archivos en un sistema remoto, se necesitará utilizar un programa cliente FTP. Un cliente FTP es un programa que se instala en la computadora del usuario, y que emplea el protocolo FTP para conectarse a un servidor FTP y transferir archivos, ya sea para descargarlos o para subirlos.

Para utilizar un cliente FTP, se necesita conocer el nombre del archivo, la computadora en que reside (servidor, en el caso de descarga de archivos), el servidor al que se quiere transferir el archivo (en caso de querer subirlo al servidor), y la carpeta o directorio en la que se encuentra.

Algunos clientes de FTP básicos en modo consola vienen integrados en los sistemas operativos, incluyendo Microsoft Windows, DOS, GNU/Linux y Unix. Sin embargo, hay disponibles clientes con opciones añadidas e interfaz gráfica. Aunque muchos navegadores tienen ya integrado FTP, es más confiable a la hora de conectarse con servidores FTP no anónimos utilizar un programa cliente.

1.5 Acceso anónimo

Los servidores FTP anónimos ofrecen sus servicios libremente a todos los usuarios, permiten acceder a sus archivos sin necesidad de tener una cuenta de usuario. Es la manera más cómoda fuera del servicio web de permitir que todo el mundo tenga acceso a cierta información sin que para ello el administrador de un sistema tenga que crear una cuenta para cada usuario.

Si un servidor posee servicio ‘FTP anónimo’ solamente con proporcionar el nombre de usuario "anonymous", cuando pregunte por tu usuario tendrás acceso a ese sistema. No se necesita ninguna contraseña preestablecida, aunque tendrás que introducir una sólo para ese momento, normalmente se suele utilizar la dirección de correo electrónico propia.

Solamente con eso se consigue acceso a los archivos del FTP, aunque con menos privilegios que un usuario normal. Normalmente solo se podrá leer y copiar los archivos existentes, pero no modificarse ni crear otros nuevos.

Normalmente, se utiliza un servidor FTP anónimo para depositar archivos de gran tamaño que no tienen utilidad si no son transferidos completamente a la máquina del usuario, como por ejemplo programas o aplicaciones. Por el contrario, los servidores de páginas web almacenan información destinada a la lectura en línea y no se requiere que la información sea transferida a la máquina del usuario.

1.6 Acceso de usuario

Si se desea tener privilegios de acceso a cualquier parte del sistema de archivos del servidor FTP, de modificación de archivos existentes, y de posibilidad de subir archivos, generalmente se suele realizar mediante una cuenta de usuario. En el servidor se guarda la información de las distintas cuentas de usuario que pueden acceder a él, de manera que para iniciar una sesión FTP se debe introducir un nombre de usuario y una contraseña.

1.7 Jaula FTP (chroot ftp)

Chroot es una llamada al sistema en UNIX que permite configurar un directorio como "raíz" del sistema de archivos para un proceso y sus hijos. En otras palabras, permite configurar el sistema de forma tal que se puedan lanzar procesos confinados dentro de un determinado directorio. Para ellos, dicho directorio será el "/" (la raíz). Cualquier archivo o directorio que esté fuera de la jaula será inaccesible.

Es posible implementar esta función en los servidores FTP con el objetivo que los clientes puedan navegar dentro de un directorio definido y no fuera de él. La implementación depende del servidor FTP, en algunos basta con editar un archivo de configuración y en otros es necesario utilizar comandos del sistema operativo.

1.8 Cliente FTP basado en Web

Un "cliente FTP basado en Web" no es más que un Cliente FTP al cual se puede acceder a través del navegador Web sin necesidad de tener otra aplicación para ello. El usuario accede a un servidor web (http) que lista los contenidos de un servidor ftp. El usuario se conecta mediante http a un servidor web, y el servidor web se conecta mediante el protocolo ftp al servidor ftp. El servidor web actúa de intermediario haciendo pasar la información desde el servidor ftp en los puertos 20 y 21 hacia el puerto 80 http que ve el usuario.

Al disponer de un Cliente FTP basado en Web se puede acceder al servidor FTP remoto, como si se estuviera realizando cualquier otro tipo de navegación Web. A través de un Cliente FTP basado en Web se puede, crear, copiar, renombrar y eliminar archivos y directorios, cambiar permisos, editar, ver, subir y descargar archivos, así como cualquier otra función del protocolo FTP que el servidor FTP remoto permita.

1.9 Acceso de invitado (guest)

El acceso sin restricciones al servidor que proporcionan las cuentas de usuario implica problemas de seguridad, lo que ha dado lugar a un tercer tipo de acceso FTP denominado invitado (guest), que se puede contemplar como una mezcla de los dos anteriores.

La idea de este mecanismo es la siguiente: se trata de permitir que cada usuario conecte a la máquina mediante su login y su password, pero evitando que tenga acceso a partes del sistema de archivos que no necesita para realizar su trabajo, de esta forma accederá a un entorno restringido, algo muy similar a lo que sucede en los accesos anónimos, pero con más privilegios.

1.10 Modos de conexión del cliente FTP

FTP admite dos modos de conexión del cliente. Estos modos se denominan Activo (o Estándar, o PORT, debido a que el cliente envía comandos tipo PORT al servidor por el canal de control al establecer la conexión) y Pasivo (o PASV, porque en este caso envía comandos tipo PASV). Tanto en el modo Activo como en el modo Pasivo, el cliente establece una conexión con el servidor mediante el puerto 21, que establece el canal de control.

1.11 Modo Activo

En modo Activo, el servidor siempre crea el canal de datos en su puerto 20, mientras que en el lado del cliente el canal de datos se asocia a un puerto aleatorio mayor que el 1024 (Véase figura 1.2). Para ello, el cliente manda un comando PORT al servidor por el canal de control indicándole ese número de puerto, de manera que el servidor pueda abrirle una conexión de datos por donde se transferirán los archivos y los listados, en el puerto especificado.

Lo anterior tiene un grave problema de seguridad, y es que la máquina cliente debe estar dispuesta a aceptar cualquier conexión de entrada en un puerto superior al 1024, con los problemas que ello implica si tenemos el equipo conectado a una red insegura como Internet. De hecho, los firewall que se instalen en el equipo para evitar ataques seguramente rechazarán esas conexiones aleatorias. Para solucionar esto se desarrolló el modo Pasivo.

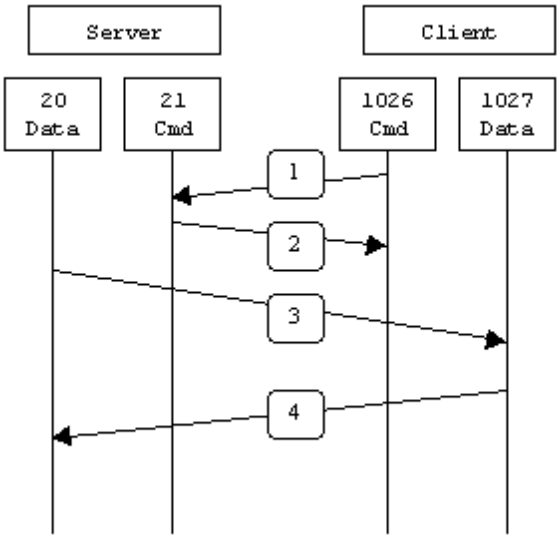


Figura 1.2 Modo de conexión ftp activo

1.12 Modo Pasivo

Cuando el cliente envía un comando PASV sobre el canal de control, el servidor FTP le indica por el canal de control, el puerto (mayor a 1023 del servidor, ej. 2040) al que debe conectarse el cliente. El cliente inicia una conexión desde el puerto siguiente al puerto de control (ej. 1036) hacia el puerto del servidor especificado anteriormente (ej. 2040). Lo anterior se muestra en la figura 1.3.

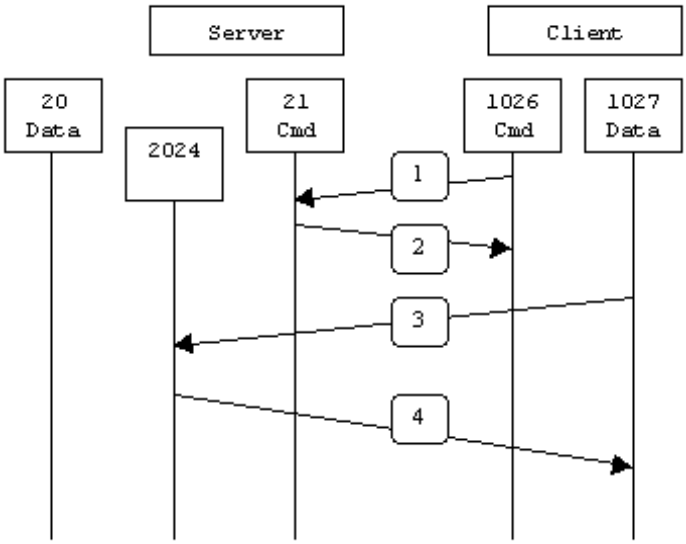


Figura 1.3 Modo de conexión ftp pasivo

Antes de cada nueva transferencia, tanto en el modo Activo como en el Pasivo, el cliente debe enviar otra vez un comando de control (PORT o PASV, según el modo en el que haya conectado), y el servidor recibirá esa conexión de datos en un nuevo puerto aleatorio (si está en modo pasivo) o por el puerto 20 (si está en modo activo) [5].

1.13 Explotando las vulnerabilidades de FTP

Como se mencionó anteriormente, ftp es un protocolo de transferencia de archivos muy eficiente en cuanto a funcionalidad, pero carece de seguridad poniendo en riesgo la información que se transmite a través de la red.

Para que se pueda hacer uso de ftp, la cuenta de usuario debe ser creada en el sistema operativo, generalmente sistemas UNIX, asignarle una contraseña y otorgarle permisos en ciertos directorios para almacenar y/o descargar archivos del servidor.

En la figura 1.4 se observa una conexión por ftp desde un equipo Windows hacia el servidor de archivos UNIX transfer.server.com utilizando la cuenta finanzas:

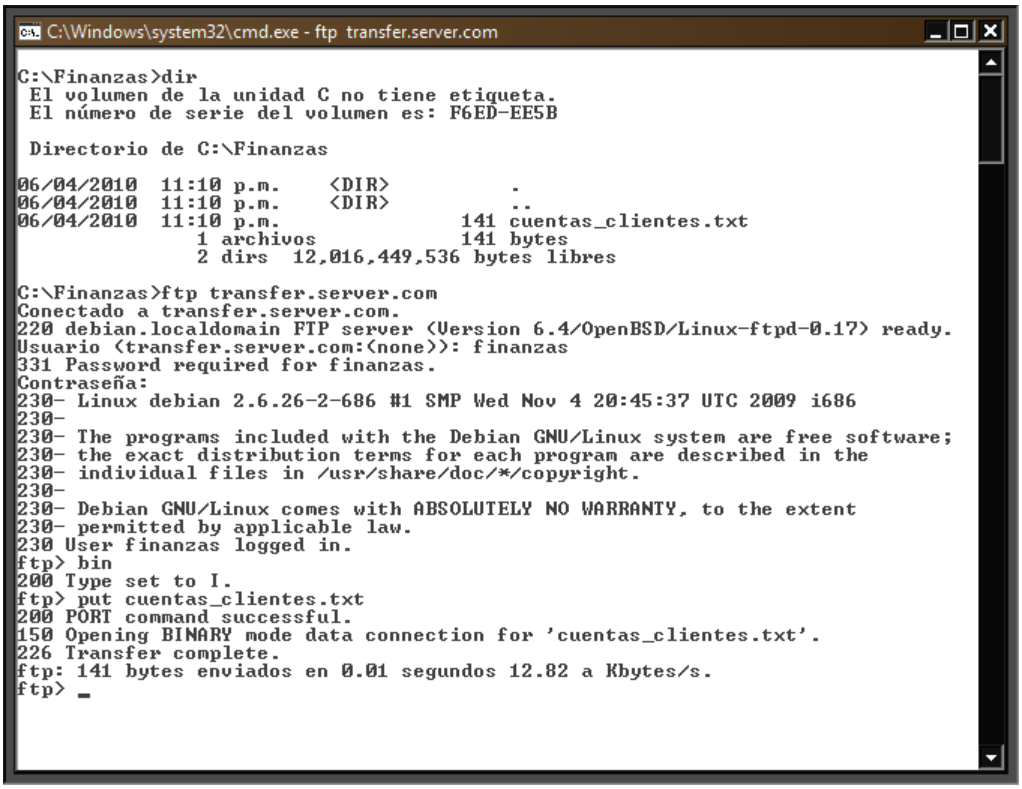


Figura 1.4 Conexión ftp entre un sistema Windows y un servidor Unix

Al mismo tiempo, desde otro equipo se instaló y activó el sniffer tcpdump para capturar el tráfico de la sesión ftp (puerto 21 para sesión y puerto 20 para los datos).

Como puede observarse en la salida del sniffer (Véase figura 1.5), la cuenta utilizada es finanzas y la contraseña es P4ssw0rd:

```
debian:~# uname -a
Linux debian 2.6.26-2-686 #1 SMP Wed Nov 4 20:45:37 UTC 2009 i686 GNU/Linux
debian:~# tcpdump -A port ftp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
01:19:36.934703 IP 192.168.209.1.57135 > transfer.server.com.ftp: S 1277518083:1277518083(0) win
E..4..@....[...../..L%a..... ..............
01:19:42.545819 IP transfer.server.com.ftp > 192.168.209.1.57135: S 1696433317:1696433317(0) ack
E..4..@.@...../e...L%a.....)..............
01:19:42.546019 IP 192.168.209.1.57135 > transfer.server.com.ftp: . ack 1 win 2048
..........f...../..L%a.e...P...
01:19:42.548231 IP transfer.server.com.ftp > 192.168.209.1.57135: P 1:81(80) ack 1 win 365
E..x..@.@."~...../e...L%a.P..mR...220 debian.localdomain FTP server (Version
01:19:42.739618 IP 192.168.209.1.57135 > transfer.server.com.ftp: . ack 81 win 2028
..........d...../..L%a.e...P...
01:19:47.131572 IP 192.168.209.1.57135 > transfer.server.com.ftp: P 1:16(15) ack 81 win 2028
E..7..@....R...../..L%a.e...P.....USER finanzas

01:19:47.131817 IP transfer.server.com.ftp > 192.168.209.1.57135: . ack 16 win 365
E..(..@.@."...../e...L%a.P..m....
01:19:47.171508 IP transfer.server.com.ftp > 192.168.209.1.57135: P 81:118(37) ack 16 win 365
E..M..@.@."...../e...L%a.P..mz...331 Password required for finanzas.

01:19:47.367385 IP 192.168.209.1.57135 > transfer.server.com.ftp: . ack 118 win 2018
c..........._...../..L%a.e...P...
01:19:51.909195 IP 192.168.209.1.57135 > transfer.server.com.ftp: P 16:31(15) ack 118 win 2018
E..7..@....N...../..L%a.e...P.....PASS P4ssw0rd

01:19:51.910061 IP transfer.server.com.ftp > 192.168.209.1.57135: . ack 31 win 365
E..(..@.@."...../e...L%a"P..m....
01:19:51.916253 IP transfer.server.com.ftp > 192.168.209.1.57135: P 118:190(72) ack 31 win 365
E..p..@.@."...../e...L%a"P..m....230- Linux debian 2.6.26-2-686 #1 SMP Wed
01:19:52.114833 IP 192.168.209.1.57135 > transfer.server.com.ftp: . ack 190 win 2000
..........[...../..L%a"e..cP...
01:19:52.115658 IP transfer.server.com.ftp > 192.168.209.1.57135: P 190:548(358) ack 31 win 365
```

Figura 1.5 Captura de tráfico ftp utilizando un *sniffer*

Además es posible obtener información sobre el contenido del archivo, como se muestra en la figura 1.6:

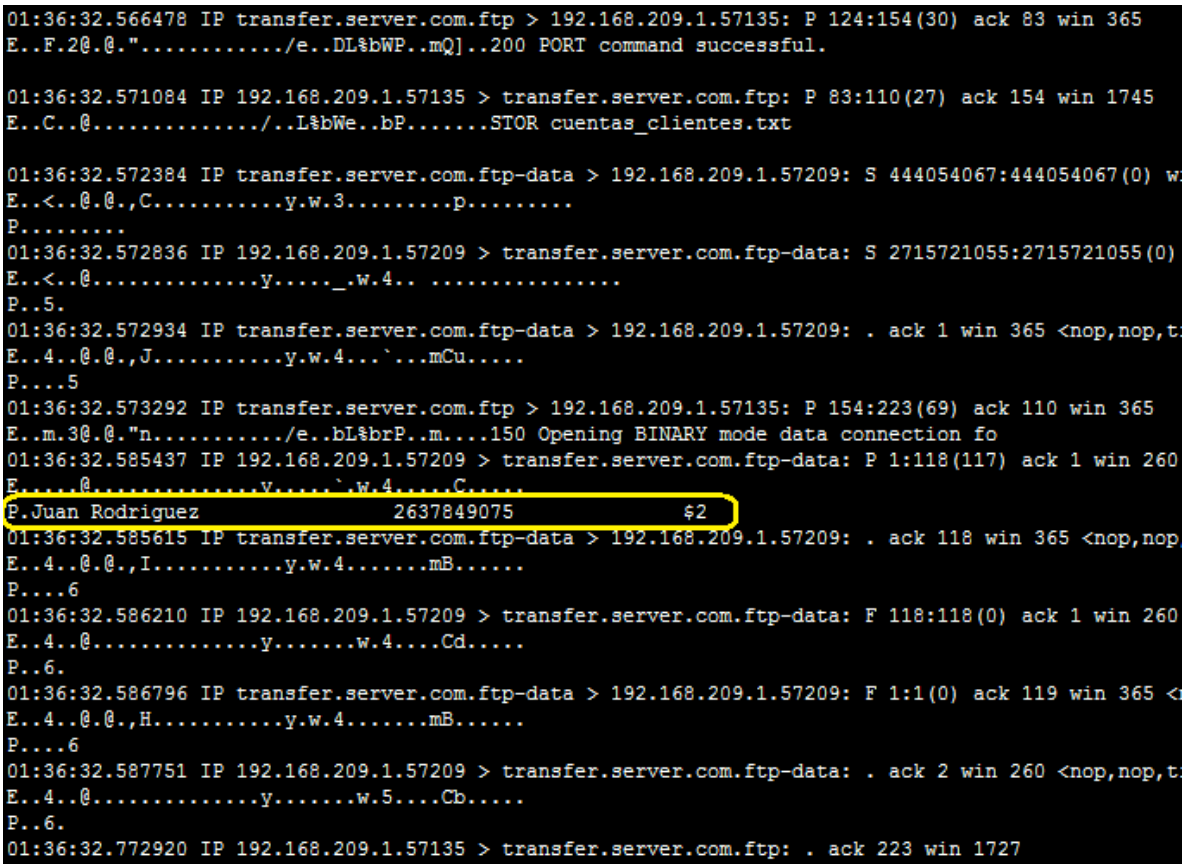


Figura 1.6 Captura del contenido del archivo

Esta técnica es muy utilizada por los atacantes para obtener información confidencial ya que ftp es un servicio muy utilizado. En esta ocasión se pudo capturar el nombre y el password del usuario en el servidor Unix. Con estas credenciales el atacante podría acceder directamente al sistema y robarse los archivos además de información sensible.

En el caso del servidor de un banco esta información podría contener datos confidenciales de los clientes como son datos personales, números de cuenta, números de Seguro Social, datos de las empresas, saldos, información de auditoría, estrategias de negocio etc., para posteriormente transferir los fondos a cuentas propias, robo de identidad, extorsión, vender la información a los competidores.

Dependiendo de las acciones del atacante, la confidencialidad, integridad y disponibilidad de la información son violadas al utilizarse ftp. Por eso los estándares

de seguridad de la información no recomiendan la utilización de este servicio de red, a menos que cuente con otros complementos que protejan los datos transferidos.

1.14 Criptografía

Actualmente, la criptografía es probablemente una de las ciencias con mayor crecimiento dada su gran importancia en el campo de la seguridad informática. Sobre todo porque aún cuando esta ciencia nace desde que el hombre tiene la necesidad de ocultar y mantener segura la información que considera secreta o delicada, es apenas en el siglo pasado que el desarrollo de la computación y las tecnologías relacionadas con la transmisión de datos, las redes, y en sí, el proceso, la transmisión y almacenamiento de información, tienen un crecimiento vertiginoso que conjuntamente se incrementó no sólo con la necesidad de mantener segura la información sino también con las técnicas asociadas a su protección.

La criptografía es la ciencia encargada de transformar la información de manera tal que ésta quede encubierta y sea incomprensible para todo aquel que no tenga la autorización correspondiente para acceder a ella. Sus principales objetivos son resguardar la confidencialidad de los datos y la autenticidad del par remitente/emisor, y a todos aquellos que se dedican a estudiar y desarrollar métodos para resguardar así la información se les llama criptógrafos.

En contraparte, los “curiosos” que no están autorizados para conocer información confidencial, pero que están deseosos de acceder a ella y están dispuestos a hacer lo que sea necesario para descubrirlo son contar con la autorización correspondiente conforman el grupo de los criptoanalistas, así, el criptoanálisis es la disciplina encargada del estudio de los métodos para romper los mecanismos de cifrado hechos para que de esta manera, sin necesidad de conocer clave alguna, sea posible obtener o en su defecto interpretar ilícitamente aquella información cifrada anteriormente.

Las tres piezas principales del proceso de cifrado son los algoritmos, las llaves y los mensajes que se quieren proteger. Los algoritmos son un conjunto finito y ordenado de operaciones que combina la llave con la información a fin de ocultarla y protegerla.

Una clave o llave es una cadena o serie de signos que pueden ser de carácter alfanumérico conocidos solamente por el emisor y/o receptor para resguardar la información. (En la figura 1.12 se muestra un ejemplo de llave privada.)

En la figura 1.7 se muestran los componentes de un sistema criptográfico.

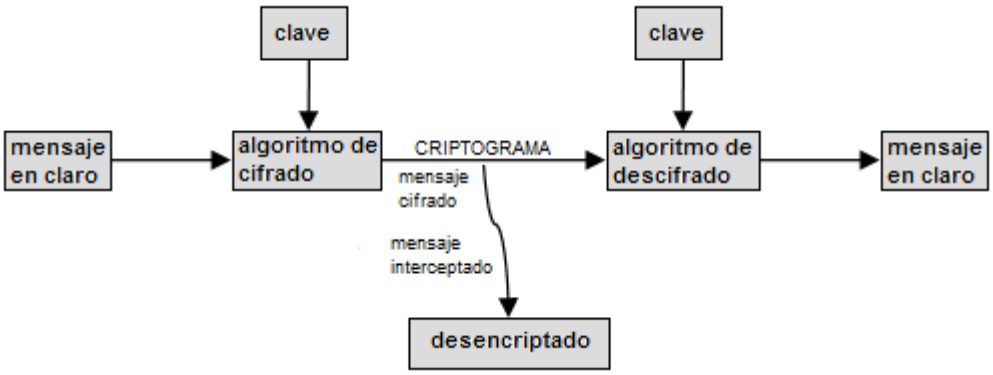


Figura 1.7 Sistema criptográfico

Los sistemas de cifrado pueden ser simétricos, los cuales utilizan llaves simétricas (también llamadas llaves secretas) o asimétricos, los cuales usan llaves asimétricas (conocidas como llaves pública y privada).

1.14.1 Sistemas simétricos

Los sistemas de cifrado simétrico son aquellos que hacen uso ya sea de la misma clave o mismo juego de claves para realizar tanto el proceso de cifrado como el proceso inverso, esto es, el proceso de descifrado (obsérvese figura 1.8), por lo tanto, se hace obvio que esa clave o juego de claves deben ser celosamente resguardadas, permanecer en secreto y cuidar que no sean divulgadas o descubiertas por entidades no autorizadas.

Los principales algoritmos simétricos son: IDEA (International Data Encryption Algorithm), Blowfish, RC5 (Rivest Cipher), DES (Data Encryption Standard), 3DES, y AES (Advanced Data Encryption Standard).

Figura 1.8 Criptografía simétrica

1.14.2 Criptografía asimétrica

A diferencia de la criptografía simétrica que solamente requiere de una clave tanto para cifrar como para descifrar, la criptografía asimétrica hace uso de un par de claves relacionadas matemáticamente entre sí, una que se emplea para cifrar y otra para descifrar.

Los algoritmos asimétricos y su efectividad se basan en lo que se ha dado por llamar funciones de un solo sentido, las cuales corresponden al término *trapdoor one-way* que denota precisamente esta idea, funciones fáciles de calcular, pero que por el contrario requieren una gran potencia para calcular el proceso inverso, de manera que se vuelven sumamente difíciles (prácticamente imposibles) de invertir, funciones a las cuales también se les suele llamar *funciones unidireccionales*.

Con un algoritmo de este tipo cada usuario necesita un par de claves, de las cuales una debe conservarse como privada (siempre secreta) y la otra como pública (siempre visible), y no se requiere ningún canal seguro para el intercambio (Véase figura 1.9). Por esta situación también se le conoce como “algoritmos de clave pública” y mientras la clave privada permanezca en secreto, la clave pública puede estar visible durante mucho tiempo sin poner en riesgo la seguridad de los datos intercambiados a través de estos algoritmos.

Los principales algoritmos asimétricos son: Diffie-Hellman, ElGamal, RSA (Rivest-Shamir-Adleman) y DSA (Digital Signature Algoritm) [2].

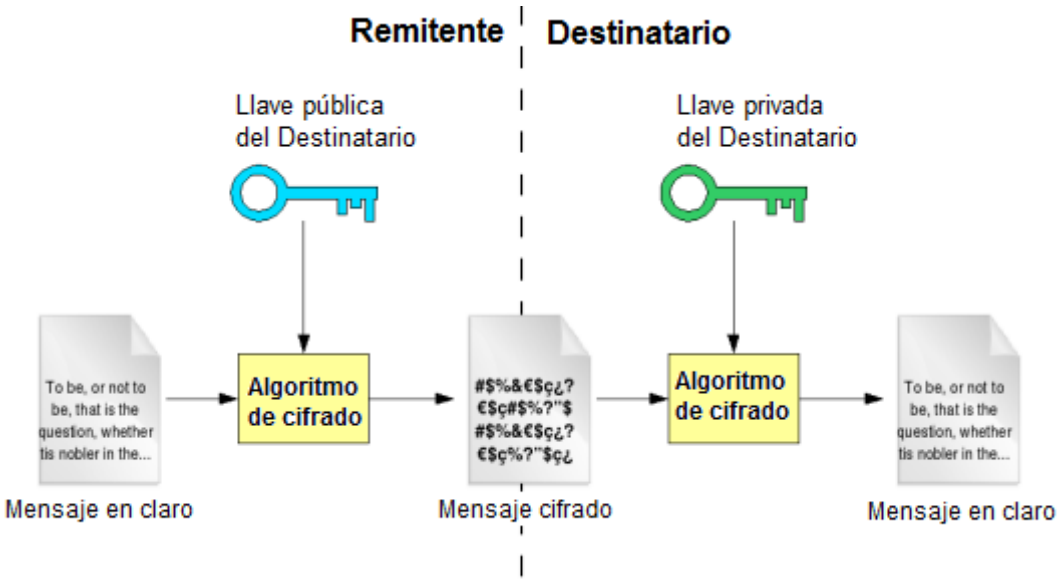


Figura 1.9 Criptografía asimétrica

Dentro de la criptografía asimétrica se tienen algunas implementaciones de seguridad que a continuación se describen:

a) Infraestructura de Llave Pública (PKI)

Dentro de la criptografía asimétrica, para obtener las claves públicas de manera segura se puede utilizar una infraestructura de clave pública.

PKI se refiere a un subsistema que proporciona manejo de certificados digitales. Los certificados digitales contienen una llave pública y una privada que se utilizan para el cifrado de los archivos, firmas digitales y seguridad en la autenticación.

La empresa dedicada a implementar soluciones de seguridad informática, Entrust, ha creado sus propios certificados basados en infraestructura PKI.

Los certificados se almacenan en un servidor alterno y son verificados utilizando una “cadena de validación”. Cuando los datos son correctos, el tráfico sftp es cifrado con la llave pública del destinatario, el cual utilizará su llave privada para obtener la información en claro [6].

b) The Gnu Privacy Guard (GPG)

PGP es el nombre de un programa de cifrado asimétrico creado en 1991 por Philip Zimmerman. Desde entonces, PGP se ha convertido en el modelo dominante del software de cifrado personal.

En julio de 1998, PGP Inc. propuso un estándar OpenPGP a la IETF. El objetivo de este estándar es el de proporcionar la guía para la creación de software de cifrado personal, y el término OpenPGP podría ser aplicado a cualquier programa que cumpliera con el estándar. La IETF aceptó la propuesta y formó en grupo de trabajo OpenPGP Working Group para desarrollar el estándar.

El RFC 2440 describe la especificación actual OpenPGP; su sucesor es el RFC 4880 y ha sido hecha la propuesta del estándar OpenPGP.

El software más popular de la especificación OpenPGP es GnuPG. Está disponible para sistemas Linux, sistemas de código abierto Unix como BSD y los más modernos sistemas comerciales Unix como Sun Solaris. GnuPG (comúnmente referido como GPG) ha sido portado a los sistemas operativos Mac OS y Windows [7].

c) Utilizando GPG

Lo primero que se necesita es generar un par de llaves [8]. OpenPGP es un sistema criptográfico basado en infraestructura de llave pública, por lo que se requiere un par de llaves - pública y privada - para el cifrado y

descifrado de los datos. En la figura 1.10 se muestra un ejemplo de creación de llaves GPG:

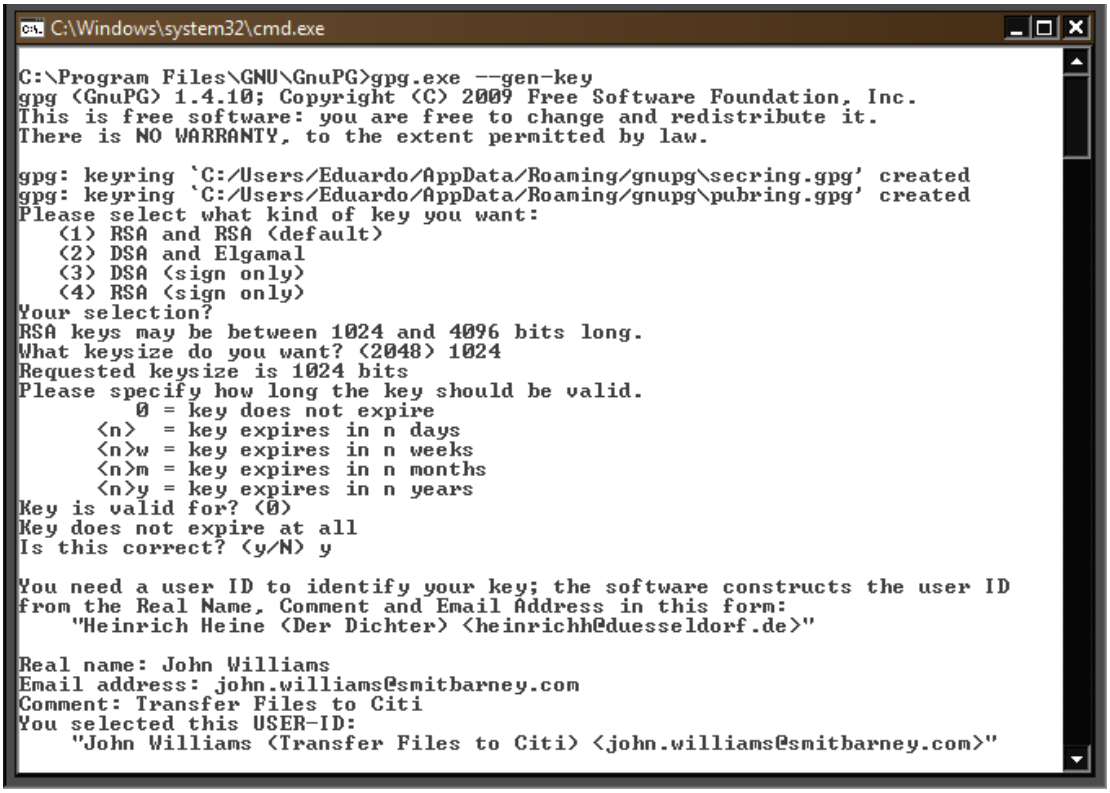


Figura 1.10 Creación de llaves GPG

Cuando se quiere cifrar algo es necesaria la llave privada.

La llave privada es más que un conjunto de contraseñas, es una cadena pseudoleatoria de caracteres (Véase figura 1.11). Un “llavero” GPG es un servidor dónde se almacenan y mantienen todas las llaves públicas.

Figura 1.11 Ejemplo de una llave privada GPG

Para que otros puedan cifrar y descifrar datos, ellos necesitan conocer las llaves públicas de los demás. En la siguiente figura se muestra el ejemplo de una llave pública:

```
lirry@debian:~$ gpg --export -a "Gustavo Palma Avila" > public.key
lirry@debian:~$ cat public.key
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.4.9 (GNU/Linux)

mQGiBElOuZ0RBACYjWqapJvKSsUQTzUetYsmHi0oanT+WC5mNqrMv2VtYzgOK364
3IWAvpWx7XRbXfNz5rhZAdv8pT04EQUdIMkmZ/jw54rXUFpZidPL5b8ku9VOxnY0
sZLhed5UL+jUr5KuPBo63xYBubSS5SkgBAa4dVDLrbzNG56aJR+FFIaPCwCgpjuT
uz6GNjaOsR8bBNZ44fN1yD8D/0litrDISbCR/sqhiliJ9yuWiL5RadVRFWT6U90k
RNWRir5Xd/ra3e7kpX5TBJptpwDHbKbsCO+rV8vYGuvi0Zkl6Rl37pEirtY0AurH
O5nALmYIqURjolxpBQMuqUH69OI2zD+89ZfixCPm1zYfcZM+/AMkNhfxUz9y4XHn
euVDBACFq2YoXyAWskEbvO2B2VtMaSUCxVrYCLQaFoziw78JEjrZ+NvgpQxeYGrd
0cWRTVH9yJBqIcQbOlM3YMCudiUJmD6XdPknJ72+fVJEghqx/nhQ6m5s8oB6ouny
71L9yfThBuD3niw58VpUrhZ2uSom7NRDtK0Q2DSTzEGWc5FTOrRKR3VzdGF2byBQ
YWxtYSBBdmlsYSAoTGxhdmUgZGUgdHJhbnNmZXJlbmNpYSkgPGd1c3Rhdm8ucGFs
bWFhdmlsYUBjaXRpLmNvbT6IYAQTEQIAIAUCSU65nQIbAwYLCQgHAwIEFQIIAwQW
AgMBAh4BAheAAAoJEG+Cfg5gJwr/6uQAn3Ma1OtXXVvm00cpaQYzpquiOWl+AJ9m
WzmQ9gLH3oA2Kdx6lwqeQlme0bkBDQRJTrmdEAQAqtA764eQL26x6Lc+0vDGhPBR
Hgy1Wg27Wlslz6JEcXsFcGT+mSxHKFlcvbGpY3t/FmITF2LmfQX48Kbo736d3pjP
onhplHvt0alYkHtcPhcWKHSrnV6+LSce8OjLqwPg1NBUlyBdpuZODscdfZlCJrhW
hDa+cTfF5jCEz5p1FX8AAwUD/3aNv2YF4nnhb1af9Ohh1YjwPTasq1IByyxeWB6R
6UGIIqLm27v6WCR6lf1tmO40FBajd1+bZYrD+8S5d8VPXHEO9yuRmtEEGTrEgdx1
62YN0Pgzq7yoA+1e9ddWKpd/KyY99eQH/m7lx4pKADYUyRYIOLEyenJkiFlkVDh7
w7y5iEkEGBECAAkFAklOuZ0CGwwACgkQb4J+DmAnCv9UZQCfUHBX4eu4d7jEk8rD
IJCSV8wjzd0AnjSRsdzknK7VV5TiQ3jrheC7MZGz
=Gi+E
-----END PGP PUBLIC KEY BLOCK-----
lirry@debian:~$
```

Figura 1.12 Ejemplo de una llave pública GPG

Una vez teniendo esto ya es posible cifrar y descifrar archivos. En el siguiente ejemplo el usuario **John Williams** de la empresa Smith Barney va a cifrar el archivo clients_smitbarney.txt utilizando la llave pública del empleado de Citigroup, **Eduardo Palma Avila**. El archivo generado es *clients_smitbarney.txt.gpg*, como se muestra en la figura 1.13:

Figura 1.13 Cifrado de un archivo por GPG

Para descifrar el archivo es necesaria la llave privada de Eduardo. Si John intenta descifrar los datos, obtiene el siguiente mensaje de error (Figura 1.14):

Figura 1.14 Mensaje de error al descifrar un archivo sin la llave privada

En cambio, del lado de *Citigroup*, debido a que se posee la llave privada es posible descifrar sin ningún problema y leer el contenido del archivo (Véase figura 1.15):

```
finanzas@debian:~$ ls -la clients_smitbarney.txt.gpg
-rw-r--r-- 1 finanzas finanzas 502 abr 10  2010 clients_smitbarney.txt.gpg
finanzas@debian:~$
finanzas@debian:~$ gpg --list-secret-keys
/home/finanzas/.gnupg/secring.gpg
---------------------------------
sec   1024D/51C63357 2008-12-21
uid                  Eduardo Palma Avila (Llave de Citi) <eduardo.palmaavila@citi.com>
ssb   1024g/D39B2CAC 2008-12-21

finanzas@debian:~$ gpg -d clients_smitbarney.txt.gpg
gpg: cifrado con clave ELG-E de 1024 bits, ID D39B2CAC, creada el 2008-12-21
      "Eduardo Palma Avila (Llave de Citi) <eduardo.palmaavila@citi.com>"
John Terry              2637849075              $250,000
Jason Chan              2745905278              $1,154,893
Ka Hong Cheu            2863491078              $1,638,940
David Velazquez         2935372173              $635,540
Brian Rennie            2943628263              $4,638,940finanzas@debian:~$
finanzas@debian:~$
```

Figura 1.15 Descifrado de un archivo

1.15 Protocolo SSH SFTP

El protocolo SSH SFTP proporciona la funcionalidad para tener acceso, transferir y administrar archivos sobre un flujo de datos confiable ya que todo el tráfico viaja cifrado evitando que los atacantes puedan capturar datos utilizando programas interceptores de paquetes (sniffers), como se observó anteriormente.

Fue diseñado por el grupo Internet Engineering Task Force (IETF) como una extensión del protocolo Secure Shell (SSH) versión 2.0 para proporcionar la seguridad en la transferencia de archivos, pero es también compatible con otras aplicaciones como lo son TLS (Transport Layer Security) y VPN. IETF señala que aunque SFTP es parte de SSH-2, este protocolo es independiente del resto de la suite SSH-2.

Este protocolo asume que la información viaja sobre un canal seguro, tal como lo hace SSH, que el servidor ya ha autenticado al cliente, y que la identidad del cliente está disponible para el protocolo.

Comparado con versiones anteriores del protocolo SCP, el cual sólo permite transferencia de archivos, el protocolo SFTP permite algunas operaciones en los archivos remotos – es más como un protocolo de sistema de archivos. Algunas de estas operaciones son: continuar la transferencia que ha sido interrumpida, listado de directorios, y borrado de archivos de manera remota.

SFTP no es FTP ejecutándose sobre SSH, en lugar de ello es un nuevo protocolo diseñado por el mismo grupo de trabajo IETF SECSH. Algunas veces se confunde con el protocolo SMTP (Simple File Transfer Protocol).

Este protocolo por sí mismo no proporciona autenticación y seguridad; algún otro protocolo se encarga de asegurar estas características. SFTP es más frecuentemente utilizado como un subsistema del protocolo SSH versión 2 (sftp2.exe - como se realizó en el ejemplo mostrado anteriormente). También es posible ejecutarlo sobre SSH-1 y otros flujos de datos.

Para el envío, los archivos transferidos pueden estar asociados con sus atributos básicos, como lo son las marcas de tiempo. Esta es una ventaja común sobre FTP, el cual no proporciona la fecha y hora originales del envío.

La tabla 1.1 muestra las diferencias entre los protocolos ftp y sftp:

Tabla 1.1 – Diferencias entre ftp y sftp

ftp	sftp
El tráfico de sesión y datos viaja en claro lo que hace posible que la información pueda ser capturada por un atacante.	Todo el tráfico viaja cifrado a través de un canal SSH, lo que evita que los datos puedan ser interceptados.
Sólo permite el intercambio de archivos entre el cliente y el servidor.	Permite algunas operaciones sobre los archivos como lo son: continuar la transferencia que ha sido interrumpida, listado de directorios, y borrado de archivos de manera remota.
FTP es accesible anónimamente, utilizando cuentas que no existen en el sistema operativo.	Sólo es posible utilizarlo con cuentas existentes en el sistema operativo.
Protocolo eficiente para transferir archivos, incluso utilizando proxies.	El control de tráfico es ineficiente cuando se utilizan proxies tradicionales.
Velocidad de transmisión de archivos rápida.	Velocidad de transmisión de archivos lenta.
Trabaja en los puertos 20 (datos) y 21 (control)	Trabaja en el puerto 22.

Para mostrar algunas de las ventajas de seguridad de sftp sobre ftp Se realizó el siguiente ejemplo, una conexión entre el mismo sistema Windows y Unix, pero utilizando el protocolo sftp (Véase figura 1.16):

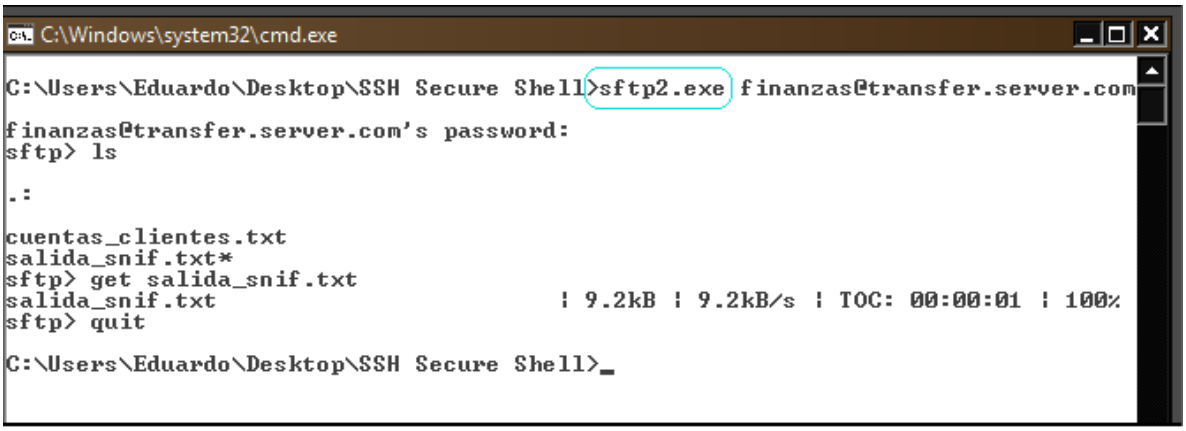


Figura 1.16 Conexión sftp entre un sistema Windows y un servidor Unix

Como se observa en la figura 1.17, el tráfico capturado contiene datos cifrados por lo que no es posible obtener información de la sesión ni del archivo transferido:

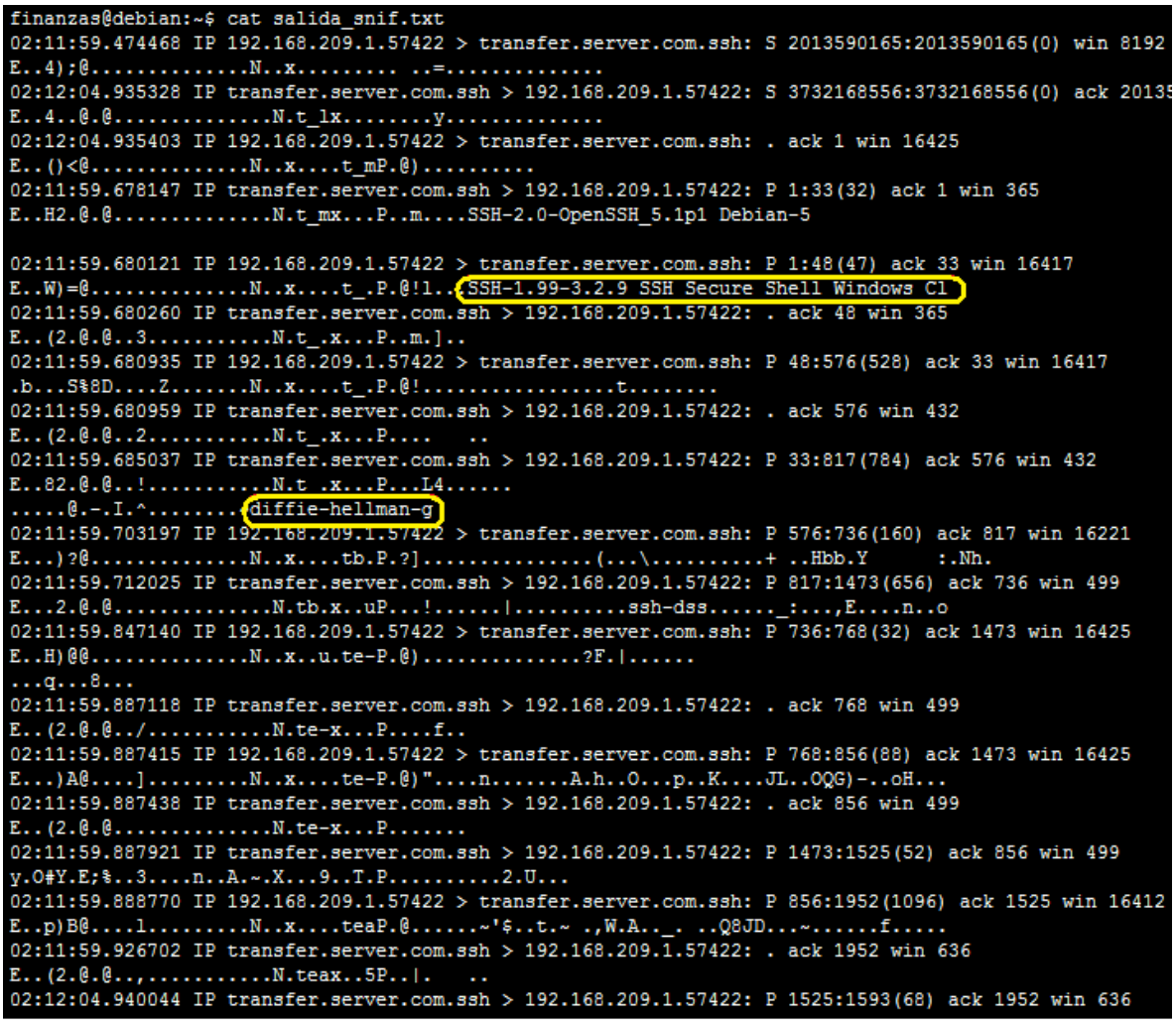


Figura 1.17 Captura de tráfico sftp

Se observan cadenas legibles referentes al protocolo y al intercambio de llaves utilizando el algoritmo Diffie-Hellman. Estas claves se utilizan para cifrar y descifrar la información, por lo que sólo los servidores interconectados pueden obtener los datos.

Por sí mismo, SecureFTP crea un canal seguro al utilizar el protocolo sftp para la transferencia de archivos. Lo que lo hace diferente es que tiene otras opciones que le proporcionan una capa adicional de seguridad: cifrado por Infraestructura PKI Entrust y cifrado por intercambio de llaves GPG; además de que puede notificar vía correo electrónico a las personas indicadas en caso de una transferencia fallida y/o exitosa.

Capítulo 2

Antecedentes del proyecto y definición del problema

SecureFTP es un sistema utilizado para trasferir archivos de una manera segura entre empleados internos y personal de otras instituciones. Utiliza el protocolo sftp, y otras opciones de cifrado de llave asimétrica que le proporcionan mayor seguridad a la transmisión de información.

2.1 Antecedentes del proyecto

A continuación se definen las principales características del sistema SecureFTP, como lo son sus componentes de infraestructura y las rutas que definen la transferencia de los archivos.

2.1.1 Infraestructura

Se tiene un esquema de alta disponibilidad ya que el servidor que resguarda la aplicación posee un servidor espejo en el cual la información se replica una vez que es modificada, con el objetivo de asegurar la disponibilidad requerida. En caso de que algún evento inesperado ocasione la falla del servidor principal, el servidor redundante entra en funcionamiento para garantizar la continuidad de la operación. El servidor espejo se encuentra ubicado a una distancia considerable del servidor principal para evitar que el desastre también afecte al sitio alterno (Véase figura 2.1).

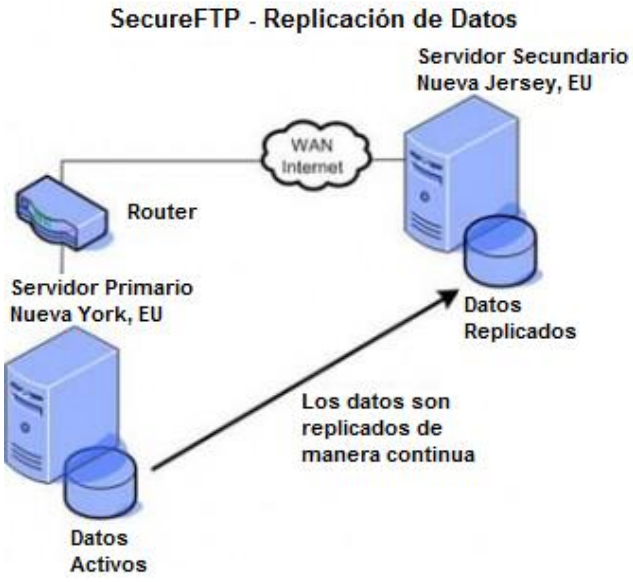


Figura 2.1 SecureFTP - Replicación de datos

2.1.2 Características

La confidencialidad en *SecureFTP* se logra al proporcionar un *login id* (userid), una contraseña a cada usuario del sistema y prefijos de los nombre de archivos que serán transferidos, los cuales son previamente establecidos entre el empleado de Citi y el cliente externo.

La integridad y confidencialidad se logran por medio del cifrado. Esta técnica proporciona un canal seguro, en el cual la información (sesión y datos) viaja cifrada lo que evita pueda ser capturada o que sufra alguna alteración inesperada durante la transmisión por los que se asegura la integridad de la misma.

En la siguiente sección anterior se habló de la Criptografía, la cual ha demostrado ser el método más efectivo para permitir un intercambio de mensajes que sólo puedan ser leídos por personas a las que van dirigidos y que poseen los medios para descifrarlos.

SecureFTP es un sistema que se ejecuta en un sistema operativo UNIX Solaris OS versión 5.8. Los usuarios se conectan a este sistema por medio del protocolo

sftp para subir y descargar los archivos. Esta transferencia se ejecuta a través de un canal seguro y además se tiene la opción para que los usuarios agreguen una capa de protección adicional al utilizar cifrado GPG (GnuPG) a través de intercambio de llaves pública y privada o a través de cifrado Entrust®, utilizando una cadena de validación. Como se mencionó anteriormente, para este tipo de cifrado, es necesario que el cliente proporcione su llave pública GPG para que sea almacenada en el sistema; a su vez, Citigroup debe proporcionar la llave pública para poder recibir archivos.

Las llaves privadas deben almacenarse en un lugar seguro para evitar que sean comprometidas. Es recomendable que se almacenen de manera cifrada y con permisos de lectura sólo para el administrador.

2.1.3 Definición de rutas

Para que una transferencia de archivos pueda llevarse a cabo, el administrador del sistema SecureFTP debe establecer una ruta entre el empleado y el cliente.

Para la creación de una ruta es necesaria la existencia previa de las cuentas de usuario, del cliente y del empleado, en el sistema operativo UNIX.

Para crear las cuentas en necesario levantar una solicitud previamente aprobada por el negocio a los administradores de seguridad. Una vez validada y aprobado el requerimiento se asignan nuevas cuentas de usuario con contraseñas temporales que se deben cambiar la primera vez que utilicen el sistema, cumpliendo así con la política de contraseñas.

Las rutas pueden ser internas y externas: la ruta interna es utilizada por el cliente para enviar un archivo a la red de Citigroup (figura 2.2). En la ruta externa ocurre lo contrario, el empleado envía un archivo, a través de un canal seguro, al cliente de otra compañía (figura 2.3).

Los archivos pueden ser reenviados por medio del protocolo ftp hacia otro servidor utilizando la opción *passon* y los usuarios pueden recibir notificaciones por correo electrónico en caso de falla o éxito en la transmisión.

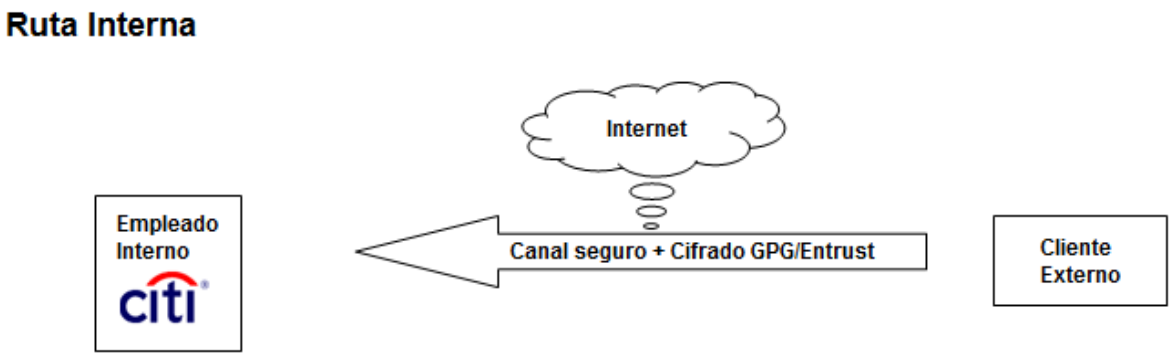


Figura 2.2 Esquema de una ruta interna

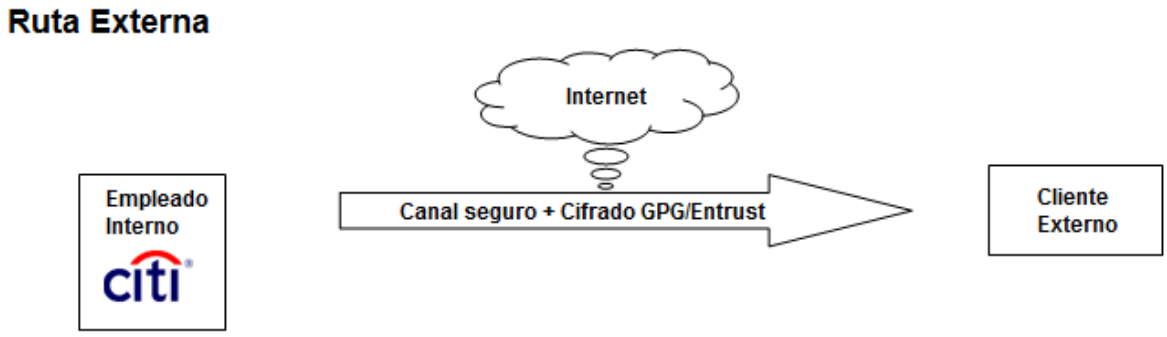


Figura 2.3 Esquema de una ruta externa

Una ruta se compone de los siguientes parámetros:

- Emisor (source)
- Receptor (target)
- Prefijo (prefix)
- Passon (opción de reenvío automático hacia otro servidor utilizando el protocolo de transferencia de archivos ftp)
- Bandera de cifrado (PGP o Entrust®) (encrypt)
- Cadena de autenticación (exclusiva para cifrado por Entrust®) (validation_string)
- Notificación por correo electrónico en caso de una transferencia exitosa (success-email-notification)

- Notificación por correo electrónico en caso de una transferencia fallida (error-email-notification)

La configuración de las rutas se lleva a cabo en el archivo *sftpaccess* de formato ASCII (Véase figura 2.4):

```
lirry@debian:~/sftp/secureftp$ cat sftpaccess
#SecureFTP - Configuration file
#Format:
#source  target  prefix  passon     encryp  entrust success error
citi_emp external_emp docs_out none e_g none none eduardo.palmaavila@citi.com
external_emp citi_emp docs_in passon e_g none none eduardo.palmaavila@citi.com
```

Figura 2.4 Archivo sftpaccess

En este ejemplo se muestran dos rutas entre el empleado *citi_emp* y el cliente *external_emp*, la ruta interna tiene la opción de reenvío (passon) y ambas tienen habilitado el cifrado por PGP. Se enviarán notificaciones de error en el envío a la dirección eduardo.palmaavila@citi.com.

2.2 Definición del problema

Hasta el momento no se cuenta con una interfaz de administración del archivo sftpaccess para la definición y manipulación de rutas en *SecureFTP*.

La administración se realiza de manera manual utilizando el editor de textos UNIX vi.

Este editor está hecho para escribir y modificar archivos de texto sencillos, programas, etc. No permite justificar párrafos, utilizar distintos tipos de letra, escribir a varias columnas, insertar gráficos, etc. Para la manipulación de textos es necesario utilizar comandos como “Shift + y” para copiar una línea, p para pegar, x para borrar un carácter, dd para borrar una línea completa, cc para copiarla, etc.

En general, los usuarios lo consideran demasiado complicado y se requiere especial cuidado para no alterar de manera imprevista el archivo.

Si más de un usuario modifica simultáneamente algún archivo el resultado es inesperado, la corrupción del archivo y, en el caso de *sftpaccess*, el borrado o modificación de cientos de rutas, causando un impacto muy grave a los usuarios de la empresa que tienen contacto directo con otras compañías al enviar y recibir archivos con información altamente sensible.

Otra desventaja es el tiempo. Para la creación de una nueva ruta es necesario copiar una existente y después modificarla utilizando algunos de los comandos descritos anteriormente. Por lo complicado, esto hay que repetirlo varias veces ya que es difícil que la nueva ruta sea insertada al primer intento. El tiempo es muy variable y va desde 4 hasta 10 minutos en el caso de modificación y actualización.

Con una interfaz eficaz y amigable, características principales del sistema de administración propuesto, se planea la reducción de los tiempos a 20 segundos, y se evitan errores por modificación inesperada o por corrupción debido a su uso simultáneo.

Capítulo 3

Desarrollo

Como se mencionó anteriormente, el archivo de configuración sftpaccess tiene formato ASCII por lo que es compatible con scripts UNIX Shell.

Por tal razón se decidió utilizar este lenguaje de programación para manipular la configuración de las rutas.

Otra ventaja es que se pueden utilizar comandos del sistema operativo en caso de que se requiera la interacción con el sistema de archivos o llamadas al sistema [9]. En este caso, cuando se crea el archivo para reenvío de archivos y se elige la opción de prueba, el sistema ejecuta el script en modo debug para que en el caso de alguna falla, ésta pueda detectarse y corregirse fácilmente. Esto se muestra en la figura 3.1.

```
lirry@debian:~/sftp/secureftp$ sh -x passon.morgan_stanley
+ USERS=/export/ftp/users
+ morgan_stanley=/export/ftp/users/morgan_stanley
+ echo 'Dir: /export/ftp/users/morgan_stanley'
Dir: /export/ftp/users/morgan_stanley
+ ls -al /export/ftp/users/morgan_stanley
total 16
drwxr-xr-x 3 login_id users    4096 nov 25 15:49 .
drwxr-xr-x 3 root     root     4096 nov 25 15:43 ..
-rw------- 1 login_id login_id   37 nov 25 15:49 .bash_history
drwxr-xr-x 2 login_id login_id 4096 nov 25 15:49 files
+ echo ''

+ cd /export/ftp/users/morgan_stanley
+ ftp -n -iv transfer.server.com
Connected to transfer.server.com.
220 debian.localdomain FTP server (Version 6.4/OpenBSD/Linux-ftpd-0.17) ready.
331 Password required for login_id.
230- Linux debian 2.6.26-2-686 #1 SMP Wed Nov 4 20:45:37 UTC 2009 i686
230-
230- The programs included with the Debian GNU/Linux system are free software;
230- the exact distribution terms for each program are described in the
230- individual files in /usr/share/doc/*/copyright.
230-
230- Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
230- permitted by applicable law.
230 User login_id logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
257 "/export/ftp/users/morgan_stanley" is current directory.
250 CWD command successful.
257 "/export/ftp/users/morgan_stanley/files" is current directory.
200 Type set to I.
Interactive mode on.
(local-file) (remote-file)
221 Goodbye.
```

Figura 3.1 Ejecución del script para reenvío de archivos

Con el uso de esta opción se utilizan comandos del SO y el sistema de archivos ya que se están copiando archivos a directorios específicos, además de que deben otorgarse privilegios de ejecución al mismo script.

Nota: Para mayor claridad se utilizarán pantallas negras para mostrar el flujo de ejecución del programa y pantallas blancas para mostrar fragmentos de código, ejemplos de archivos y ejecución de instrucciones fuera de la estructura del programa.

3.1 Opciones de Ejecución

Las opciones que tiene la interfaz de administración son las siguientes:

- Crear Ruta – esta opción permite crea una ruta para el envío de archivos.
- Borrar Ruta – esta opción tiene dos modos de operación: borra una ruta en específico (borrado simple) ó todas en las que esté involucrado un usuario (borrado múltiple).
- Modificar Ruta - al elegir esta opción es posible modificar alguno de los 8 parámetros que componen una ruta.

En la figura 3.2 se muestra el diagrama de flujo del programa:

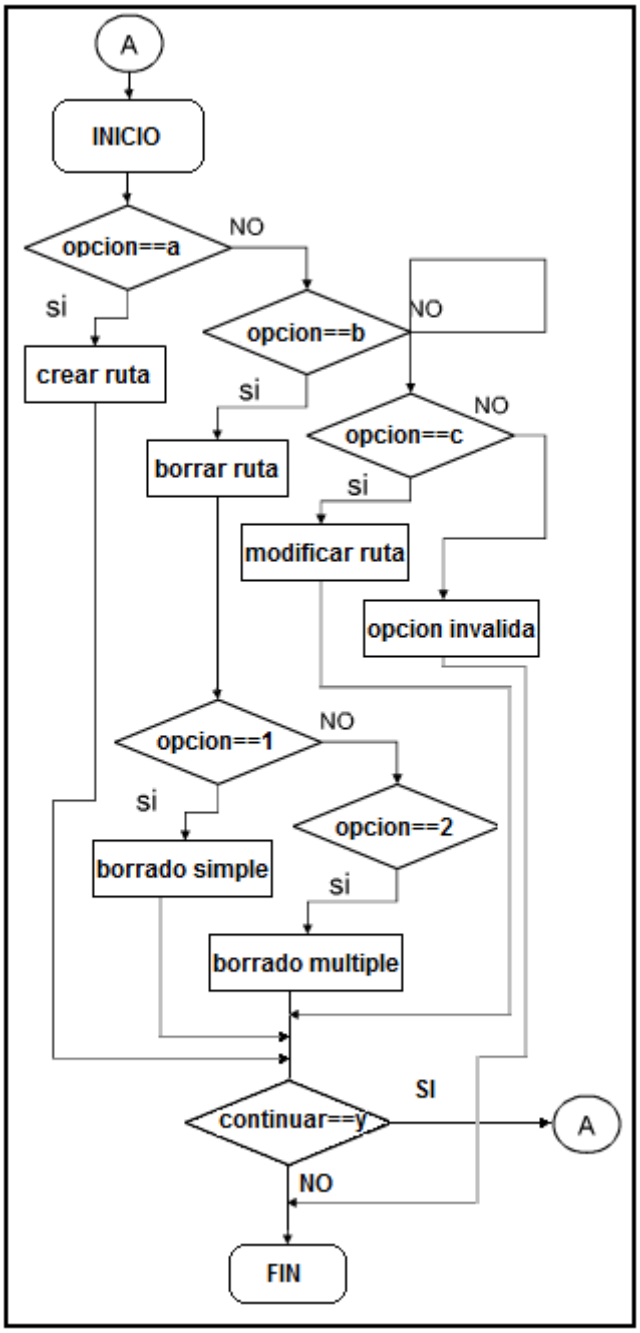


Figura 3.2 Diagrama de flujo de la interfaz de administración

3.1.1 Creación de una Ruta

Para la creación de una ruta son necesarios 8 parámetros (emisor, receptor, prefijo, passon, cifrado, cadena de validación, notificación por correo electrónico en caso de éxito o error en la transferencia).

Si la ruta requiere la opción de reenvío por ftp (passon) es necesario crear un script adicional que requiere el hostname ó dirección IP del servidor, un login Id, contraseña y el directorio en el que será almacenado.

En la figura 3.3 se muestra un ejemplo del script passon:

```
lirry@debian:~/sftp/secureftp$ cat passon.morgan_stanley
#! /bin/sh
#
#        $1 File name to passon

USERS=/export/ftp/users
morgan_stanley=$USERS/morgan_stanley

echo "Dir: $morgan_stanley"; ls -al $morgan_stanley; echo ""
cd $morgan_stanley || { echo "Can't CD to $morgan_stanley - ABORTING !!!"; exit 255; }

ftp -n -iv transfer.server.com  <<-EOF
user login_id p4ssw0rd
        pwd
        cd /home/login_id
        pwd
        bin
        prompt
        put $1
        bye
EOF
lirry@debian:~/sftp/secureftp$
```

Figura 3.3 Script *passon*

En este caso el archivo será reenviado al servidor *transfer.server.com* utilizando protocolo ftp utilizando las opciones elegidas por el administrador:

3.1.1.1 Opciones de cifrado

En cuanto a las opciones de cifrado, éstas se incluyen en un menú de selección:

1.- g_e (Cifrado GPG para ruta interna)

2.- g_d (Cifrado GPG para ruta externa)

3.- d_e (Cifrado Entrust para ruta interna)

4.- e_d (Cifrado Entrust para ruta externa)

La tabla 3.1 muestra la definición de cada una de las opciones de cifrado que incluye el SecureFTP:

Tabla 3.1 – Opciones de cifrado

Valor de la cadena	Descripción de la cadena de cifrado
d_e	Indica que el archivo será enviado del usuario interno al externo y será cifrado utilizando el estándar Entrust®.
e_d	Indica que el archivo encriptado será enviado del usuario externo al interno y será descifrado utilizando el estándar Entrust®.
g_e	Indica que el archivo será enviado del usuario interno al externo y será cifrado utilizando el estándar GPG (GnuPG).
g_d	Indica que el archivo encriptado será enviado por usuario externo al interno y será descifrado utilizando el estándar GPG (GnuPG).
none	Indica que será enviado un archivo en texto plano.

En el caso de cifrado Entrust (d_e ó e_d) es necesario proporcionar la cadena de validación para certificar la autenticidad del archivo. La cadena debe contener 8 caracteres alfanuméricos separados por un guión medio (ej. ABCD-123F)

El manejo y administración de las llaves públicas Entrust y GPG es llevada a cabo por los mismos administradores del sistema, sin embargo este proceso queda fuera del alcance de este proyecto.

3.1.1.2Notificaciones por correo electrónico

Para las notificaciones por correo electrónico (éxito y error) se pueden proporcionar más de una dirección separadas por dos puntos ":", hasta un máximo de 9. La interfaz de administración pregunta el número de direcciones y las ordena de manera que puedan ser codificadas por el sistema. Además, valida que se proporcionen direcciones válidas con el formato cadena@cadena.cadena. El fragmento de código en la figura 3.4 muestra la manera en que se lleva a cabo la validación del contenido de las direcciones de correo:

```
echo -e "***** Successful Notification Email Addresses (Y/N) ?*****"
read success
case $success in
        y|Y) echo "     How many addresses do you want?"
                read number
                i=1
                for (( i = 1; i <= $number; i++ ))
                do
                        echo -e "       Successful Email Address $i"
                        read arr_address[$i]
                        if ! echo ${arr_address[$i]} | egrep '^.+@+.+\.+[a-zA-Z]+$' > /dev/null; then
                                echo -e "Invalid Email Addrees..."
                                ${arr_address[$i]}=" "
                        fi

                done
                #echo "ARREGLO***   ${arr_address[@]} ****"
                #if ! echo ${arr_address[@]} | egrep '^.+@+.+\.+[a-zA-Z]+$' > /dev/null; then
                 #       echo -e "Invalid Email Addrees..."
                #       exit 10
                #fi

                array_separator=:
                array_contents=$( printf "%s$array_separator" "${arr_address[@]}" )
                success_address=`echo $array_contents | sed 's/.$//g'`
```

Figura 3.4 Código para la validación de las cadenas de email

Para hacer esta validación se utiliza el comando Unix egrep para la búsqueda de cadenas y expresiones regulares [10] para verificar que el patrón cadena@cadena.cadena.

En la siguiente sección se muestra el flujo de información que sigue el programa al crear una ruta en el archivo sftpaccess:

3.1.1.3 Armado de las rutas

Para la creación de la ruta se deben de definir el nombre del empleado, nombre del cliente y prefijo del archivo. La figura 3.5 muestra el flujo de ejecución para la creación de una ruta:

```
lirry@debian:~/sftp/secureftp$ ./routes
mar nov 25 13:50:18 CST 2008

*********************************************
         SECFTPACCESS file Utility
         Please choose your option
*********************************************
          a. Create Route
          b. Delete Route
          c. Modify Existing Route
*********************************************
Option:
a

***** Source *****
smith_barney
***** Target *****
morgan_stanley
***** Prefix *****
smith2morgan
***** Passon (Y/N)? *****
y
- Please provide the server IP or server name
transfer.server.com
- Login ID
login_id
- Password
p4ssw0rd
- Directory
/home/login_id
File 'passon.morgan_stanley' is being created...
-------------------------------------
#! /bin/sh
#
#        $1 File name to passon

USERS=/export/ftp/users
morgan_stanley=$USERS/morgan_stanley

echo "Dir: $morgan_stanley"; ls -al $morgan_stanley; echo ""
cd $morgan_stanley || { echo "Can't CD to $morgan_stanley - ABORTING !!!"; exit 255; }

ftp -n -iv transfer.server.com  <<-EOF
user login_id p4ssw0rd
        pwd
        cd /home/login_id
        pwd
        bin
        prompt
        put $1
        bye
EOF
-------------------------------------

Do you want to test passon.morgan_stanley file? (y/n)
```

Figura 3.5 Flujo de ejecución para la creación de una ruta

Como se observa en la figura 3.5, se eligió la opción para crear automáticamente el script de reenvío passon, por los que se proporcionaron la dirección IP del servidor, el nombre de usuario, la contraseña y el directorio destino.

Una vez creado el archivo, el sistema tiene la opción para probar automáticamente la conectividad con el servidor de reenvío utilizando las credenciales proporcionadas en el paso anterior (Véase figura 3.6):

```
Connected to transfer.server.com.
220 debian.localdomain FTP server (Version 6.4/OpenBSD/Linux-ftpd-0.17) ready.
331 Password required for login_id.
230- Linux debian 2.6.26-2-686 #1 SMP Wed Nov 4 20:45:37 UTC 2009 i686
230-
230- The programs included with the Debian GNU/Linux system are free software;
230- the exact distribution terms for each program are described in the
230- individual files in /usr/share/doc/*/copyright.
230-
230- Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
230- permitted by applicable law.
230 User login_id logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
257 "/export/ftp/users/morgan_stanley" is current directory.
250 CWD command successful.
257 "/export/ftp/users/morgan_stanley/files" is current directory.
200 Type set to I.
Interactive mode on.
(local-file) (remote-file)
221 Goodbye.
```

Figura 3.6 Prueba de conectividad

Cuando el proceso de reenvío termina, se deben de proporcionar las opciones de cifrado y las direcciones de correo electrónico para la notificación en caso de error y éxito. En la figura 3.7 se muestra el flujo de ejecución para culminar con la creación de la ruta.

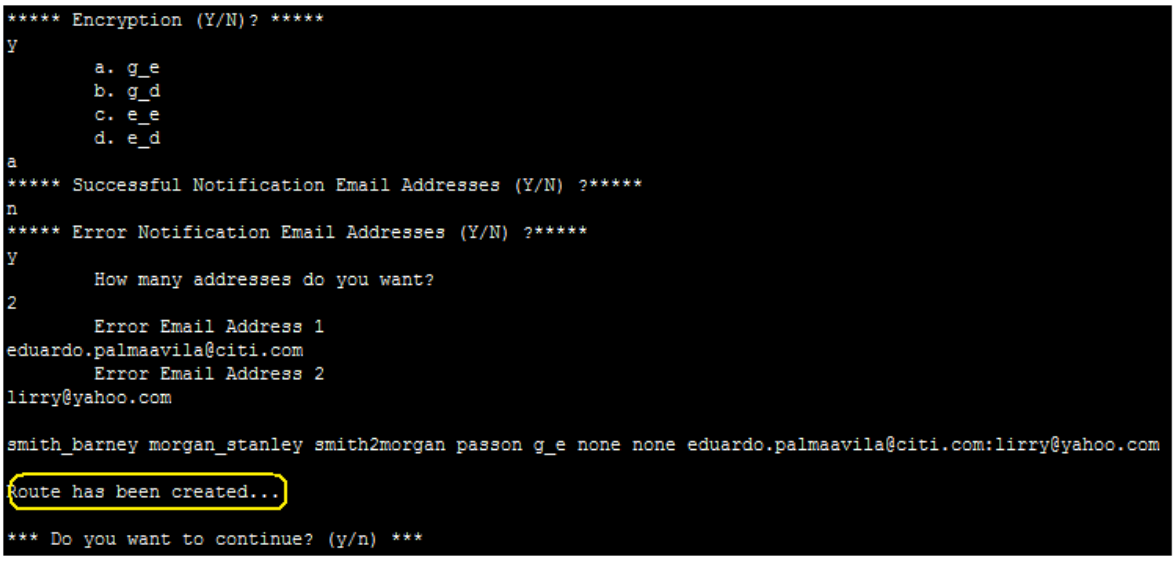


Figura 3.7 Flujo final para la creación de una ruta

Antes de terminar la ejecución se tiene la opción para volver al menú principal e iniciar el proceso.

3.1.2 Borrado de Rutas

Antes de borrar alguna ruta, el sistema crea una copia de seguridad que sirve para restaurar el archivo original en caso de alguna falla o error.

Existen dos opciones para el borrado de rutas: borrado simple y borrado múltiple.

3.1.2.1 Borrado simple

En el borrado se incluye la programación awk [11]. Awk es un lenguaje de programación diseñado para procesar datos basados en texto, ya sean archivos o flujos de datos. En este caso este lenguaje fue útil en la búsqueda ya que permitió identificar y ordenar cada uno de los campos que conforman una ruta.

Para el borrado simple se deben de proporcionar el emisor, receptor y prefijo. Con esta opción solo se borrará la ruta que cumpla con las características especificadas. En caso de que no exista la ruta el programa muestra el mensaje de error mostrado en la siguiente figura 3.8:

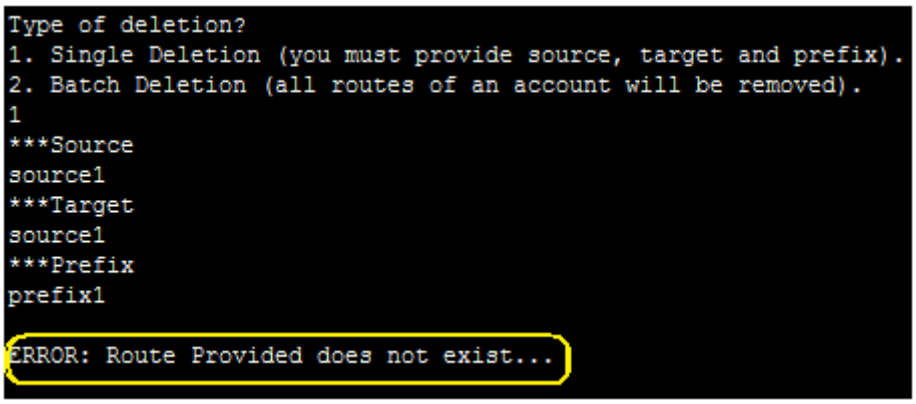


Figura 3.8 Mensaje de error cuando se proporciona una ruta inexistente

Al introducir los datos, el sistema verifica la existencia de los mismos y crea un vector [12] (matriz de 1xn, dónde n es el número de registros). Utilizando awk busca las cadenas proporcionadas los campos específicos $1, $2 y $3. Si las coincidencias son exitosas asigna el valor 1 al elemento del vector. Cuando el programa encuentra un 1 ejecuta las operaciones de borrado. El sistema devuelve las rutas en la pantalla y pide al usuario que confirme el borrado de las mismas. Una vez hecha la confirmación el flujo continúa y termina con un mensaje de confirmación que las rutas se han eliminado de manera exitosa (Véase figura 3.9).

En caso de que no exista alguna coincidencia (vector nulo) el programa devuelve un mensaje de error mostrado en la figura 3.8.

```
## Single Deletion ###
1) echo -e "***Source"
        read source
        echo -e "***Target"
        read target
        echo -e "***Prefix"
        read prefix
        flag=`awk -v source="$source" -v target="$target" -v prefix="$prefix" -v nsource="$nsource" '{
        if (($1 == source) && ($2 == target) && ($3 == prefix))
        {
                print 1
        }
        else print 0
        }' sftpaccess`
        for element in `echo $flag`
        do
                total=`expr $total + $element`
        done
        #echo "total = $total";

        if [ $total = "1" ]
        then
                #********** Bloque Borrar *************
                cat sftpaccess | \
                awk -v source="$source" -v target="$target" -v prefix="$prefix" '
                { if (($1 == source) && ($2 == target) && ($3 == prefix)) $0=""; print}' | awk '/./' > sftpaccess2
                echo " "
                diff sftpaccess sftpaccess2 | grep '<'
                echo " "
                cat sftpaccess2 > sftpaccess
                rm -f sftpaccess2
                echo -e " "
                echo -e "Route has been deleted..."
                echo -e " "
```

Figura 3.9 Código para la validación de la existencia de la ruta

Si el programa encuentra la ruta, la muestra en pantalla junto con la confirmación de que ha sido borrada, como se muestra a continuación (figura 3.10):

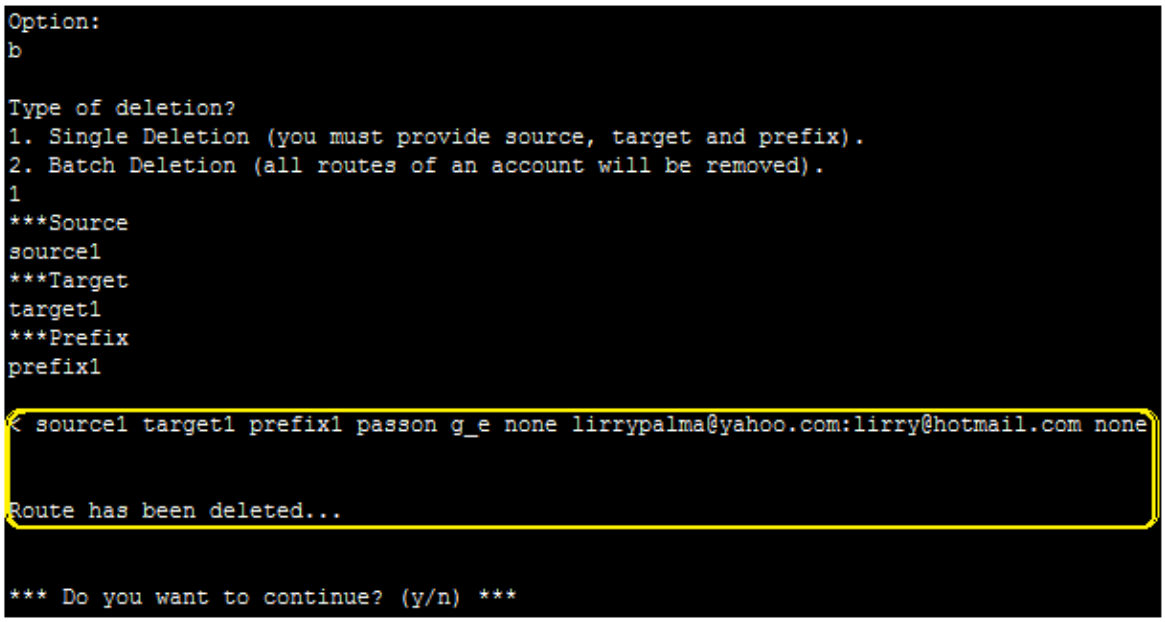


Figura 3.10 Flujo de ejecución cuando la ruta existe

3.1.2.2 Borrado múltiple

Para el caso de borrado múltiple solo es necesario proporcionar una de las cuentas (empleado o cliente) y el sistema borrará de manera automática todas las rutas (internas y/o externas) en las que se encuentre involucrada dicha cuenta. Esta opción es muy útil ya que borra en menos de 3 segundos todas las rutas involucradas, a diferencia de hasta 1 minuto que toma el borrar una sola de manera manual.

En este caso sólo se valida que se cumplan una de las dos condiciones, que el valor proporcionado en la entrada estándar sea igual al campo 1 ($1) ó al campo 2 ($2).

El programa identifica cada renglón en dónde se encuentra ese valor y por diferencia de archivos utilizando el comando del sistema diff crea un nuevo archivo ya sin los renglones que contienen el valor. En la figura 3.11 se muestra el fragmento de código dónde se realizan estas acciones:

```
awk -v account="$account" '
        { if (($1 == account) || ($2 == account)) $0=""; print}' | awk '/./' > sftpaccess2
        diff -d sftpaccess sftpaccess2 > /dev/null
        if [ $? -eq 0 ]; then
                rm -f sftpaccess2
                echo -e " "
                echo -e "Account provided does not have routes..."
                echo -e " "
        else [ $? -eq 1 ];
                count=`cut -d" " -f1,2 sftpaccess | grep $account | wc -l`
                echo -e "You are about to delete $count routes of sftpaccess file, do you want to continue? (yes/no)"
                read confirm
                case $confirm in
                        yes|YES)
                                diff sftpaccess sftpaccess2 | grep '<'
                                cat sftpaccess2 > sftpaccess
                                echo " "

                                echo " "
                                rm -f sftpaccess2
                                echo -e " "
                                echo -e "Routes of account '$account' have been deleted..."
                                echo -e " "
```

Figura 3.11 Código en dónde se realiza el borrado múltiple

Si hay coincidencias busca y cuenta el número de rutas que serán borradas. Pregunta al usuario su confirmación para que el programa internamente realice una sustitución de archivos ya sin las rutas.

Si no encuentra alguna coincidencia manda un mensaje de error y termina la ejecución del programa.

A continuación se muestra el flujo en la operación del borrado múltiple (Véase figura 3.12).

```
Type of deletion?
1. Single Deletion (you must provide source, target and prefix).
2. Batch Deletion (all routes of an account will be removed).
2
*** Secure FTP account?
eduardo
You are about to delete 5 routes of sftpaccess file, do you want to continue? (yes/no)
yes
< eduardo avila tets none e_g none eduardo palma
< eduardo citidoc citi none none none test@cit.com test@citi.com
< eduardo palma avila passon none none none none
< eduardo maximum MAXO none none none none lirry@hotmail.com
< maximum eduardo MAXI passon g_e none lirry@hotmail.com


Routes of account 'eduardo' have been deleted...


*** Do you want to continue? (y/n) ***
```

Figura 3.12 Flujo de ejecución del borrado múltiple

3.1.3 Modificación de Rutas

Para la modificación es necesario proporcionar el emisor, receptor y prefijo que conforman una ruta. Si ninguna ruta con los datos especificados existe, el flujo del programa termina y muestra un mensaje de error.

Para realizar la sustitución de los valores existentes con los nuevos proporcionados a través de la entrada estándar fue necesaria la creación de una función.

Primero realiza la validación de los campos $1, $2 y $3. Si se cumple esta condición asigna el valor de 1 al elemento del vector.

El campo 4 de la función significa el parámetro que será sustituido.

El programa contiene un menú con las posibles opciones. El usuario indica el campo que desea modificar y el nuevo valor que desea asignar. La figura 3.13 muestra el código desarrollado que realiza estas opciones:

```
#************* Modificar ***************
echo -e "       ****************************************"
echo -e "         What parameter do you want to change?"
echo -e "       ****************************************"
echo -e "       1. Source"
echo -e "       2. Target"
echo -e "       3. File Prefix"
echo -e "       4. Passon"
echo -e "       5. Encryption"
echo -e "       6. Entrust validation string"
echo -e "       7. Successful email address"
echo -e "       8. Error email address"
echo -e "       ****************************************"
echo -e "       Option:"
read option
echo -e "   "
case $option in
1) echo -e "    New Source"
        read nsource
        while ( [ "X$nsource" == "X" ] || [ ! -z $(echo $nsource | sed 's/[a-zA-Z0-9]//g') ] || [ ${#nsource} -ne 7 ] ); do
                echo "  [ Warning ] - Not null value, only alphanum, lenght=7... try again!"
                echo ""
                read nsource
        done
        change $source $target $prefix $option $nsource
;;
```

Figura 3.13 Código para la asignación de nuevos valores

La variable *$field* con el valor *new* contienen la nueva información, los cuales son procesados por la función *change()* que realiza la sustitución y almacena la nueva información en un archivo temporal para posteriormente reemplazar el archivo original (Véase figura 3.14):

```
#*** Function change ***#
change ()
{
        awk -v source="$1" -v target="$2" -v prefix="$3" -v field="$4" -v new="$5" '{
        if (($1 == source) && ($2 == target) && ($3 == prefix))
        $'field'=new; print
        }' sftpaccess > sftpaccess_bk
        echo -e " "
        diff sftpaccess sftpaccess_bk | grep '>'
        echo -e " "
        cp sftpaccess_bk sftpaccess
        rm -f sftpaccess_bk

        echo -e "*******************************************"
        echo -e "    [ OK ] - Route has been updated!...      "
        echo -e "*******************************************"
}
```

Figura 3.14 Función principal para la modificación de una ruta

Si los campos $1, $2 y $3 no existen en el registro, el programa asigna el valor de cero al elemento del vector y si se tiene un vector nulo el programa envía el mensaje de error mostrado en la figura 3.15:

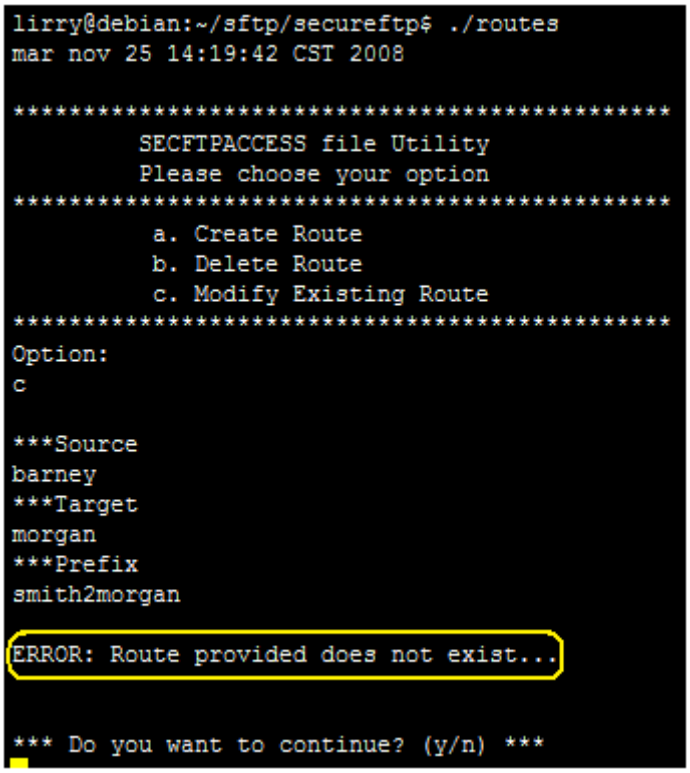


Figura 3.15 Mensaje de error cuando la ruta no existe

En caso de que exista aparece un menú de selección en el que el programa pregunta al usuario cual de los 8 campos desea modificar (emisor, receptor, prefijo, passon, tipo de cifrado, cadena de validación, notificación exitosa o notificación de error.)

Después de elegir la opción deseada el sistema pregunta por el nuevo valor y lo sustituye por el valor anterior, mostrando un mensaje de confirmación. El flujo de ejecución descrito se muestra en la figura 3.16:

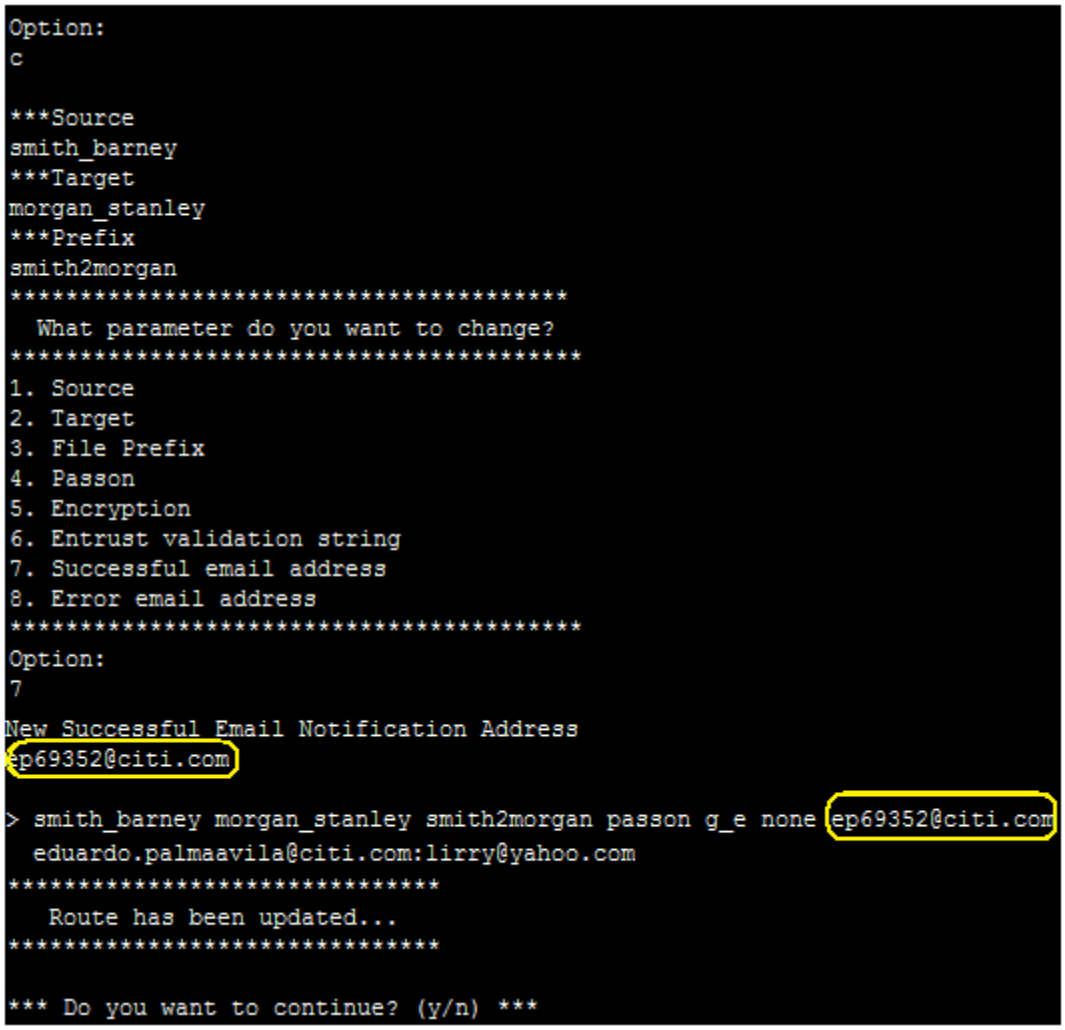


Figura 3.16 Flujo de ejecución en la modificación de una ruta

Capítulo 4

Análisis y Metodología empleados

El método utilizado para el desarrollo de este sistema fue el siguiente:

4.1 Análisis de requisitos

Primeramente se eligieron los requisitos necesarios para este sistema: compiladores propios del sistema operativo, desarrollo de una interfaz amigable, y efectiva para la reducción en la tasa de error y en el tiempo de respuesta, capaz de ser utilizada por cualquier administrador aún sin conocimientos en sistemas UNIX.

Las opciones necesarias son agregar, modificar y eliminar rutas.

4.2 Diseño y Arquitectura

Cualquier sistema operativo UNIX contempla los archivos de una manera muy simple y general dentro de un sistema único. Ve de la misma manera los directorios, archivos ordinarios, los dispositivos tales como impresoras, discos, teclados y terminales de pantalla. En UNIX todo es un archivo.

Este tipo de sistemas ofrece interfaces de programación, a diferencia de otros sistemas operativos propietarios que una visión simplificada del comportamiento de las aplicaciones, usualmente “entrada -> proceso -> salida”. UNIX ofrece una interfaz para la programación de aplicaciones de tal manera que el proceso puede ser modificado por los usuarios para obtener los resultados deseados.

UNIX ofrece herramientas de software ya que desde sus inicios introdujo una nueva idea: que los problemas pueden ser resueltos y las aplicaciones pueden ser creadas al interconectar algunas pocas partes. Estas partes son usualmente componentes que son diseñados para hacer un solo trabajo y hacerlo eficazmente.

Desde la línea de comandos se pueden ejecutar comandos básicos que van desde copiar el contenido de un archivo en otro, buscar si una cadena existe, listar las propiedades de algún archivo, etcétera. Al unir estas características podemos construir sistemas poderosos y robustos que ejecuten lo que el programador necesita.

La principal ventaja es que el sistema SecureFTP se ejecuta sobre un sistema UNIX por lo cual interactúa con archivos de tipo ASCII. Para ello se decidió utilizar todas las herramientas que posee el sistema: un lenguaje de programación capaz de interactuar entre el sistema de archivos y las llamadas al sistema.

El funcionamiento del sistema se realiza al introducir las instrucciones en forma de texto, a fin de que el intérprete de líneas de comandos lo ejecute.

Por las características que proporciona el Shell de UNIX fue posible crear un programa que contiene todas las instrucciones necesarias para posteriormente ser ejecutadas por el intérprete del sistema operativo, y generar una interfaz fácil de utilizar y efectiva para los propósitos descritos anteriormente.

En la figura 4.1 se muestra la estructura básica del sistema antes del desarrollo de la Interfaz de administración:

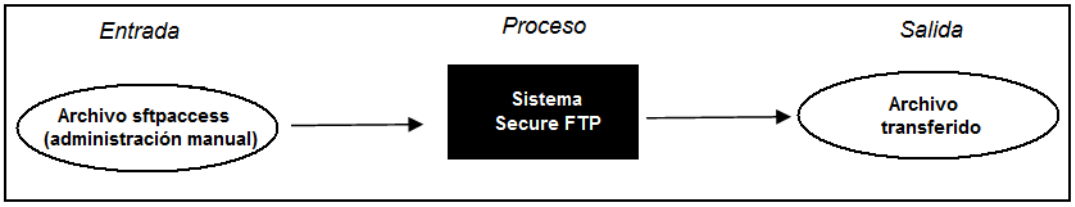


Figura 4.1 Estructura del sistema antes del desarrollo de la interfaz

La entrada es proporcionada por el archivo de configuración sftpaccess, en el cual se definen las rutas y las características de las transferencias. El sistema procesa esa información para llevar a cabo la transferencia de los archivos.

En este caso se utilizaron las herramientas que proporciona el sistema operativo para modificar el proceso del módulo de entrada, es decir, se creó una interfaz de configuración para el archivo secftpaccess (Véase figura 4.2):

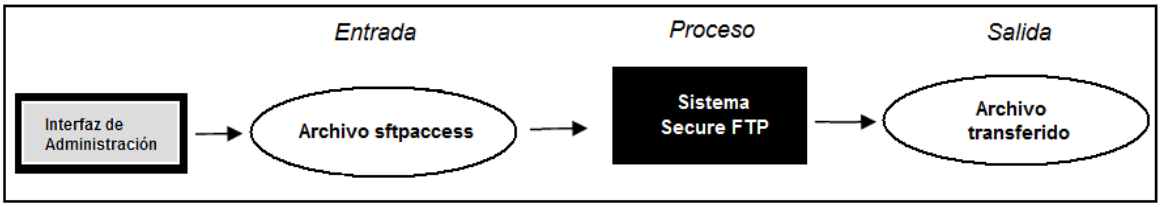


Figura 4.2 Estructura del sistema después del desarrollo de la interfaz

4.3 Programación

De acuerdo a las etapas previas se decidió utilizar programación estructurada en UNIX Shell script con un menú de selección que contiene las tres opciones requeridas: agregar, modificar y borrar rutas.

Dependiendo de la opción elegida el sistema valida la entrada proporcionada por el usuario y si es válida pasa a la siguiente opción.

Se completa la ejecución y como resultado se obtiene la modificación del archivo de configuración de acuerdo a las necesidades requeridas por los clientes. Como se observó anteriormente, se utilizaron diversas funciones del sistema operativo para llevar a cabo las instrucciones especificadas por el usuario.

La programación se realizó en un sistema Linux Debian 1.2, utilizando Bourne Shell. Una ventaja es que este tipo de scripts es portable en todos los sistemas Unix. El programa fue desarrollado en su totalidad en Linux y aunque el sistema SecureFTP corre sobre Unix Sun Solaris es completamente adaptable a esta arquitectura.

La interfaz se desarrolló en 3 módulos y una vez que las pruebas unitarias fueron exitosas, las partes se integraron en un solo sistema.

4.4 Pruebas

Las pruebas se desarrollaron en un sistema Sun Solaris 5.8 UAT (User Acceptance Testing) destinado específicamente para realizar pruebas en los sistemas antes que ser liberados a producción.

Se llevaron a cabo diversas pruebas. A continuación se describe cada una de ellas:

a) Prueba Unitaria

El sistema se dividió en tres módulos: creación, borrado y modificación de las rutas.

La idea fue escribir casos de prueba para cada función de forma que cada caso fuera independiente del resto.

Durante esta fase se detectó un problema ya que el sistema fue desarrollado en Debian utilizando el comando awk. Sin embargo, el sistema operativo dónde está montado SecureFTP es Solaris 5.8 y no reconoce esta instrucción. Para ello fue necesario utilizar gawk, versión GNU de awk [13]. Se modificaron los comandos y parámetros para lograr la portabilidad y funcionalidad del sistema para posteriormente migrarlo a producción.

Esta etapa de pruebas dio a conocer que los sistemas operativos Unix no son compatibles al 100% como se pensaba en un inicio.

Una vez hechas las modificaciones necesarias se llevaron a cabo las pruebas unitarias, las cuales fueron exitosas ya que cada módulo respondió de una manera esperada a las entradas proporcionadas.

b) Prueba de Integración

Una vez concluidas las pruebas unitarias, los módulos se integraron en un solo programa al agregarle el menú principal e integrar los fragmentos de código dentro de instrucciones case.

Se realizó una prueba integral y los módulos respondieron en conjunto de manera adecuada.

Una vez hecho esto se agregó una instrucción para que el programa no termine su ejecución una vez concluida la tarea elegida, sino que pregunte al usuario si desea realizar otra acción antes de finalizar, esto con el objetivo de darle continuidad el flujo del programa.

c) Pruebas de caja blanca

Durante esta etapa se realizó una prueba de cobertura de caminos (véase figura 4.3), es decir, se recorrieron todos los posibles caminos de ejecución, así como la definición y uso de las variables, comprobación de los ciclos utilizados para 0, 1 y n iteraciones. En el programa, n es igual al número de renglones del archivo *sftpaccess*.

Durante esta etapa se detectaron los siguientes errores:

- En el módulo de modificación de ruta no existía la ejecución para insertar los valores definidos de tipo de cifrado, cadena de validación y direcciones de correo, tal y como sucede en el módulo de creación. El objetivo es que durante la modificación se introduzcan el mismo tipo de valores y formato que en la creación. Para solucionarlo se definieron las estructuras similares al módulo de creación de ruta.
- En algunas situaciones, principalmente cuando se elegía un valor incorrecto dentro del menú de opciones, la ejecución del programa terminaba inmediatamente. En lugar de utilizar la instrucción *break* se cambió por *shift* para salir sólo del ciclo interior; ahora el programa pregunta si desea continuar con la ejecución antes de terminar completamente. El objetivo es darle mayor fluidez a la ejecución, aún cuando el usuario elige una opción incorrecta. Esto se muestra en la figura 4.4.

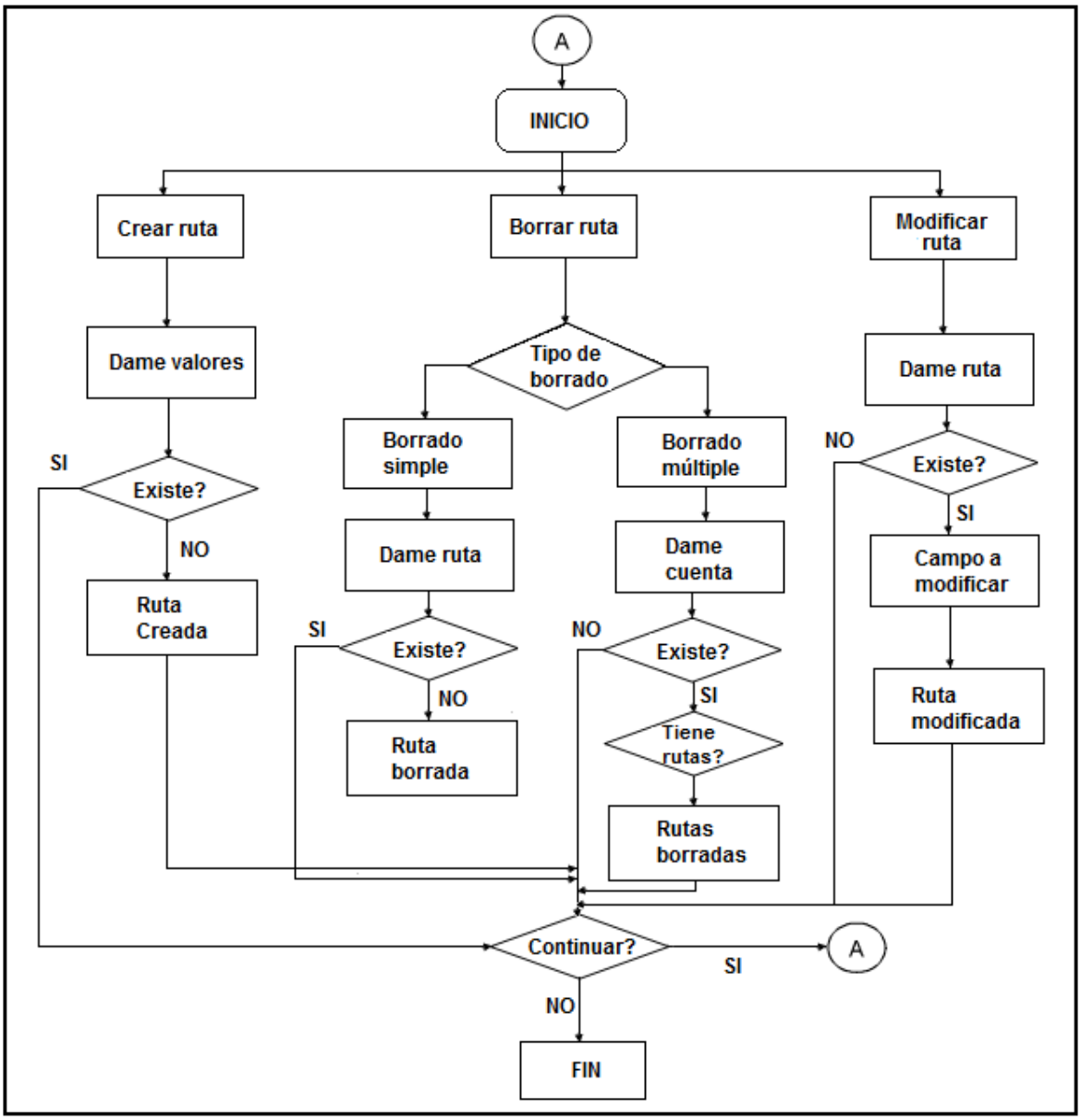


Figura 4.3 Diagrama de flujo

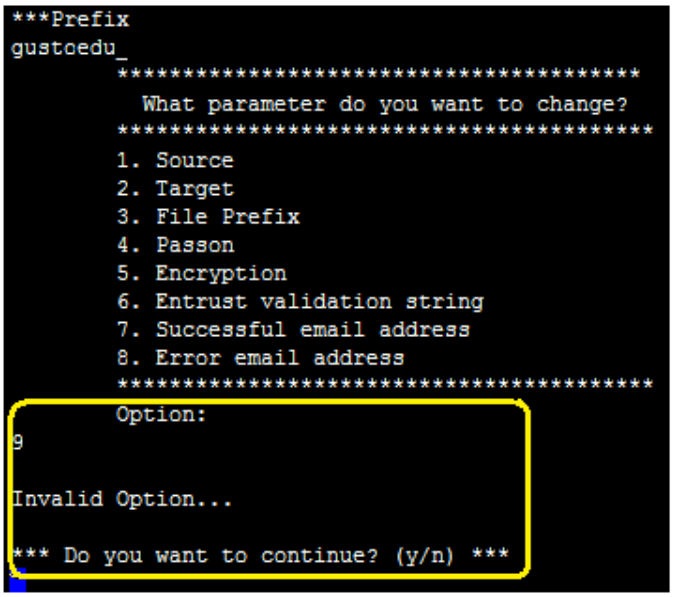


Figura 4.4 Ejecución durante una selección inválida.

d) Pruebas de caja negra

Conociendo la función específica de cada módulo, se proporcionaron distintas entradas de datos y se estudiaron las salidas, lo que trajo como consecuencia los siguientes resultados.

- El sistema aceptaba entradas inválidas como longitud indefinida, uso de caracteres no permitidos, datos nulos. La figura 4.5 muestra la creación de una ruta sin validar los datos de entrada [14].

  Para solucionar esto se insertaron las estructuras para la validación de longitud, uso de caracteres permitidos y validación datos nulos. La figura 4.6 presenta la estructura de código utilizada para realizar estas validaciones y la figura 4.7 la salida al intentar introducir valores incorrectos, ya con esta estructura.

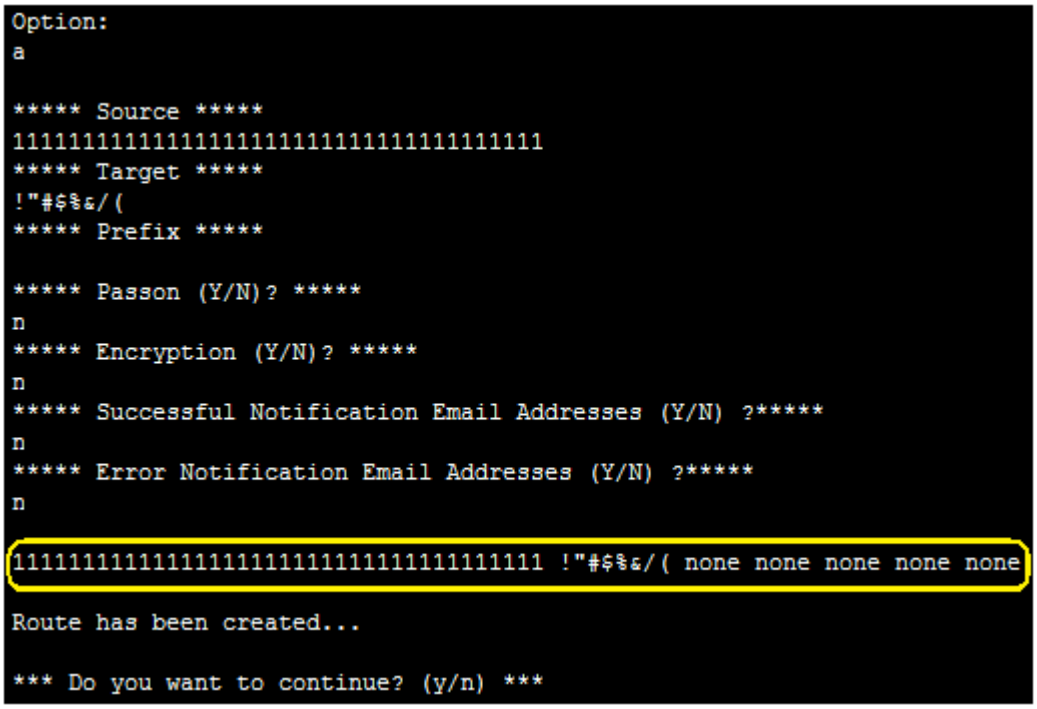


Figura 4.5 Creación de una ruta sin la validación de datos de entrada.

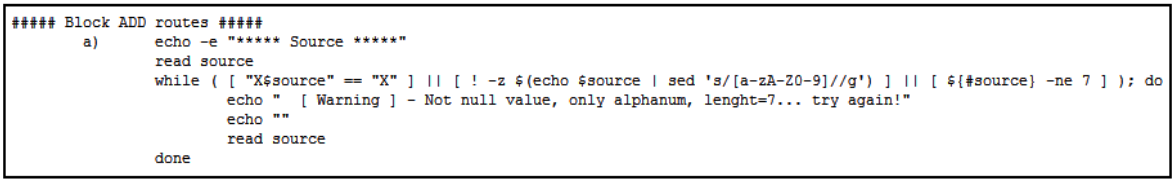


Figura 4.6 Estructura para la validación de datos válidos.

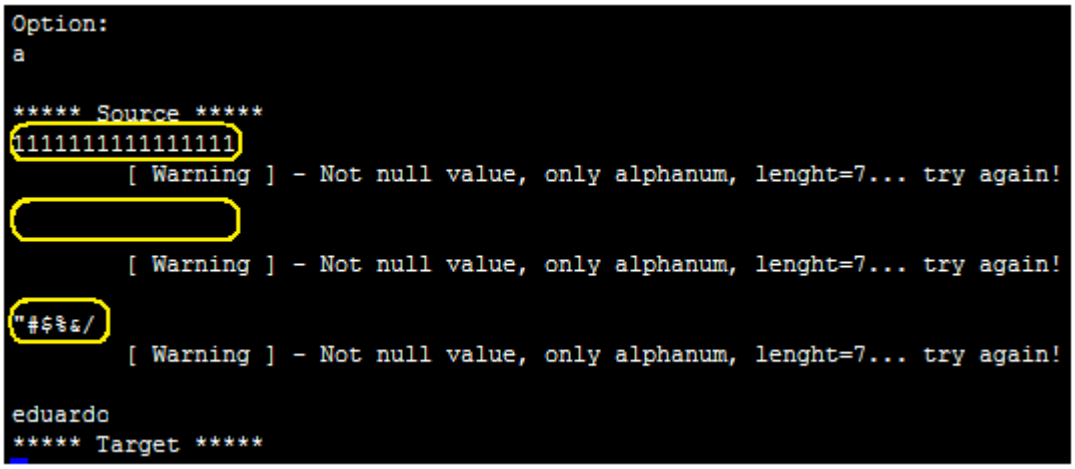


Figura 4.7 Creación de una ruta con la validación de datos de entrada.

- En el caso de la creación de rutas, el sistema no validaba la existencia previa de una ruta por lo que permitía la duplicidad de registros. Se agregó la validación de la existencia de la ruta, si existe el sistema muestra un mensaje de error (Véase figura 4.8)

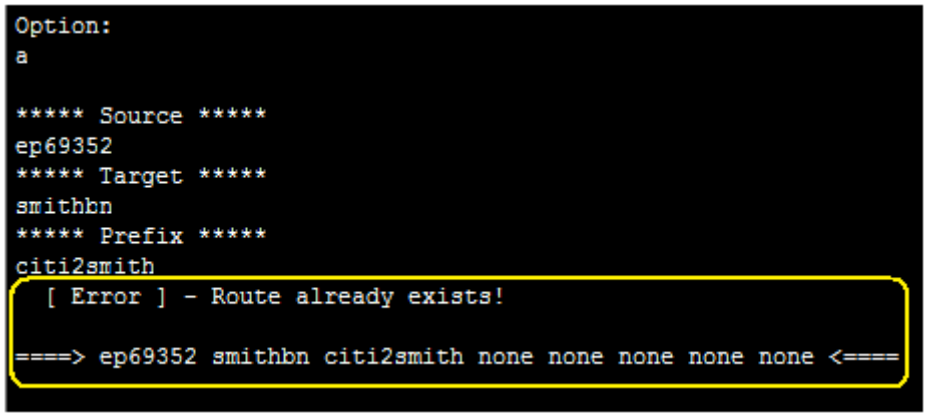


Figura 4.8 Validación de la existencia de una ruta.

A partir de esta fase de pruebas se definieron las características de los datos. (Véase tabla 4.1).

Tabla 4.1 – Diccionario de datos

Dato	Nulo	Tipo	Long min	Long max	Formato
source	NO	alfanum	7	7	
target	NO	alfanum	7	7	
prefix	NO	alfanum + "_" + "-"	4	10	
passon	NO	y|Y | n|N	1	1	
login	NO	alfanum	7	7	
password	NO	alfanum + @#$%^&+=	7	10	
directory	NO	alfanum + "/" + "\"	8	25	
encryption	NO	a | b | c | d	1	1	
entrust	NO	A-Z + 0-9	11	11	CCC-CC-CCCC
success email	NO	[a-zA-Z0-9.-_]+\@+[a-zA-Z0-9.-_]+	7	30	cadena@cadena
error email	NO	[a-zA-Z0-9.-_]+\@+[a-zA-Z0-9.-_]+	7	30	cadena@cadena

- Otro error encontrado fue durante la validación de las direcciones de correo electrónico. El sistema permitía la introducción ilimitada de direcciones. Se agregó la validación para permitir sólo de 1 a 9. En las figuras 4.9 y 4.10 se muestran la estructura utilizada y el mensaje de error al intentar introducir más datos de los permitidos.

```
echo -e "***** Successful Notification Email Addresses (Y/N) ?*****"
read success
case $success in
        y|Y) echo "     How many addresses do you want?"
                read number
                while ( [ "X$number" == "X" ] || [ ! -z $(echo $number | sed 's/^[1-9]$//g') ] || [ ${#number} -ne 1 ] ); do
                        echo "  [ Warning ] - Not null value, only numbers, max 9 addresses... try again!"
                        echo ""
                        read number
                done
```

Figura 4.9 Estructura para la validación del número de direcciones de correo.

```
        Option:
7

    ***** Successful Email Notification Address (Y/N)? *****
y
        New Successful Email Notification Address
                How many addresses do you want?
10
  [ Warning ] - Not null value, only numbers, max 9 addresses... try again!

2
       Successful Email Address 1
eduardo.palmaavila@citi.com
       Successful Email Address 2
admin-sftp@citi.com

> ep69352 smithbn citi2smith none none none eduardo.palmaavila@citi.com:admin-sftp@citi.com none

*******************************************
    [ OK ] - Route has been updated!...
*******************************************
```

Figura 4.10 Mensaje de error en el número de direcciones de correo.

e) Prueba de estrés

Una vez realizadas las pruebas anteriores y haber validado que las funciones del programa hacen lo que tiene qué hacer bajo circunstancias normales, se realizó una prueba de estrés. El objetivo fue determinar el comportamiento de

la interfaz bajo condiciones extraordinarias, principalmente las transacciones simultáneas.

En esta etapa se pidió a todos los administradores, en total 11, su participación simultánea. Todos realizaron pruebas en los distintos módulos y situaciones posibles.

Las pruebas fueron exitosas debido a 2 factores:

1) El archivo no se corrompió, a pesar de todas las instrucciones ejecutadas.
2) Los movimientos realizados fueron validados por los usuarios y todo se ejecutó correctamente.

A pesar de esta situación y teniendo en cuenta los incidentes en los cuáles se corrompió el archivo por uso simultáneo provocando el borrado de cientos de rutas se decidió bloquear los permisos de escritura al archivo *sftpaccess*, permitiendo sólo al programa su modificación, es decir, sólo en el momento en que se tienen listas las entradas se abre el permiso y se cierra inmediatamente una vez insertadas (figura 4.11). Esto se logró utilizando el comando del sistema operativo chattr. Con la opción –i se elimina el bloqueo y se activa con la opción +i [15].

```
esac
echo -e " "
echo $source    $target $prefix $passon $encrypt       $entrust $success_address       $error_address
chattr -i sftpaccess
echo $source    $target $prefix $passon $encrypt       $entrust $success_address       $error_address >> sftpaccess
chattr +i sftpaccess
echo -e " "
echo -e "       [ OK ] Route has been created!..."
```

Figura 4.11 Bloqueo de escritura al archivo sftpaccess.

Ni siquiera el superusuario *root* tiene los privilegios necesarios para modificar el archivo (véase figura 4.12).

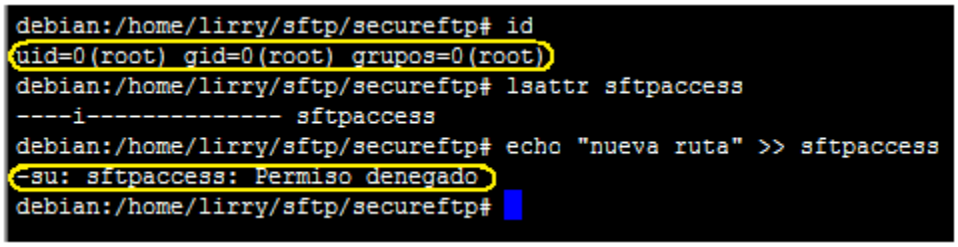


Figura 4.12 Ni el administrador puede editar el archivo sftpaccess.

De esta manera se elimina completamente la posibilidad de que el archivo pueda modificarse simultáneamente (ya sea manualmente o utilizando la interface) y, por consiguiente, que sea corrompido.

4.5 Mantenimiento

El sistema SecureFTP no tiene un tiempo de vida determinado, por lo que la interfaz seguirá utilizándose. Mientras el sistema continúe en producción se seguirá actualizando de acuerdo a las nuevas necesidades y, en su caso, los errores que surjan se corregirán.

Hasta este momento no se han reportado bugs, después de más de 14 meses de operación, y sin embargo se han agregado nuevas funciones a esta herramienta, como la opción de crear y borrar las cuentas de usuario del sistema operativo, realizar respaldos de los archivos cada que se realiza alguna modificación en el sistema; para fines de auditoría, en un archivo se almacena toda la actividad que se lleva a cabo utilizando la interfaz. Estas opciones están fuera del alcance de este reporte, por lo que no se muestran mayores detalles al respecto.

Se plantea la opción de incluir la administración de llaves públicas GPG, es decir, almacenar, firmar, borrar y modificar las llaves de los usuarios.

Como no existe una vigencia del sistema, mi perspectiva hacia el futuro es integrar todas las funciones del sistema SecureFTP para que puedan ser administradas a través de la interfaz, es decir, el manejo de certificados Entrust, llaves GPG, replicación automática de los datos hacia el servidor espejo, etcétera.

Capítulo 5

Participación profesional

El sistema fue desarrollado por iniciativa propia al detectar los problemas de administrar manualmente *SecureFTP*. Utilicé recursos propios y trabajé en fines de semana con el objetivo de no interrumpir mis actividades cotidianas, dedicadas a la administración de la seguridad de los sistemas de Citigroup. Aunque el sistema fue desarrollado en un sistema propio, por cuestiones legales, los derechos del mismo pertenecen a Banamex/Citigroup.

La idea surgió por la falta de participación en la administración del sistema y por un error que se tuvo en el cual se borraron más de 500 rutas ya que dos administradores accedieron y modificaron el archivo al mismo tiempo ocasionando la corrupción de los registros.

La falta de participación se debía a la dificultad que tenían mis compañeros por manejar este sistema. Aprovechando las facilidades que proporcionan el sistema operativo UNIX, mis conocimientos en Seguridad de la Información y programación fue posible desarrollar esta herramienta cumpliendo, además de los objetivos ya descritos, con un aumento en la eficiencia y tiempo de respuesta.

Capítulo 6

Resultados y aportaciones

La interfaz de administración del sistema SecureFTP ha cumplido con los objetivos planteados en un inicio:

- Reducción en la tasa de errores: desde que la interfaz ha sido incluida en el proceso de administración de SecureFTP la tasa de errores ha disminuido de manera considerable. Hasta el momento no se han registrado errores cometidos por los administradores ni problemas generados por la herramienta misma.
- Reducción en los tiempos de respuesta: el tiempo invertido por los administradores ha disminuido extraordinariamente. Una petición que anteriormente tomaba hasta 2 horas, ahora se concluye en cuestión de 3 a 5 minutos. Estos tiempos han incrementado la productividad de los operadores quienes aprovechan el tiempo ahorrado en otras actividades.
- Cumplimiento en el nivel de servicio con los clientes:

Lo anterior se resume en las siguientes figuras:

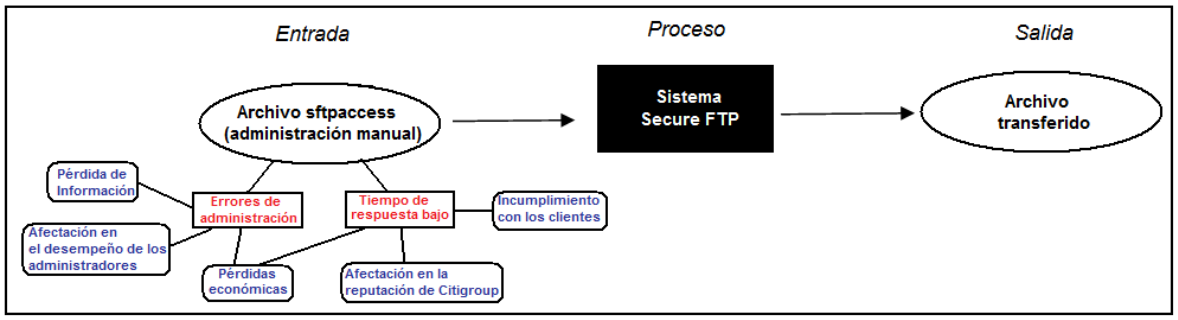


Figura 5.1 Estructura del sistema y problemas antes del desarrollo de la interfaz

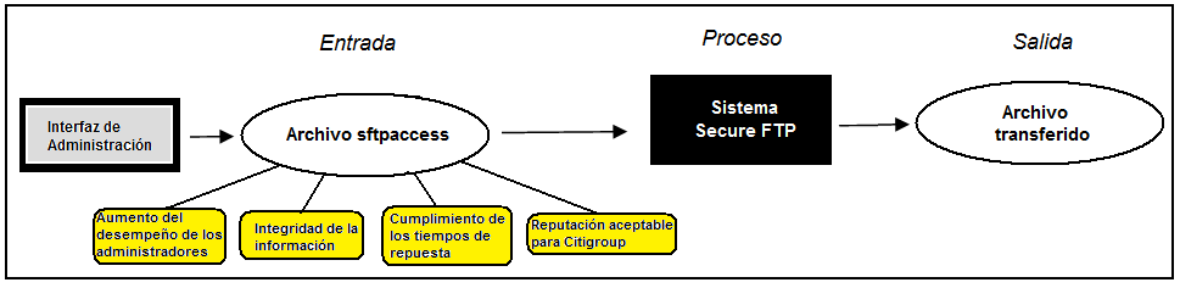


Figura 5.2 Estructura del sistema después del desarrollo de la interfaz

El sistema desarrollado ha sido avalado por la unidad de Infraestructura y Tecnología de Citigroup y se encuentra en funcionamiento desde Diciembre de 2008 siendo utilizado por administradores en México y Hungría.

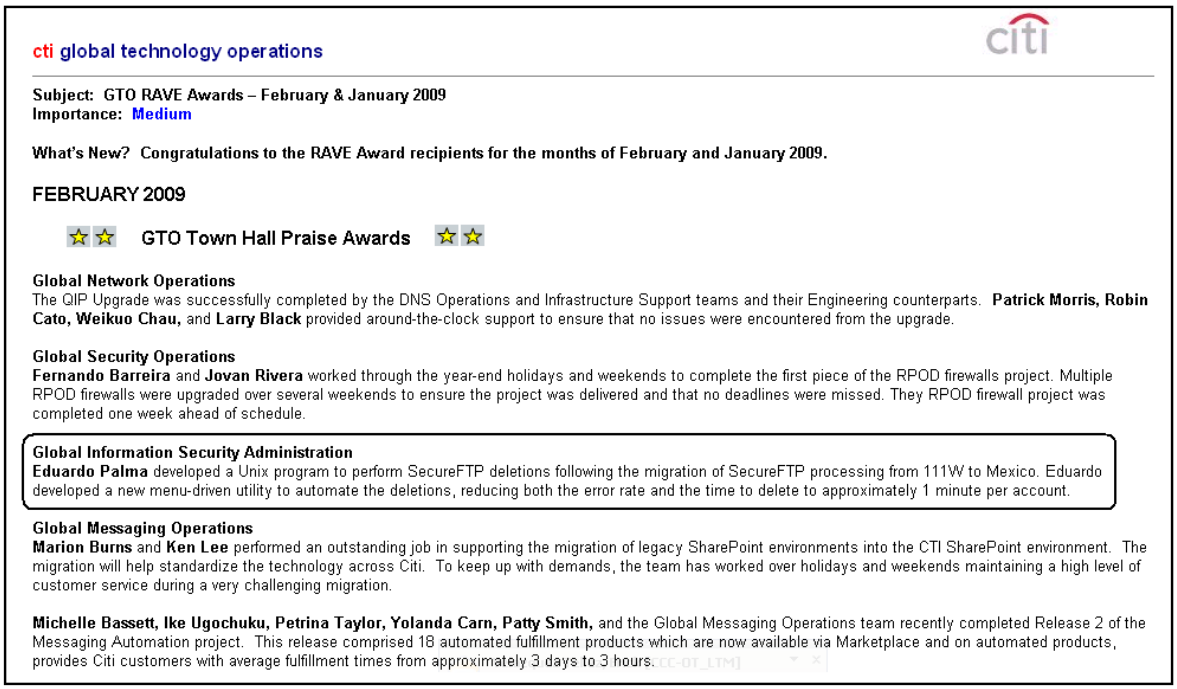
Para la empresa esto significa más que un ahorro de tiempo, recursos y el aseguramiento de los acuerdos de tiempos de respuesta con los clientes, representa la confiabilidad de la información ya que la interfaz asegura, de manera directa o indirecta, la confidencialidad, integridad y disponibilidad de la misma.

Un beneficio adicional y quizá el que mayores satisfacciones personales ha traído fue la inclusión de más administradores al manejo del sistema SecureFTP. Anteriormente sólo una o dos personas realizaban la administración del sistema debido a la complejidad. Mis compañeros se sentían inseguros y preferían evadir estas tareas. Ahora, hasta los menos experimentados usan la herramienta y comentan que es muy sencillo. Sus palabras de agradecimiento han sido la recompensa a todo el esfuerzo y el tiempo invertidos.

En Febrero de 2009, en la reunión bimestral del equipo de Tecnología de Operaciones y Comunicaciones de Citigroup, fui reconocido por esta aportación y recibí el premio RAVE Award que se otorga a empleados destacados por sus logros y aportaciones.

Figura 5.3 CTI Global Technology Operations – Town Hall

Como parte del reconocimiento se envió un mensaje por correo electrónico a los miembros de Tecnología y Operaciones de Citigroup en todo el Mundo dando a conocer a los ganadores del premio RAVE Award del mes de Febrero de 2009.



cti global technology operations

citi

Subject: GTO RAVE Awards – February & January 2009
Importance: Medium

What's New? Congratulations to the RAVE Award recipients for the months of February and January 2009.

FEBRUARY 2009

☆☆ GTO Town Hall Praise Awards ☆☆

Global Network Operations
The QIP Upgrade was successfully completed by the DNS Operations and Infrastructure Support teams and their Engineering counterparts. **Patrick Morris, Robin Cato, Weikuo Chau,** and **Larry Black** provided around-the-clock support to ensure that no issues were encountered from the upgrade.

Global Security Operations
Fernando Barreira and **Jovan Rivera** worked through the year-end holidays and weekends to complete the first piece of the RPOD firewalls project. Multiple RPOD firewalls were upgraded over several weekends to ensure the project was delivered and that no deadlines were missed. They RPOD firewall project was completed one week ahead of schedule.

Global Information Security Administration
Eduardo Palma developed a Unix program to perform SecureFTP deletions following the migration of SecureFTP processing from 111W to Mexico. Eduardo developed a new menu-driven utility to automate the deletions, reducing both the error rate and the time to delete to approximately 1 minute per account.

Global Messaging Operations
Marion Burns and **Ken Lee** performed an outstanding job in supporting the migration of legacy SharePoint environments into the CTI SharePoint environment. The migration will help standardize the technology across Citi. To keep up with demands, the team has worked over holidays and weekends maintaining a high level of customer service during a very challenging migration.

Michelle Bassett, Ike Ugochuku, Petrina Taylor, Yolanda Carn, Patty Smith, and the Global Messaging Operations team recently completed Release 2 of the Messaging Automation project. This release comprised 18 automated fulfillment products which are now available via Marketplace and on automated products, provides Citi customers with average fulfillment times from approximately 3 days to 3 hours.

Figura 5.4 Ganadores del premio RAVE Award – Febrero 2009

Actualmente ya no presto mis servicios a Banamex/Citigroup, sin embargo me siento satisfecho por haber contribuido de esta manera con mi empresa y, sobre todo, con mis compañeros.

Estoy en la mejor disposición para contribuir en la corrección de errores, si fuera el caso, para que la herramienta continúe con las características que le han dado éxito hasta este momento.

Capítulo 7
Conclusiones

Esta interfaz ha cumplido con sus objetivos y ha dado confianza a los operadores del sistema SecureFTP.

Me siento satisfecho de este logro por varios motivos. El diseño, desarrollo e implementación de este sistema me permitió colaborar con mi equipo, mejorar mis habilidades como programador, aprendí muchas cosas del sistema operativo UNIX; comprobé la importancia del diseño. Creo que al no seguirlo y el tratar de programar sin tener algo definido conlleva a errores futuros. Eso me sucedió, por ejemplo, el diccionario de datos debe ser parte del diseño inicial, sin embargo yo supe de su utilidad hasta su última etapa. La fase de pruebas me ayudó a perfeccionar el sistema y hacerlo más amigable y eficiente al acotar las entradas y salidas, facilitando la administración y posibilidad de errores, esto es, desarrollo de código seguro para garantizar la seguridad de la información.

Este aprendizaje me será de mucha utilidad para el desarrollo de futuros proyectos.

Por palabras de mis compañeros, la administración de SecureFTP se ha vuelto sencilla lo que ha permitido que más administradores se hayan involucrado en la administración de este sistema al sentirse más seguros de manipular esta herramienta y, por consiguiente, la productividad y desempeño del grupo se ha incrementado.

En cuestiones de seguridad, la interfaz proporciona manera directa o indirecta Confidencialidad, Integridad y Disponibilidad a la información. Sólo da opción a realizar tareas específicas, aplicando el principio del menor privilegio, permitiendo que los administradores realicen las acciones necesarias y no más.

Mi perspectiva hacia el futuro es la de proponer y desarrollar nuevos sistemas en busca de la eficiencia y, sobre todo, el crecimiento personal y profesional, poniendo siempre en alto el nombre de la Universidad Nacional Autónoma de México.

Glosario de términos

Algoritmo.- es una lista bien definida, ordenada y finita de operaciones que permite hallar la solución a un problema. Dado un estado inicial y una entrada, a través de pasos sucesivos y bien definidos se llega a un estado final, obteniendo una solución. Los algoritmos son objeto de estudio de la algoritmia.

ARPANET.- La red de computadoras ARPANET (Advanced Research Projects Agency Network) fue creada por encargo del Departamento de Defensa de los Estados Unidos como medio de comunicación para los diferentes organismos del país. El primer nodo se creó en la Universidad de California, Los Ángeles y fue la espina dorsal de Internet hasta 1990, tras finalizar la transición al protocolo TCP/IP iniciada en 1983.

Batch job.- (procesamiento por lote): este término se refiere a la ejecución de un programa sin el control o supervisión directa del usuario (que se denomina procesamiento interactivo). Este tipo de programas se caracterizan por que su ejecución no precisa ningún tipo de interacción con el usuario. Generalmente, este tipo de ejecución se utiliza en tareas repetitivas sobre grandes cantidades de información, ya que sería tedioso y propenso a errores realizarlo manualmente. Los programas que se ejecutan por lotes suelen especificar su funcionamiento mediante scripts o guiones (procedimientos) en los que se indica qué se quiere ejecutar y, posiblemente, qué tipo de recursos necesita reservar.

BSD.- (Berkeley Software Distribution) se utiliza para identificar un sistema operativo derivado del sistema Unix nacido a partir de los aportes realizados a ese sistema por la Universidad de California en Berkeley.

Bug.- (error en el software): un bug es el resultado de una falla o deficiencia durante el proceso de creación de programas de computadora (software). Dicha falla puede presentarse en cualquiera de las etapas del ciclo de vida del software aunque las más evidentes se dan en la etapa de desarrollo y programación.

CERT.- (Equipo de Respuesta a Incidentes de Seguridad Informática). Este equipo ayuda a coordinar la comunicación entre expertos durante las emergencias de seguridad y para ayudar a prevenir futuros incidentes. La UNAM cuenta con su propio Equipo de Respuesta a Incidentes de Seguridad Informática. Está localizado en el Departamento de Seguridad en Cómputo (DSC) de la Dirección General de Servicios de Cómputo Académico (DGSCA). El UNAM-CERT se encarga de proveer el servicio de respuesta a incidentes de seguridad en cómputo a sitios que han sido víctimas de algún "ataque", así como de publicar información respecto a vulnerabilidades de

seguridad, alertas de la misma índole y realizar investigaciones de la amplia área del cómputo y así ayudar a mejorar la seguridad de los sitios - http://www.cert.org.mx/index.html

Debian.- es una comunidad conformada por desarrolladores y usuarios, que mantiene un sistema operativo GNU basado en software libre precompilado y empaquetado, en un formato sencillo en múltiples arquitecturas de computadora y en varios núcleos.

Deloitte.- es una de las mayores empresas de servicios profesionales del mundo, por volumen de facturación, y una de las llamadas Cuatro Grandes Auditoras (Big Four auditors), junto con PricewaterhouseCoopers, Ernst & Young, y KPMG.

Diffie-Helman.- algoritmo creado por Whitfield Diffie y Martin Hellman el cual permite el intercambio secreto de llaves entre dos partes que no han tenido contacto previo, utilizando un canal inseguro, y de manera anónima (no autenticada).
Se emplea generalmente como medio para acordar claves simétricas que serán empleadas para el cifrado de una sesión.
Siendo no autenticado, sin embargo provee las bases para varios protocolos autenticados.
Su seguridad radica en la extrema dificultad (conjeturada, no demostrada) de calcular logaritmos discretos en un campo finito.

Directiva Europea de Datos.- es una directiva de la Unión Europea que regula el tratamiento de datos personales dentro de la UE. Es un componente importante de la privacidad y la ley de derechos humanos. Esta directiva fue implementada en 1995 por la Comisión Europea

Entrust.- empresa norteamericana con sede en Dallas, Texas. Esta empresa es especialista en soluciones de seguridad como infraestructura de llave pública (PKI), autenticación por varios factores (*multifactor authentication*), SSL (Secure Socket Layer), detección de fraudes y seguridad en correo electrónico.

Hacker.- Existe una controversia en la definición de hacker. Los hackers de sombrero blanco (White hats) son expertos en seguridad informática que se dedican a asegurar y proteger los sistemas de tecnologías de la información y comunicaciones. Los hackers de sombrero negro (Black hats), también conocidos como crackers, son especialistas que muestran sus habilidades en informática rompiendo sistemas de seguridad de computadoras, colapsando servidores, entrando a zonas restringidas, infectando redes o apoderándose de ellas, entre otras muchas cosas utilizando sus destrezas en métodos hacking. Se utilizará el término hacker para referirse al grupo Black Hat.

HIPAA.- La ley de Seguros Médicos - Health Insurance Portability and Accountability Act (HIPAA) fue promulgada en 1996 por el Congreso de los Estados Unidos. Fue propuesta originalmente por los Senadores Edward Kennedy y Nancy Kassebaum. Esta ley protege la cobertura del seguro médico para los trabajadores y sus familias cuando cambian o pierden su empleo. También regula los estándares nacionales para las transacciones electrónicas relacionadas con los seguros y planes de salud. Además se ocupa de la seguridad y privacidad de los datos de salud. Estos estándares se crearon para mejorar los sistemas de salud al asegurar un intercambio electrónico seguro de datos.

http.- (Hypertext Transfer Protocol o HTTP - protocolo de transferencia de hipertexto) es el protocolo usado en cada transacción de la World Wide Web. HTTP fue desarrollado por el World Wide Web Consortium y la Internet Engineering Task Force, colaboración que culminó en 1999 con la publicación de una serie de RFC, siendo el más importante de ellos el RFC 2616, que especifica la versión 1.1. HTTP define la sintaxis y la semántica que utilizan los elementos de software de la arquitectura web (clientes, servidores, proxies) para comunicarse. Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor.

IETF.- Internet Engineering Task Force (IETF) (Grupo de Trabajo en Ingeniería de Internet) es una organización internacional abierta de normalización, que tiene como objetivos el contribuir a la ingeniería de Internet, actuando en diversas áreas, como transporte, encaminamiento, seguridad. Fue creada en Estados Unidos en 1986. La IETF es mundialmente conocida por ser la entidad que regula las propuestas y los estándares de Internet, conocidos como RFC.

Insider.- El término *insider* se utiliza para definir a las personas o grupo de personas con acceso a información privilegiada. Para fines de este informe, *insider* se refiere a empleados y/o contratistas con acceso a los sistemas y a información confidencial propiedad de las empresas.

Kernel.- (núcleo) es la parte fundamental de un sistema operativo. Es el software responsable de facilitar a los distintos programas el acceso seguro al hardware de la computadora o en una forma básica, es el encargado de gestionar recursos, a través de servicios de llamada al sistema. Como hay muchos programas y el acceso al hardware es limitado, el kernel también se encarga de decidir qué programa podrá hacer uso de un dispositivo de hardware y durante cuánto tiempo, lo que se conoce como multiplexado. Acceder al hardware directamente puede ser realmente complejo, por lo que los núcleos suelen implementar una serie de abstracciones del hardware. Esto permite esconder la complejidad, y proporciona una interfaz limpia y uniforme al hardware subyacente, lo que facilita su uso para el programador.

Mainframe.- Una computadora central o mainframe es una computadora grande, potente y costosa usada principalmente por las compañías para el procesamiento de una gran cantidad de datos; por ejemplo, para el procesamiento de transacciones bancarias. A menudo, las computadoras centrales soportan miles de usuarios de manera simultánea que se conectan mediante terminales falsas. Algunos mainframes pueden ejecutar o alojar a muchos sistemas operativos y por lo tanto, no funcionan como una computadora sola, sino como varias computadoras virtuales.

Malware.- (malicious software): es un software que tiene como objetivo infiltrarse en el sistema y dañar la computadora sin el conocimiento de su dueño, con finalidades muy diversas, ya que en esta categoría encontramos desde un troyano a un spyware.

MD5.- En criptografía, MD5 (Message-Digest algorithm 5) es una función hash muy utilizada en el ámbito criptográfico con 128 bits de valor hash. MD5 ha sido empleado en una gran variedad de aplicaciones de seguridad, y también es comúnmente utilizada para verificar la integridad de los archivos. Sin embargo, ha sido demostrado que MD5 no es resistente a colisiones por lo que no es muy adecuado para aplicaciones como certificados SSL o firmas digitales que confían en esta característica. Un hash MD5 es frecuentemente representado como un número hexadecimal de 32 dígitos.

MIT.- El Instituto de Tecnología de Massachusetts es una de las principales instituciones dedicadas a la docencia y a la investigación en Estados Unidos, especialmente en ciencia, ingeniería y economía. El Instituto está situado en Cambridge, Massachusetts, y cuenta con numerosos premios Nobel entre sus profesores y antiguos alumnos. MIT es considerada como una de las mejores universidades de ciencia e ingeniería del mundo.

Modelo TCP/IP.- son las siglas de Protocolo de Control de Transmisión/Protocolo de Internet (en inglés Transmission Control Protocol/Internet Protocol), un sistema de protocolos que hacen posibles servicios Telnet, FTP, E-mail, y otros entre sistemas que no pertenecen a la misma red.
El Protocolo de Control de Transmisión (TCP) permite a dos computadoras establecer una conexión e intercambiar datos. El TCP garantiza la entrega de datos, es decir, que los datos no se pierdan durante la transmisión y también garantiza que los paquetes sean entregados en el mismo orden en el cual fueron enviados.
El Protocolo de Internet (IP) utiliza direcciones que son series de cuatro números octetos (bytes).

Relaciones simbióticas.- Relaciones simbióticas que son aquellas en las que los organismos viven íntimamente asociados. Se subclasifican en mutualismo (interacciones que benefician a

ambas especies), y comensalismo (interacciones que benefician a un organismo y no afectan al otro).

RFC.- Las Request for Comments (Petición de Comentarios) son una serie de notas sobre Internet que comenzaron a publicarse en 1969. Cada una de ellas individualmente es un documento cuyo contenido es una propuesta oficial para un nuevo protocolo de la red Internet, que se explica con todo detalle para que en caso de ser aceptado pueda ser implementado sin ambigüedades.

RSA.- algoritmo asimétrico cifrador de bloques, que utiliza una llave pública, la cual se distribuye (en forma autenticada preferentemente), y otra privada, la cual es guardada en secreto por su propietario. Fue dado a conocer en 1977 por Ron Rivest, Adi Shamir y Len Adleman en el MIT (Instituto Tecnológico de Massachusetts).

Sarbanes-Oxley.- La Ley Sarbanes Oxley, cuyo título oficial en inglés es Sarbanes-Oxley Act of 2002, Pub. L. No. 107-204, 116 Stat. 745 (30 de julio de 2002), es una ley de Estados Unidos también conocida como el Acta de Reforma de la Contabilidad Pública de Empresas y de Protección al Inversionista. También es llamada SOx, SarbOx o SOA. Esta Ley nace con el fin de monitorear a las empresas que cotizan en bolsa, evitando que las acciones de las mismas sean alteradas de manera dudosa, mientras que su valor es menor. Su finalidad es evitar fraudes y riesgo de bancarrota, protegiendo al inversor.

Simple Mail Transfer Protocol.- (SMTP) Protocolo Simple de Transferencia de Correo, es un protocolo de la capa de aplicación. Es un protocolo de red basado en texto utilizado para el intercambio de mensajes de correo electrónico entre computadoras u otros dispositivos (PDA's, teléfonos móviles, etc.). Está definido en el RFC 2821 y es un estándar oficial de Internet.

SSH.- (Secure Shell) es el nombre de un protocolo y del programa que lo implementa, y sirve para acceder a máquinas remotas a través de una red. Permite manejar por completo la computadora mediante un intérprete de comandos, y también puede redirigir el tráfico de X para poder ejecutar programas gráficos si tenemos un Servidor X (en sistemas Unix y Windows) corriendo. Todo el tráfico de red entre el cliente y el servidor viaja cifrado.

Telnet.- Telnet (TELecommunication NETwork) es el nombre de un protocolo de red (y del programa que implementa el cliente), que sirve para acceder mediante una red a otra máquina, para manejarla remotamente. Para que la conexión funcione, como en todos los servicios de Internet, la máquina a la que se acceda debe tener un programa especial que reciba y gestione las conexiones. El puerto que se utiliza generalmente es el 23.

TLS.- (Seguridad de la Capa de Transporte - TLS) es un protocolo criptográfico que proporcionan comunicaciones seguras por una red, comúnmente Internet. SSL proporciona autenticación y privacidad de la información entre extremos sobre Internet mediante el uso de criptografía.

UNIX Sun Solaris OS.- SunOS es una version del sistema operativo Unix desarrollado por Sun Microsystems para sus estaciones de trabajo y sistemas de cómputo. El nombre SunOS generalmente se refiere a las versiones 1.0 a la 4.1.4 de SunOS. Estas versiones se basaron en BSD Unix, mientras que SunOS a partir de la version 5.0 están basados en Unix System V Release 4, y son vendidas bajo el nombre de Solaris.

Vector.- un vector es un elemento de un espacio vectorial. Se utilizan en sistemas de ecuaciones lineales y tienen representación en el comportamiento algebraico de las funciones.

VPN.- Una red privada virtual o VPN, es una tecnología de red que permite una extensión de la red local sobre una red pública o no controlada, como por ejemplo Internet. Ejemplos comunes son, la posibilidad de conectar dos o más sucursales de una empresa utilizando como vínculo Internet, permitir a los miembros del equipo de soporte técnico la conexión desde su casa al centro de cómputo, o que un usuario pueda acceder a su equipo doméstico desde un sitio remoto, como por ejemplo un hotel. Todo ello utilizando la infraestructura de Internet.

World Wide Web.- es un sistema de documentos de hipertexto y/o hipermedios enlazados y accesibles a través de Internet. Con un navegador web, un usuario visualiza sitios web compuestos de páginas web que pueden contener texto, imágenes, videos u otros contenidos multimedia, y navega a través de ellas usando hiperenlaces. El proyecto fue creado alrededor de 1989 por el inglés Tim Berners-Lee y el belga Robert Cailliau mientras trabajaban en el CERN en Ginebra, Suiza, y publicado en 1992.

Worms.- (gusanos) es un tipo de malware que tiene la propiedad de duplicarse a sí mismo. Los gusanos utilizan las partes automáticas de un sistema operativo que generalmente son invisibles al usuario. A diferencia de un virus, un gusano no precisa alterar los archivos de programas, sino que reside en la memoria y se duplica a sí mismo. Los gusanos casi siempre causan problemas en la red (aunque sea simplemente consumiendo ancho de banda), mientras que los virus siempre infectan o corrompen los archivos de la computadora que atacan.

Referencias

[1] **CISSP Certification All-in-One Exam Guide**, Harris, Shon, McGraw-Hill, 4a. ed., EUA 2007.

[2] **Criptografía**, López Barrientos, Ma. Jaquelina, Universidad Nacional Autónoma de México, Facultad de Ingeniería, México 2009.

[3] http://www.cert.org/archive/pdf/ecrimesummary10.pdf, **Encuesta sobre el incremento en crímenes informáticos**, 19 de Mayo de 2010.

[4] http://www.entrust.com/resources/description.cfm?doc_id=21157, **Protegiendo el activo más importante: la información**, 15 de Mayo de 2010.

[5] http://slacksite.com/other/ftp.html, **Explicación sobre FTP Activo y Pasivo**, 16 de Mayo de 2010.

[6] http://www.entrust.com/file-encryption-software/index.htm, **Cifrado de archivos utilizando Entrust**, 18 de Mayo de 2010.

[7] http://www.gnupg.org/, **GPG (Gnu Privacy Guard)**, 19 de Mayo de 2010.

[8] http://irtfweb.ifa.hawaii.edu/~lockhart/gpg/gpg-cs.html, **Cómo utilizar GPG**, 17 de Mayo de 2010.

[9] **Advanced Programming in the UNIX Environment**, Steven, Richard, Addison-Wesley Professional Computing Series, 2a ed., EUA 2005.

[10] http://www.unix-manuals.com/refs/regex/regex.htm, **Expresiones regulares en UNIX**, 25 de Octubre de 2008.

[11] **Sed & Awk**, Dougherty, Dale, O'Reilly Media, 2a ed., EUA 1997

[12] **Apuntes de Álgebra Lineal**, Solar González, Eduardo, Noriega Limusa, 3a. ed., México, 1999.

[13] http://www.gnu.org/software/gawk/manual/gawk.html, **Guía de usuario de GNU Awk**, 25 de Octubre de 2008.

[14] **Writing Secure Code**, Howard, Michael y LeBlanc, David, Microsoft Press, 2a. ed., EUA 2003

[15] **Introducción a la Seguridad en UNIX**, Alavez, Sergio, Plan de Becarios de Seguridad en Cómputo, Dirección General de Servicios de Cómputo Académico, México, 2005.