ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN
UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Proyecto Fin de Carrera

Implantación de seguridad en entornos Web.

AUTOR: Juan Manuel Saura Martín.
DIRECTOR: Pedro Sánchez Palma.
Julio / 2006

Autor	Juan Manuel Saura Martín
E-mail del Autor	juanmanuelsaura@hotmail.com
Director(es)	Pedro Sánchez Palma
E-mail del Director	pedro.sanchez@upct.es
Título del PFC	Implantación de seguridad en entornos Web.

Resúmen

En el desarrollo Web existen numerosos problemas debido a la seguridad. Ante esta situación, al diseñar páginas Web debemos tenerla en cuenta, tanto software como hardware, para ello es necesario instalar servidores seguros, o con módulos de seguridad donde podamos modificar su configuración para dotarlos de la misma.

En este proyecto se ha hecho un estudio sobre la seguridad Web y hemos estudiado el caso práctico del servidor Apache con lenguaje de programación php y base de datos MySQL. Para tal estudio hemos utilizado protocolos de seguridad como SSL y md5.

Se ha hecho un estudio sobre autentificación de usuarios, comprobando las diferentes maneras que hay de autentificarse.

Además en este proyecto hemos estudiado los certificados digitales, tanto desde la creación de una autoridad que certifique nuestros certificados, hasta la creación de nuestro propio certificado Web.

Titulación	Ingeniero Técnico de Telecomunicación especialidad Telemática
Departamento	Departamento De Tecnologías de la información y Las Comunicaciones (TIC).
Fecha de Presentación	Julio - 2006

Me gustaría agradecer gentilmente a todas las personas que contribuyeran con recomendaciones, ayuda y aliento mientras realizaba este proyecto.

Deseo expresar al profesor Pedro Sánchez Palma por su dirección y asesoramiento, gracias a los cuales ha sido posible la realización de la investigación recogida en este proyecto. Asimismo quiero agradecer su apoyo y la estrecha colaboración que hemos mantenido durante este tiempo.

Primero quisiera agradecer a mi familia y mis Abuelos, en especial a mis padres y mi hermana. Sin su apoyo hubiera sido imposible terminar este proyecto.

Dar las gracias a esa chica que tanto me ha apoyado desde el primer momento y que siempre ha estado en esos momentos difíciles que a veces se presentan, muchas gracias Alicia.

También quisiera agradecer a todos mis colegas y amigos que me ayudaron en la elaboración del mismo.

Por último agradecer a todas aquellas personas que directa o indirectamente me han ayudado o me han apoyado en este proyecto.

A todas estas personas ¡Un millón de gracias!

ÍNDICE DE CONTENIDOS

1 LA SEGURIDAD EN SISTEMAS SOFTWARE 1

1.1 Introducción 1

1.2 La seguridad en sistemas software 1

1.3 Principios básicos 4

1.3.1 Tipos de atacantes más habituales: 4

1.4 La seguridad en entornos Web 7

2 POLÍTICAS GENERALES DE SEGURIDAD 8

2.1 Introducción 8

2.2 ¿Qué es una política de seguridad? 9

2.3 Aspectos físicos de la política de seguridad. 13

2.4 Aspectos lógicos de la política de seguridad. 15

2.5 Aspectos humanos y organizativos de la política de seguridad. 16

2.6 Aspectos legales de la política de seguridad. 18

3 MÉTODOS DE ATAQUE Y SOLUCIONES 22

3.1 Introducción 22

3.2 Elementos de la seguridad 22

3.3 Amenazas 22

3.3.1 Tipos de amenazas. 22

3.3.2 Origen de las amenazas. 23

3.3.2.1 Personas 23

3.3.2.2 Amenazas lógicas 25

3.4 Mecanismos 29

3.5 Taxonomía de los tipos de ataques. 30

3.5.1 Ataques remotos. 31

3.5.1.1 Escaneo de puertos. 31

3.5.1.2 Spoofing. 34

3.5.1.3 Negaciones de servicio. 35

3.5.1.4 Interceptación 39

3.5.2 Ataques orientados a la obtención de información sobre el objetivo. 40

3.5.3 Ataques vía Web. 41

3.5.4 Ataques a los sistemas. 43

3.5.4.1 Troyanos. 43

3.5.4.2 Hackers. 44

3.5.4.3 Virus. 44

3.5.4.4 Hoax. 44

3.5.4.5 Spam. 45

3.5.4.6 Dialers. 45

3.6 Sistemas de detección de intrusos. 45

3.6.1 Clasificación de los IDS. 48

3.6.1.1 Clasificación por situación. 48

3.6.1.2 Clasificación según los modelos de detecciones. 49

3.6.1.3 Clasificación según su naturaleza. 50

3.6.2 Topologías de los IDS. 50

3.6.3 Arquitecturas basadas en IDS. 52

3.6.3.1 Arquitecturas basadas en agentes autónomos. 52

3.6.3.2 Arquitectura de exploración de datos en tiempo real. 54

3.6.4 Características deseables de un IDS. 55
3.6.5 Metodología para la detección de intrusos y para la selección e implantación de sistemas IDS. 57
3.6.6 Detección de "problemas". 60
3.6.7 Detección de usos indebidos. 62

4 AUTENTIFICACIÓN Y CERTIFICADOS DIGITALES 65

4.1 Introducción 65

4.2 Autentificación. 65

4.3 Autentificación HTTP basada en php. 67
4.3.1 Autentificación HTTP Basic. 68
4.3.1.1 Configuración de httpd.conf 68
4.3.1.2 Creación de un usuario. 69
4.3.1.3 Creación de un grupo. 69
4.3.2 Autentificación HTTP Digest. 70
4.3.2.1 Configuración de httpd.conf 70
4.3.2.2 Creación de un usuario. 70

4.4 Firma digital. 71
4.4.1 Funcionamiento de la firma digital. 72
4.4.2 Autoridades de certificación. 72

4.5 Certificados digitales. 74

5 INTRODUCCIÓN A LOS SERVIDORES Web: EL CASO APACHE. 77

5.1 Servidores Web. 77

5.2 Servidor Web: caso Apache. 79
5.2.1 Comprobación del entorno. 80
5.2.2 Instalación de apache en Linux. 81
5.2.3 Configurar apache para seguridad 83
5.2.3.1 El protocolo SSL. 84
5.2.3.2 Creación de nuestra propia CA. 85
5.2.3.3 Creación de nuestro CSR. 86
5.2.3.4 Creación de un CRT. 87
5.2.3.5 Configuración apache. 87
5.2.3.6 Limpieza de archivos. 88

6 LENGUAJE DE PROGRAMACIÓN Web Y BASES DE DATOS. 89

6.1 Lenguaje de programación Web php 89
6.1.1 Instalación de php. 91

6.2 Sesiones vs Cookies 92
6.2.1 Sesiones 92
6.2.1.1 Funcionamiento sesiones 92
6.2.2 Cookies 93
6.2.3 Sesiones vs Cookies 93

6.3 Bases de datos 93
6.3.1 Instalación de MySQL. 94
6.3.2 phpMyAdmin 95
6.3.2.1 Instalación de phpMyAdmin 95

7 APLICACIÓN DE SEGURIDAD: CASO PRÁCTICO 99

7.1 Introducción. 99

7.2 Funcionamiento de la página Web. 99

8 CONCLUSIONES Y LÍNEAS FUTURAS DE TRABAJO 105

Apéndice A 107

Apéndice B _____ *108*
Apéndice C _____ *111*
Apéndice D _____ *115*
Apéndice E _____ *118*
Apéndice F _____ *123*
Apéndice G _____ *128*
Apéndice H _____ *131*
Apéndice I _____ *135*
Apéndice J _____ *137*
Apéndice K _____ *142*
Apéndice L _____ *147*
Apéndice M _____ *149*
BIBLIOGRAFÍA _____ *151*

ÍNDICE DE FIGURAS

Figura 1: *La seguridad como un proceso y la política como su guía.* *11*
Figura 2: *Fase de puesta en marcha del proceso de seguridad.* *11*
Figura 3: *Fase de monitorización del proceso de seguridad.* *12*
Figura 4: *Fase de anàlisis de vulnerabilidades.* *12*
Figura 5: *Consolidación y continuación del proceso de seguridad.* *13*
Figura 6: *Flujo normal de información entre emisor y receptor y posibles amenazas: (a) interrupción, (b) interceptación, (c) modificación y (d) fabricación.* *23*
Figura 7: *Origen de los ataques.* *30*
Figura 8: *Conexiones del PC.* *44*
Figura 9: *Red con IDS simple.* *51*
Figura 10: *Red completa con IDS.* *51*
Figura 11: *Arquitectura de un sistema autónomo.* *53*
Figura 12: *Arquitectura de un IDS basado en tiempo real.* *55*
Figura 13: *Pasos de un intruso.* *57*
Figura 14: *Autentificación y no repudio mediante clave privada* *66*
Figura 15: *Autentificación Basic.* *69*
Figura 16: *Fàbrica nacional de moneda y timbre.* *73*
Figura 17: *Certificados digitales.* *76*
Figura 18: *Mensaje del funcionamiento de apache.* *83*
Figura 19: *Estructura del directorio demoCA.* *86*
Figura 21: *Pàgina principal de PHP.* *92*
Figura 22: *página principal de phpMyAdmin.* *96*
Figura 23: *Bases de datos.* *97*
Figura 24: *Tablas de la base de datos.* *97*
Figura 25: *Registros de la base de datos usuarios.* *98*
Imagen 26: *Arranque del servidor Apache con soporte SSL.* *99*
Figura 27: *Arranque de la base de datos MySQL.* *100*
Figura 28: *Página inicial* *100*
Figura 29: *Advertencia de seguridad SSL* *101*
Figura 30: *Certificado SSL.* *101*
Figura 31: *Sección: Iniciar Sesión.* *102*
Figura 32: *Usuario registrado.* *102*
Figura 33: *Usuario sin privilegios.* *103*
Figura 34: *Usuario con privilegios* *103*
Figura 35: *Carpeta del servidor apache autentificada.* *104*
Figura 36: *Autentificación requerida.* *104*

ÍNDICE DE TABLAS

Tabla 1: *Comparativa Seguridad..... 79*
Tabla 2: *Paquetes del servidor seguro..... 84*

LA SEGURIDAD EN SISTEMAS SOFTWARE

1.1 Introducción

El objetivo de este proyecto fin de carrera es realizar un estudio exhaustivo de la implicación de desarrollar sistemas Web seguros. En particular se tendrá en cuenta los requisitos necesarios para adoptar un esquema de seguridad específico y las consideraciones pertinentes para el caso del lenguaje de programación PHP y un servidor de páginas Web como Apache.

Para ello se comenzará con una base teórica y se seguirá con un desarrollo práctico de los conocimientos adquiridos a lo largo del proyecto.

Los objetivos más relevantes se pueden sintetizar en los siguientes:

- Realizar un estudio del alcance de la seguridad en sistemas software, estudiando la taxonomía de los ataques, estudiando sistemas de detección de intrusos (IDS)…
- Implantación de esquemas de seguridad
- Implicación de la seguridad en el caso de aplicaciones Web. Estudio de autentificación de usuarios mediante formularios, sesiones, criptografía, firmas digitales…
- Estudiar las facilidades que ofrecen entornos de producción de software ligados al Web como son el lenguaje de programación PHP y el servidor de páginas Web Apache. Para ello se instalarán módulos de seguridad al servidor Apache.
- Adoptar algunos esquemas de seguridad específicos para las plataformas mencionadas y recoger de manera esquemática los patrones adoptados.
- Realización de una demostración donde se explicará como instalar Apache y PHP para seguridad, con los módulos mencionados para adoptar la misma. Se creará una base de datos para guardar los password de los usuarios, se llevará a la práctica la creación y firma de certificados digitales bajo el protocolo de seguridad SSL

1.2 La seguridad en sistemas software

Nuestro mundo está repleto de sistemas, compuestos por máquinas y por programas. Casi cada objeto que nos rodea es un sistema, compuesto por una o varias máquinas, controladas por una o varias piezas de componentes que se denominan software.

Muchos de estos sistemas forman parte de redes, privadas o de empresa, públicas, grandes o pequeñas, interconectadas unas con otras y comunicándose entre sí mediante otro gran sistema hardware gestionado por un conjunto de aplicaciones con distintos objetivos, a los que se denominan protocolos.

La red Internet es, con una muy alta probabilidad, el sistema más complejo que se haya desarrollado. Esta compuesto por millones de ordenadores interconectados mediante una red física de muy elevada complejidad. Además cada ordenador contiene gran cantidad de programas que interactúan entre sí o con otros ordenadores de la red.

Podemos decir que los sistemas son complejos y capaces de interactuar, además muchos de ellos hacen cosas no pensadas ni diseñadas (bugs) por sus creadores y usuarios,

estas propiedades hacen que sea difícil conseguir que un sistema, como la red de una Universidad, o de una empresa o la red Internet, sea seguro.

Hasta finales de 1988 muy poca gente tomaba en serio el tema de la seguridad. Mientras que por una parte Internet iba creciendo exponencialmente, por otra el auge de la informática de consumo iba produciendo un aumento espectacular en el número de piratas informáticos.

Sin embargo, el 22 de noviembre de 1988 Robert T. Morris [1] protagonizo el primer gran incidente de la seguridad informática: uno de sus programas se convirtió en el famoso *worm* o gusano de Internet. Miles de computadoras conectados a la red se vieron inutilizados durante días, y las perdidas se estiman en millones de dólares. Desde ese momento el tema de la seguridad en sistemas operativos, redes, seguridad física y lógica ha sido un factor a tener muy en cuenta por cualquier organización.

Uno de los temas que es inevitable comentar en la actualidad al hablar de Internet y de la posibilidad de realizar negocios en la red, es la seguridad. Cuando entra en Internet, está entrando en una gran sociedad en línea. Y, como en toda sociedad humana, no todo es agradable. En el mundo en línea existen amenazas a la seguridad, igual que en el resto del mundo. Si aprende a protegerse (a proteger su equipo y sus datos), se asegurará que el tiempo que pase en línea será productivo y divertido. Las computadoras e Internet son ahora una parte familiar de nuestras vidas. Quizá no las veamos a menudo, pero ahí están, involucradas de alguna manera en la mayoría de nuestras actividades diarias, en los negocios de cualquier empresa, en las instituciones educativas, en las diferentes áreas de gobierno, sin el apoyo de éstas herramientas ninguna de ellas sería capaz de manejar la impresionante cantidad de información que parece caracterizar a nuestra sociedad. Pero también existe una problemática en ellas, la seguridad. Cada vez más personas necesitaran conocer el manejo de las computadoras así como las protecciones que día a día se van ofreciendo para garantizarnos la seguridad en el manejo de la información.

Podemos entender como seguridad una característica de cualquier sistema que nos indica que ese sistema esta libre de todo peligro, daño o riesgo, y que es, en cierta manera, infalible. Como esta característica, particularizando para el caso de Web, sistemas operativos o redes de computadores, es sencillamente imposible, se suaviza la definición de seguridad y hablaremos de **fiabilidad** [2] (probabilidad de que un sistema se comporte tal y como se espera de él).

La popularización de Internet ha conseguido que la mayoría de los hogares cuenten con ordenadores conectados a la red. En un principio, navegar por Internet era demasiado caro como para hacerlo por mero divertimento, pero la amplia oferta de conexiones de banda ancha y de tarifas planas ha conseguido que en los hogares donde existe un ordenador, casi todos los miembros de la familia tengan su propia cuenta de correo electrónico o busquen información en la red de forma sistemática. Por supuesto, esto incluye a los niños y adolescentes, quienes han encontrado en Internet un entretenimiento de similar magnitud al que puede tener la televisión.

Entre las principales razones de la popularización y el éxito de Internet está el hecho de ser una red abierta. Como el protocolo utilizado por los ordenadores que se conectan a Internet, TCP-IP, es gratuito, cualquier red y cualquier ordenador pueden conectarse sin más costes que los de la conexión. No hay ningún propietario de Internet, no hay ninguna autoridad central que pueda imponer un precio o unas condiciones diferentes de las estrictamente técnicas.

Hay cientos de millones de usuarios en Internet. El cálculo estadístico de cuántos usuarios tienen acceso a Internet ha perdido ya sentido. Hay clubes, cafés-Internet y locutorios públicos gestionados por instituciones privadas o públicas en ciudades de todo el mundo, incluyendo los países menos desarrollados, por lo que son miles de millones los usuarios que pueden en cualquier momento, por un coste inferior a un euro, conectarse a Internet durante un rato. Esta extraordinaria facilidad de acceso y popularidad es el principal atractivo desde el punto de vista comercial pero también es la causa de que Internet esté abierto a todo tipo de indeseables.

Internet se ha convertido en una parte esencial del trabajo, con el tiempo, el contenido Web se ha vuelto más interesante, al evolucionar desde folletos en línea estáticos con texto y fotografías sencillos hasta archivos multimedia dinámicos e interactivos con gráficos animados. A medida que Internet y sus contenidos son más sofisticados y accesibles, también son más variadas y frecuentes las amenazas a la seguridad de la red, a la productividad de los empleados, a la responsabilidad corporativa y al consumo del ancho de banda.

En realidad, cualquier calle comercial de cualquier ciudad del mundo es también accesible a los malhechores. Cualquier transacción económica realizada por medios tradicionales es susceptible de ser aprovechada por ladrones. Las comunicaciones comerciales realizadas por medios tradicionales, cartas o teléfono, son mucho más fáciles de interceptar que las comunicaciones a través de Internet. Realizar actividades delictivas a través de Internet requiere unos conocimientos técnicos sofisticados que no están al alcance de cualquiera.

Por otra parte, las posibilidades de protección de las comunicaciones electrónicas son mucho mayores que las que permiten los medios tradicionales. Hay programas de ordenador gratuitos y muy fáciles de usar que permiten a cualquier usuario la encriptación de sus mensajes de forma que queda plenamente garantizado que sólo el destinatario podrá entenderlos. Los certificados y firmas electrónicas garantizan la identidad de los sujetos con mucha mayor garantía que cualquier fedatario tradicional. Los sistemas de almacenamiento de datos y su protección frente a accidentes fortuitos o ataques intencionados son más fáciles, baratos y seguros que las cajas fuertes o cámaras de seguridad.

Lo que ocurre es que no hay una “cultura” de la seguridad en Internet. La sociedad en que vivimos nos ha enseñado desde que éramos niños unas reglas básicas de protección de nuestras propiedades. El gesto de cerrar la puerta de casa, los límites que nos imponemos a la cantidad de efectivo que llevamos en el bolsillo, la forma en que reaccionamos cuando nos aborda un extraño por la calle, son comportamientos que hemos aprendido a lo largo de nuestra vida. En cambio nuestra experiencia con Internet es muy breve y ni nuestros padres ni nuestros profesores nos dijeron nunca cómo debíamos comportarnos en el ciberespacio.

La protección legal del comercio electrónico ha requerido también la elaboración de nuevas normas. La protección frente a la publicidad indeseada cuyo coste de transmisión recae sobre el consumidor requiere ahora un tratamiento diferente que cuando el coste recaía exclusivamente sobre el anunciante. El reconocimiento jurídico de las firmas electrónicas y del arbitraje electrónico en los países de la Unión Europea ha establecido un marco legal que garantiza la calidad de los certificados y agiliza los trámites judiciales. Los gobiernos de todo el mundo están interesados en promover el desarrollo del comercio electrónico por lo que están impulsando reformas legales y fiscales que permiten y agilicen las transacciones a través de Internet.

La seguridad en Internet y las leyes que la protegen, están basadas principalmente en los sistemas de encriptación. Esos sistemas son los que permiten que las informaciones que circulan por Internet sean indescifrables, ininteligibles, para cualquier persona que no sea aquella a la que va destinada.

En general podemos hablar de dos grandes divisiones en la seguridad:

- **seguridad física:** se refiere a la protección del Hardware y de los soportes de datos, así como a la de los edificios e instalaciones que los albergan. Contempla las situaciones de incendios, sabotajes, robos, catástrofes naturales, etc.
- **seguridad lógica:** se refiere a la seguridad de uso del software, a la protección de los datos, procesos y programas, así como la del ordenador y autorizado acceso de los usuarios a la información.

En este proyecto nos referiremos a seguridad lógica, ya que en lo que se refiere a seguridad física pertenece a otras instancias distintas de un programador Web.

1.3 Principios básicos

A grandes rasgos se entiende que mantener un sistema fiable consiste básicamente en garantizar tres aspectos:

- confidencialidad: nos dice que los objetos de un sistema han de ser accedidos únicamente por elementos autorizados a ello. Asegura el secreto de las comunicaciones contenidas en los mensajes.
- integridad: significa que los objetos sólo pueden ser modificados por elementos autorizados, y de una manera controlada. Hace referencia al hecho de que la información no pueda ser manipulada en el proceso de envío.
- disponibilidad: indica que los objetos del sistema tienen que permanecer accesibles a elementos autorizados.

Se ha de conseguir la seguridad más completa para los sistemas, para ello se deberá saber que se protege, contra quien se protege o en quien se puede confiar y en quien no. Se decidirá que empleados tienen acceso a que activos y porqué, que tipo de acceso se va a dar a cada persona de la organización que colabore con la empresa, así mismo se debe pensar en que tipo de acceso van a tener los posible clientes de la organización.

Se debe meditar lo más posible sobre quién y por que querría atacar a la organización, pero el perfil del atacante no hay que buscarlo solo en un usuario de Internet sino también dentro de la organización, ya que diferentes encuestas sobre pérdidas económicas [3], [4] dicen que:

1. Alrededor del 65% de los encuestados reconocen haber sido atacados en el último año.
2. Alrededor del 60% de los encuestados reconocen que algunos de sus ataques fueron realizados desde el interior de sus organizaciones.
3. Alrededor del 60% de los encuestados reconocen que algunos de sus ataques fueron realizados mediante conexiones desde la red Internet.

1.3.1 Tipos de atacantes más habituales:

Hacker: los hackers eran personas muy expertas en un sistema operativo, protocolos, etc. En caso de encontrarse fallos de seguridad de un sistema, lo notificaban al fabricante, incluso, a veces, facilitando una posible solución. Pero por distintas motivaciones

(venganza, ira, reto intelectual,...) algunas de estas personas ataca las redes o sistemas de una organización. Estas motivaciones pueden ser:

1. Empleados que son despedidos injustamente, desde su punto de vista y destruyen datos de la organización.
2. Empleados que entienden que son injustamente pagados, y alteran los registros de personal.
3. Hackers que roban mensajes de correo electrónico de unos políticos para otros políticos.
4. Individuos que acceden a sitios Web muy conocidos para cambiar su contenido.

Otro tipo de atacante es el **"amateur que juega"** cada vez más extendido, el perfil es el de una persona joven sin experiencia de sistemas ni de redes que usa herramientas automatizadas contra sistemas por Internet, para ver qué pasa. Una persona puede crear una aplicación que desconecte redes de Internet, valiéndose de las vulnerabilidades existentes en ellos, hacerla de fácil uso, crear un manual de usuario de la herramienta y publicarla en Internet, accesible a mucha gente a la vez, que la podrán usar de forma incontrolada para estas y otras muchas aplicaciones.

Finalmente se hablará de un tercer tipo de atacante, el menos numeroso pero el más peligroso, el **profesional.** Es el individuo que presta sus conocimientos y experiencia para atacar con un objetivo concreto, que puede ser el robo, la alteración de información con fines delictivos o el sabotaje. El profesional obtiene gran cantidad de información para estar bien seguro de cuales son los puntos débiles y por dónde puede empezar su trabajo. El siguiente paso suele ser el ataque de acceso a algún sistema, pero borrando sus huellas y sin alterar nada. Suele dejarse varios caminos de huida posibles, suele atacar mediante saltos y tener muy claro dónde, cuándo y durante cuánto tiempo estará seguro en su ataque.

Para saber como protegernos o qué tecnologías, mecanismos, sistemas concretos, procesos vamos a utilizar, se debe conocer dos temas de bastante complejidad:

1. los distintos tipos de ataques posibles.
2. las distintas defensas posibles.

Se sabe que nunca se van a conocer todos los distintos tipos de ataque posibles. Los tipos más importantes son:

1. Ataques para obtener información: son ataques que tienen como objetivo obtener información como direcciones IP de redes y sistemas, sistemas operativos de cada equipo, números de puertos abiertos, aplicaciones y sus versiones, contraseñas, etc.
2. Ataques de acceso no autorizado: son ataques de personas no autorizadas a los sistemas. Suelen ser pruebas de mayor interés que demostrarse a sí mismos que pueden llegar a ese sistema. Pueden ser peligrosos si el acceso se hace a través de una cuenta privilegiada del sistema.
3. Ataques con revelación de información: una vez se tiene el acceso anterior, se puede utilizar ese acceso para acceder a información secreta, con idea de aprovecharse de esa información, de borrarla o de modificarla con algún fin.
4. Ataques de denegación de servicio: Buscan dejar no disponible un sistema, un servicio o una red, habitualmente agotando el recurso, sea este el ancho de banda, espacio en disco, conexiones TCP, etc. Desafortunadamente, este tipo de ataque no suele necesitar ningún acceso previo a ningún sistema.

Estos son los ataques más importantes, pero cada vez más suelen ser combinaciones de varios de estos ataques.

En cuanto a las defensas posibles se destacan las más importantes:

1. Esquemas de seguridad de sistemas operativos: Especialmente para el caso de servidores son información muy relevante y de dispositivos de gestión de red, se deben mantener unos buenos esquemas de seguridad de ficheros, usuarios y aplicaciones, así como de servicios de red controlados desde tales sistemas.
2. Sistemas de identificación o autentificación seguros: hay muchos, desde las contraseñas hasta los sistemas biométricos, certificados digitales, tarjetas de identificación o distintas combinaciones. Se pueden usar tanto para acceso local a dispositivos como para acceso remoto.
3. Sistemas de cortafuegos (o firewalls): Sistemas de control de la información en forma de mensajes que entran o salen de una red. Son muy populares hoy en día y los hay de muchos tipos distintos.
4. Sistemas criptográficos: son sistemas que permiten, de varias formas distintas, mantener la integridad y la autentificación de mensajes o datos, así como la privacidad o confidencialidad de los mismos. Se integran en protocolos y parte de distintos sistemas operativos y aplicaciones, que usan algoritmos criptográficos.
5. Sistemas antivirus: son aplicaciones locales, o distribuidas, que permiten defenderse de los virus informáticos, nombre que reciben algunas aplicaciones utilizadas para cualquiera de los ataques citados arriba.
6. Sistemas de análisis de vulnerabilidades: son aplicaciones que permiten buscar en sistemas y aplicaciones instaladas distintos bugs o vulnerabilidades conocidas, para decidir si se corrigen, se cambian de versión o se dejan como están.
7. Sistemas de detección de intrusiones: son sistemas o aplicaciones que permiten, en tiempo real, detectar determinados tipos de ataques y alertar sobre ellos, a la vez que, en algunos casos, pueden pararlos.

Por último, se debe saber cuanto dinero se puede emplear en implantar y mantener el sistema de seguridad, se tendrá en cuenta el dinero que va a ser empleado en cada una de las siguientes tareas:

1. Adquisición de herramientas hardware y software, que implementen algunas de las defensas citadas.
2. El tiempo empleado en configuraciones y educar a los usuarios en su uso.
3. El tiempo empleado en la administración, mantenimiento y reconfiguración para permitir nuevos servicios, auditar las herramientas, etc.
4. El tiempo empleado en volver a una situación estable, después de la inconveniencia para los usuarios de alguno de los nuevos sistemas.

Finalmente se debería tratar de condensar todo este conocimiento en un análisis de riesgos que tendría 4 puntos clave:

1. Valorar los activos.
2. Entender todas las posible amenazas
3. Monitorizar y conocer todas las debilidades y vulnerabilidades del sistema.

4. Tratar de poner en marcha todas las medidas posibles para disminuir la probabilidad de tener pérdidas.

1.4 La seguridad en entornos Web

Las aplicaciones Web, por definición, permiten el acceso de usuarios a recursos centrales, el servidor Web y, a través de éste, a otros como los servidores de base de datos. Con los conocimientos y la implementación correcta de medidas de seguridad, se pueden proteger los recursos así como proporcionar un entorno seguro donde los usuarios trabajen cómodos con su aplicación.

Una aplicación Web, especialmente una que se despliega en Internet, es un objetivo mucho más atractivo para un atacante que una aplicación autónoma o cliente-servidor típica. Hay varias razones para esto:

- ***Disponibilidad y accesibilidad:*** Muchas aplicaciones Web están disponibles para los usuarios públicos en cualquier momento del día o de la noche. Como los servidores Web tienen que permitir el acceso a usuarios públicos y no tienen la protección completa de los cortafuegos típicos de una empresa.
- ***Familiaridad:*** La mayoría de los atacantes, incluso los menos sofisticados, conocen las interfaces Web. Un navegador Web es fácil de obtener y es uno de los programas de aplicación más comunes. El protocolo HTTP está bien definido, y existen muchas herramientas de hacking creados específicamente para ayudar a los atacantes a penetrar y comprometer aplicaciones Web.
- ***Facilidad:*** La configuración de un servidor Web, contenedor Web y aplicación Web para uso público es extremadamente compleja. Los atacantes, frecuentemente, pueden aprovechar esta complejidad y explotar deficiencias en la configuración de la aplicación o del sistema.
- ***Publicidad:*** El aliciente de algunos atacantes experimentados es la publicidad, la fama, el ego o un simple deseo de probar que pueden hacer algo que pocas otras personas pueden hacer. Desfigurar o comprometer un sito Web popular tiene mucho más valor desde la perspectiva de las “relaciones públicas” que comprometer una aplicación interna que utilizan unos cuantos empleados corporativos.

POLÍTICAS GENERALES DE SEGURIDAD

2.1 Introducción

El término política de seguridad se suele definir como el conjunto de requisitos definidos por los responsables directos o indirectos de un sistema que indica en términos generales qué está y qué no está permitido en el área de seguridad durante la operación general de dicho sistema. Al tratarse de términos generales, aplicables a situaciones o recursos muy diversos, suele ser necesario refinar los requisitos de la política para convertirlos en indicaciones precisas de qué es lo permitido y lo denegado en cierta parte de la operación del sistema, lo que se denomina política de aplicación específica.

La política de seguridad es la forma razonable de contestar, ordenadamente, a los distintos problemas de seguridad que pueden aparecer en las redes de las organizaciones.

La política de seguridad de una organización es algo así como las normas, reglas o leyes (escritas, cuanto más públicas, mejor) que rigen la vida de la organización en cuanto a qué se puede hacer y que no se puede hacer. Algunas de sus características son:

- Define el comportamiento apropiado para cada caso.
- Establece qué herramientas son necesarias y qué procedimientos.
- Sirve para comunicar un consenso del uso de datos y aplicaciones dentro de la organización.
- Proporciona una base para la demostración del uso inapropiado de recursos, por parte de empleados o de externos.

Una política de seguridad puede ser prohibitiva, si todo lo que no está expresamente permitido está denegado, o si todo lo que no está expresamente prohibido está permitido. Evidentemente la primera aproximación es mucho mejor que la segunda de cara a mantener la seguridad de un sistema, en este caso la política contemplará todas las actividades que se pueden realizar en los sistemas, y el resto serían consideradas ilegales.

Cualquier política ha de contemplar seis elementos claves en la seguridad de un sistema informático:

- Disponibilidad: Es necesario garantizar que los recursos del sistema se encontrarán disponibles cuando se necesitan, especialmente la información crítica.
- Utilidad: Los recursos del sistema y la información manejada en el mismo ha de ser útil para alguna función.
- Integridad: La información del sistema ha de estar disponible tal y como se almacenó por un agente autorizado.
- Autenticidad: El sistema ha de ser capaz de verificar la identidad de sus usuarios, y los usuarios la del sistema.
- Confidencialidad: La información sólo ha de estar disponible para agentes autorizados, especialmente su propietario.

- Posesión: Los propietarios de un sistema han de ser capaces de controlarlo en todo momento, perder este control en favor de un usuario malicioso compromete la seguridad del sistema hacia el resto de usuarios.

Para cubrir de forma adecuada los seis elementos anteriores, con el objetivo permanente de garantizar la seguridad corporativa, una política se suele dividir en puntos más concretos a veces llamados normativas. El estándar ISO 17799 [5] define las siguientes líneas de actuación:

- Seguridad organizacional: Aspectos relativos a la gestión de la seguridad dentro de la organización (cooperación con elementos externos, outsourcing, estructura del área de seguridad…).
- Clasificación y control de activos: Inventario de activos y definición de sus mecanismos de control, así como etiquetado y clasificación de la información corporativa.
- Seguridad del personal: Formación en materias de seguridad, cláusulas de confidencialidad, reporte de incidentes, monitorización de personal…
- Seguridad física y del entorno: Bajo este punto se engloban aspectos relativos a la seguridad física de los recintos donde se encuentran los diferentes recursos de la organización y de los sistemas en sí, así como la definición de controles genéricos de seguridad.
- Gestión de comunicaciones y operaciones: Este es uno de los puntos más interesantes desde un punto de vista estrictamente técnico, ya que engloba aspectos de la seguridad relativos a la operación de los sistemas y telecomunicaciones, como los controles de red, la protección frente a malware, la gestión de copias de seguridad o el intercambio de software dentro de la organización.
- Controles de acceso: Definición y gestión de puntos de control de acceso a los recursos informáticos de la organización: contraseñas, seguridad perimetral, monitorización de accesos…
- Desarrollo y mantenimiento de sistemas: Seguridad en el desarrollo y las aplicaciones, cifrado de datos, control de software...
- Gestión de continuidad de negocio: Definición de planes de continuidad, análisis de impacto, simulacros de catástrofes…
- Requisitos legales: Evidentemente, una política ha de cumplir con la normativa vigente en el país donde se aplica, si una organización se extiende a lo largo de diferentes países, su política tiene que ser coherente con la normativa del más restrictivo de ellos. En este apartado de la política se establecen las relaciones con cada ley, derechos de propiedad intelectual, tratamiento de datos de carácter personal, exportación de cifrado… junto a todos los aspectos relacionados con registros de eventos en los recursos (logs) y su mantenimiento.

2.2 ¿Qué es una política de seguridad?

El **EITF**(Emerging Issues task force) [6] define una política de seguridad como:

> "Una serie de sentencias formales (normas) que deben cumplir todas las personas que tengan acceso a cualquier información y/o tecnología de una organización"

El propósito general de una política de seguridad es informar a los usuarios, trabajadores y personal de dirección de los requisitos obligatorios para proteger los valores tecnológicos e información de la organización. La política debería especificar los mecanismos a través de los cuales estos requisitos puedan ser conocidos. Otro propósito es proporcionar una base para adquirir, configurar y auditar los sistemas de ordenadores y las redes. Por tanto, emplear un conjunto de herramientas de seguridad, sin una política de seguridad implícita, no tendría mucho sentido.

El uso adecuado de estas herramientas debe ser parte de la política de seguridad. Los objetivos concretos que se buscan vendrán determinados, en buena parte, por la capacidad que se tenga de cumplir con una serie de aspectos como son:

- Debe poderse implantar.
- Debe entenderse.
- Debe hacerse cumplir.
- Debe definir responsabilidades.
- Debe permitir que siga realizándose el trabajo normal.
- Debe se exhaustiva.
- Debe incluir mecanismos de respuesta.
- Debe tener mecanismos de actualización.
- Debe cumplir la legislación.

Se ha de contemplar siempre el **principio de privilegio mínimo**, que consiste en tratar de minimizar el número de usuarios con privilegios de administrador, el conjunto de equipos externos con acceso a sistemas locales y, en general, el número de situaciones en las que alguien o algo tienen privilegios de acceso, que no necesita, para que el trabajo salga adelante.

Otro aspecto que debe tratar de conseguirse es el ilustrado por los **principios de defensa en profundidad** y de **diversidad de defensa**. Se debe intentar tener más de un nivel de defensa y, a poder ser, de distinta naturaleza para, de esta forma, hacer más difícil el trabajo del supuesto atacante que no sólo debe vencer más de una defensa, sino que cada una le obliga a tener distintos tipos de conocimiento.

Si conseguimos tener un nivel de protección suficientemente seguro, debemos disponer de un **punto central de gestión** de la seguridad, en el que centralizar la gestión de la autenticación, autorización, del tráfico de seguridad, que soporte así mismo el registro de eventos centralizado y las alarmas.

Otra técnica es la de **identificar el punto** (o los puntos) **más débil** de la organización. Por ejemplo, cualquier configuración avanzada del cortafuegos más sofisticado del que se disponga es inútil si también se puede acceder a la organización mediante modem y una contraseña estática.

Seguir el **principio del cierre completo** que consiste en garantizar que, en el caso de ataque con éxito a un componente de seguridad, el sistema de seguridad no pasa a permitir el acceso completo a toda la red, sino a no permitir ya ningún acceso.

El más importante es el principio de simplicidad, que persigue que se cumplan todos los anteriores a la vez que se puede gestionar todo el sistema, de manera simple y

entendible. Además, es el principio que debe dirigir la propia construcción de cada norma, que no debería ser de un tamaño mayor de 2 páginas.

En la siguiente figura podemos ver como la política permite guiar el proceso de mantenimiento de seguridad:

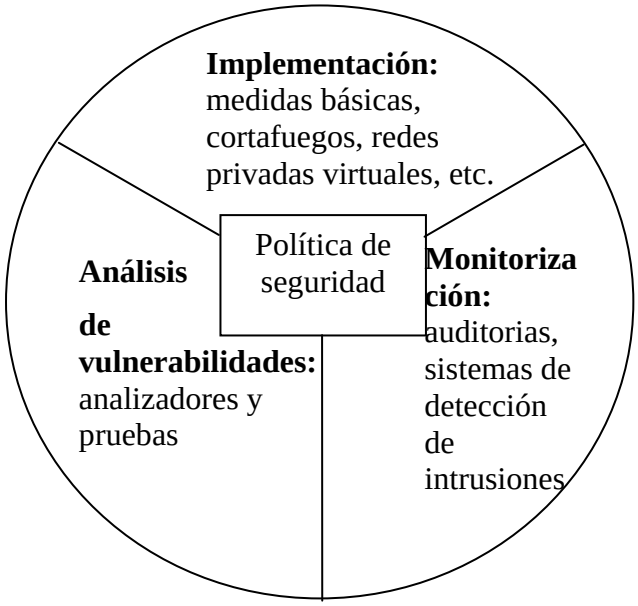

Figura 1: La seguridad como un proceso y la política como su guía.

Como resultado de la primera versión de la política de seguridad, se implementarán todos los procesos de seguridad física y lógica, lo que incluye, configuración de encaminadores, cortafuegos, redes privadas virtuales, etc. Esto se puede considerar como el primer paso del proceso, la **fase de puesta en marcha** del proceso de seguridad.

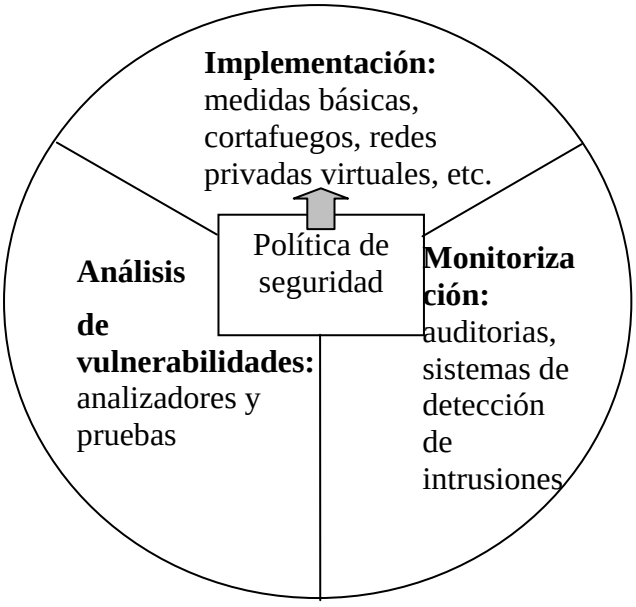

Figura 2: Fase de puesta en marcha del proceso de seguridad.

Si solo hiciéramos esto estaríamos incumpliendo normas básicas de una política de seguridad. El siguiente paso es la **fase de monitorización** de la red, en busca de:

- Incumplimientos de la política de seguridad.

- Posibles nuevas amenazas no tenidas en cuenta.

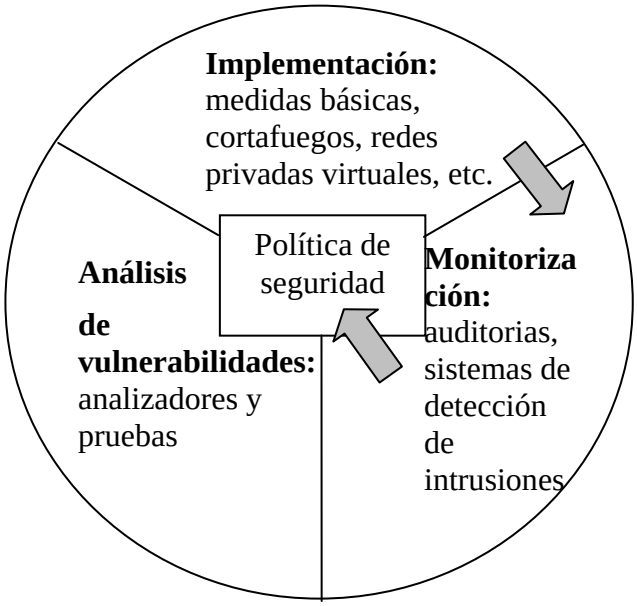

Figura 3: Fase de monitorización del proceso de seguridad.

Esta fase implica gestionar los registros de actividades y de auditoria y suele implicar la puesta en marcha, configuración y mantenimiento de lo que se llama sistemas de detección de intrusiones. Si en esta fase se deduce que hay que hacer algún cambio en la política de seguridad, para evitar incumplimientos o para tener en cuenta nuevas amenazas posibles, se estará empezando a crear la versión 2 de la política.

La siguiente fase sería la **fase de análisis de vulnerabilidades**, que sirve para buscar, mediante scanners, o analizadores de vulnerabilidades problemas relacionados con bugs.

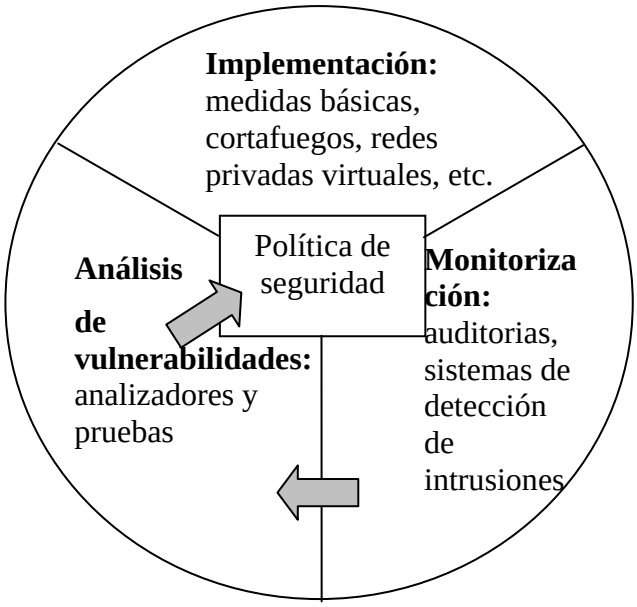

Figura 4: Fase de anàlisis de vulnerabilidades.

Finalmente, con todos los cambios consolidados, habrá que aplicar la nueva versión de la política de seguridad a todos los dispositivos que se vean involucrados, así como a los procedimientos necesarios.

Figura 5: Consolidación y continuación del proceso de seguridad.

A partir de aquí se repite la primera fase, y la rueda seguirá girando. La periodicidad con la que cada una de las fases debe ponerse en marcha depende de la política que tendrá en cuenta el tamaño de la red y otros múltiples factores. No se pretende defender tanto una periodicidad concreta como señalar la importancia de ver **la política de seguridad** de una organización **como algo en constante movimiento**, que permite que el mantenimiento de la seguridad sea **un proceso vivo y administrable** de forma estructurada y organizada.

2.3 Aspectos físicos de la política de seguridad.

Cualquier política de seguridad debe tener en cuenta una serie de procedimientos relacionados con la seguridad física, tanto en el aspecto del control de acceso físico a equipos, como en el de tener planes de contingencia y emergencia, así como de recuperación frente a desastres.

Debe estar claramente estipulado quién tiene derecho a acceder, y en qué casos concretos, a cada una de los siguientes dispositivos:

- Encaminadores, especialmente los de perímetro de seguridad.
- Conmutadores y concentradores.
- Servidores, concretamente aquellos que dispongan de la información más sensible de la organización.
- Cualquier tipo de cortafuegos.
- Cualquier tipo de punto extremo de una red privada virtual, que suelen ser, realmente, encaminadores y cortafuegos.
- Cualquier manejador de medios de almacenamiento, que permita acceder a soportes magnéticos con información sensible.

También debe haber unas normas claras sobre el control de acceso a los edificios donde estén situados los ordenadores y redes de la organización, identificando:

- ¿Quién puede entrar al edificio?
- ¿Quién puede entrar a determinadas salas del edificio, donde residan las máquinas identificadas como especialmente sensibles?
- ¿Cómo debe garantizarse tal tipo de acceso? Podría ser mediante algún tipo de token de acceso o mediante medidas biométricas, etc.
- ¿Quién puede acceder a determinados dispositivos específicos de Sali, como impresoras?
- ¿Qué documentos no deben tener copias en papel sueltas y dónde destruirlas y cómo destruirlas?
- ¿Qué acceso se le da a una persona que viene a colaborar, en términos de ordenador, cuenta, nivel de acceso? Lo mismo hay que tener en cuenta para una visita, si es que se diera el caso de necesitar acceso.
- ¿Es necesaria la implantación de cámaras de seguridad? Si lo es, ¿qué condiciones debe cumplir la empresa contratada a tal fin?
- ¿Es necesaria la presencia de guardias de seguridad?

Otra serie de ideas entre las cuales se pueden resaltar son:

- Las actividades críticas deben de situarse lejos de las áreas de acceso público.
- Los edificios deben ser discretos y se deben minimizar las indicaciones sobre su propósito, evitando signos obvios de las actividades realizadas en ellos.
- Los listines de los teléfonos y de las salas de la organización no deben identificar localizaciones informáticas (excepto las oficinas y áreas de recepción).
- Los materiales peligrosos y/o combustibles deben almacenarse a una distancia de seguridad del emplazamiento de los ordenadores. Un ejemplo muy simple es un vaso de agua, si es derramado en uno de los servidores podría causar graves daños a los servicios que realiza la empresa.
- El equipamiento de copias de seguridad y las propias copias de seguridad deben ubicarse en sitios diferentes y a una distancia conveniente de seguridad.
- Se debe instalar equipamiento apropiado de seguridad: detectores de calor y humos, salidas de emergencia y sistemas de extinción de incendios. Todo este equipamiento debe revisarse periódicamente de acuerdo con las instrucciones de los fabricantes. Los empleados deben estar entrenados en su uso adecuado.
- Los procedimientos de emergencia deben estar bien documentados y revisados regularmente.

Se debe señalar la importancia de disponer de los procedimientos para la copia periódica de la información, especialmente de la más relevante, que incluya, además, un cierto control de calidad, para asegurar que la copia de datos es inmediatamente utilizable. Se debe disponer de copias de:

- La información de ordenadores de los sistemas centrales.

- La información de ordenadores de la redes de área local.
- La información de aplicaciones y de bases de datos.

Podría ser necesario plantearse la necesidad de tener un centro en otro lugar físico, en el que residan todas estas copias, disponiendo así de una seguridad física adicional.

2.4 Aspectos lógicos de la política de seguridad.

Entre las normas y procedimientos relacionados con aspectos lógicos se puede separar lo que se denomina normas básicas o fundamentales, como:

- Política de uso aceptable.
- Política de acceso remoto.
- Política de protección de la información.
- Política de seguridad perimetral, aunque ésta implicará, también, procedimientos de configuración de encaminadores y de cortafuegos y puntos finales de redes privadas virtuales.
- Política de protección anti-virus.
- Política de contraseñas, de la que dependerán los procedimientos de servidores, estaciones de trabajo y acceso remoto y a través de redes privadas virtuales.
- Política de actuación frente a incidentes.

Entre las que no serían normas básicas, pero, aún así, en una organización extensa serían importantes, se tendrían:

- Política de uso de sistemas de detección de intrusiones, tanto de tipo red como de tipo *host*, de la que se hablará en extensión en el capítulo.
- Política de gestión de los *logs* de sistemas y de auditorias, de las que se hablará, también, en el capítulo.
- Política de administración de los laboratorios de seguridad, en los que, entre otras cosas, se harán todos los test posibles fuera de la red real, en búsqueda de posibles *bugs* de seguridad en sistemas de todo tipo, con ayuda de analizadores de vulnerabilidades y de herramientas propias.
- Política de comunicaciones gíreles.
- Política de uso de redes privadas virtuales.
- Cualquier otra política aplicable a las características de la organización.

Toda política de seguridad debe tener normas sobre uso aceptable, que definan el uso apropiado de los recursos informáticos de la organización. Los usuarios deberían leer y firmar tales normas, como parte del proceso de petición de cuentas de trabajo. Debe establecer claramente la responsabilidad de los usuarios con respecto a la protección de la información almacenada en sus cuentas. Debe señalar que permisos pueden tener los usuarios sobre ficheros que tengan accesibles, pero no sean suyos. Debe estipular el uso aceptable del correo electrónico, del acceso Web y de todo tipo de acceso a Internet, así como discutir los usos aceptables, no relacionados con el objeto de la organización, de los recursos informáticos.

Otra política es la que hace explícitas las normas sobre acceso remoto a los recursos informáticos de la organización. Debe cubrir todos los métodos disponibles para acceder, remotamente, a los recursos de la red interna:

- Acceso mediante modem, vía SLIP o PPP.
- Acceso mediante RDSI o ADSL.
- Acceso mediante ssh o telnet, desde Internet.
- Acceso mediante cualquier tipo de red privada virtual.

Otra política fundamental es la que trata de la protección de la información, que debe dar una guía sobre el procesado, almacenamiento y transmisión de la información, por parte de los usuarios. Su objetivo fundamental es garantizar que la información está protegida apropiadamente frente a la posible modificación. Otro aspecto importante que debe cubrir es la definición de los niveles de sensibilidad de la información de la organización, qué información es pública, cuál es semi-pública y cuál está restringida y a qué niveles.

La siguiente política es la de protección frente a posibles ataques por virus informáticos. Esta, debe proporcionar las líneas generales de los informes sobre infecciones por virus, así como los procedimientos de contención de los mismos. También debe contener las explicaciones de los distintos niveles de riesgo dependiendo del tipo de virus, así como la necesidad de selección del tipo de programa antivirus que se vaya a utilizar y debe discutirse la frecuencia de actualización de los datos del mismo software.

Otra política tiene que ver con las contraseñas. Debe contener las directrices de cómo gestionar las contraseñas de usuario y de administrador, así como:

- Las reglas de creación de las contraseñas.
- Las reglas de cómo evitar la revelación de contraseñas.
- Las reglas para el desarrollo de aplicaciones en las que haga falta la contraseña.
- Las reglas de uso de todo tipo de contraseñas de protocolos.

2.5 Aspectos humanos y organizativos de la política de seguridad.

Se puede hacer la siguiente división del personal que interactuará con los sistemas y máquinas de la organización:

- El administrador o responsable directo del sistema.
- Las personas que deben tener acceso al sistema como usuarios.
- Las personas relacionadas con el sistema peor que no necesitan usarlo.
- Las personas ajenas al sistema.

Los administradores deben ser personas correctamente cualificadas, serán los únicos con acceso completo al sistema que administran. Es fundamental asegurar a esta persona tan bien como se asegure al resto del sistema. No sirve de nada disponer del sistema más sofisticado, desde el punto de vista de la seguridad, si esta persona puede fallar en cualquier momento. La única forma de conseguirlo es buscando su fidelidad, tanto mediante recompensas como penalizaciones adecuadas y proporcionadas al valor de los bienes que custodia. No hacerlo así sería exponer a la organización a problemas realmente graves. Si tales personas reciben un salario insuficiente, o inapropiado, y una penalización

insuficiente, esto les hará más susceptibles al chantaje o al soborno, mientras que unas penalizaciones excesivas pueden llevar a la desesperación, y pérdida del miedo a las consecuencias, o a un estado de nerviosismo permanente, que disminuya su capacidad.

Es posible reducir la responsabilidad de los administradores, transfiriéndola a sus jefes, reduciendo así, aparentemente, la necesidad de cuidar de los administradores. En realidad, esto no es más que una transferencia ficticia: todo el que tiene acceso físico al ordenador tiene en última instancia control absoluto, y este traslado de responsabilidades sólo refleja una falta de confianza notoria, que rompe la necesaria relación de cordialidad y fidelidad, poniendo en peligro el sistema de seguridad.

Los usuarios, típicamente empleados fijos, personal temporal o clientes de un sistema no necesitan tener el control absoluto de los sistemas servidores de la organización. Por ello, la solución generalmente usada es establecer sistemas de acceso parciales, con requisitos de seguridad menores, de tal forma que el usuario tenga control absoluto sobre su sistema particular, disminuyendo la repercusión de un eventual compromiso y la importancia de la responsabilidad, pero permitiendo sólo el acceso necesario a los sistemas servidores.

No todos los usuarios deben tener el mismo grado de acceso a los datos, ni tendrán, por tanto, la misma responsabilidad, por ello se debe establecer distintos niveles de usuarios. Cada usuario debe de ser consciente de sus responsabilidades.

Entre las personas relacionadas con los sistemas, especialmente servidores de la organización, pero que no necesitan usarlos habitualmente, se puede hacer una división, personal ejecutivo y personal de mantenimiento.

El personal ejecutivo, debido a su disposición dentro de la empresa, no suele necesitar acceder a los sistemas servidores en el día a día, pero, en determinados momentos, pueden necesitar mostrar datos o información de los mismos. Esto, sin lugar a dudas, pone en peligro la seguridad del sistema y, por tanto será necesario, tomar las medidas adecuadas y habilitar los mecanismos que lo permitan, manteniendo, a la vez, las normas de seguridad sobre el sistema y los datos de que se trate.

El personal de mantenimiento suele tener acceso al perímetro protegido. En ellos confluyen circunstancias especiales: por un lado suelen ser personas no técnicas, ni relacionadas con los equipos y, por tanto, es más probables que cometan errores comprensibles en su interacción con ellos. Por otro lado, el que no se les exija conocimientos en tecnología para desempeñar su trabajo no quiere decir que no los tengan. Por ello, si deben acceder al sistema, es recomendable que lo hagan bajo supervisión de una persona responsable.

Para las personas ajenas a los sistemas, es muy fácil resumir la política: no deben acceder, en ninguna circunstancia. Una buena política de seguridad debe determinar el acceso, controlado correctamente, de las personas con derecho a tal acceso, pero debe garantizar, claramente, la imposibilidad para el resto de las personas.

Hay una serie de procedimientos organizativos a tener en cuenta, en relación con todas las personas de la organización, que deben ser tomados por la dirección de recursos humanos, o semejante, que se pueden resumir de la siguiente manera:

- Que las definiciones de puesto de trabajo contemplen todo lo necesario en cuanto a responsabilidades de seguridad y sus sistemas.

- Que los empleados que vayan a hacer uso de información sensible, hayan sido correctamente contratados, teniendo especial cuidado con el caso de los empleados temporales.
- Tanto los nuevos empleados, como los colaboradores externos o consultores o técnicos deberán firmar contratos, o acuerdos, de confidencialidad antes de su conexión y acceso a los recursos y sistemas de información de la organización.
- Es necesario que tales empleados se den cuenta de la importancia que tienen en el mantenimiento de la seguridad de sistemas y equipo y de los posibles procedimientos disciplinarios a los que se verán expuestos, en el caso de incumplir sus responsabilidades. La mejor manera de implementar este procedimiento es mediante un plan de formación para todo el personal de la organización, adecuándolo a las necesidades de cada puesto.

2.6 Aspectos legales de la política de seguridad.

Se deben señalar las normas legales que hay que tener en cuenta, hoy en día, en el desarrollo de cualquier política de seguridad, así como una serie de puntos clave, que deben aparecer en ella, como consecuencia de la aplicación de tales normas.

Las normas vigentes, que habrá que tener en cuenta, son:

- Real decreto 994/1999, reglamento de medidas de seguridad de ficheros con datos personales.
- Ley orgánica de protección de datos 15/1999(LOPD).
- Ley de servicios de la sociedad de la información y de comercio electrónico, LSSICE, 34/2002.

Con respecto al real decreto 94/1999, hay que tener en cuenta, entre otras, una serie de normas de obligado cumplimiento:

- Según su artículo 11.

> "El responsable de Seguridad de la Dirección de Sistemas, los Responsables de Seguridad designados en otras Direcciones y los Encargados de tratamiento por terceros, serán los encargados de coordinar y controlar las medidas de seguridad definidas para los ficheros, incluyendo mediante la utilización de las herramientas necesarias, la administración en la asignación a los usuarios y ficheros en los niveles y categorías de seguridad requeridas por el Propietario de los datos para su aplicación en cuanto al acceso y tratamiento de los datos se refiere".

- También en el artículo 11.

> "La Dirección de Sistemas, establecerá un mecanismo que permita la identificación de forma inequívoca y personalizada de todo aquel usuario que intente acceder al sistemas de información y la verificación de que está autorizado".

- Según sus artículos 2 y 18.

  > "Se limitará la posibilidad de intentar reiteradamente el acceso no autorizado al sistema de información".

- Según el artículo 24.

  > "De cada acceso se guardarán, como mínimo la identificación del usuario, la fecha y hora en que se realizó, el fichero accedido, el tipo de acceso y si ha sido autorizado o denegado. En caso de que el acceso haya sido autorizado, se debe guardar la información que permita identificar el registro accedido. Los mecanismos que permiten el registro de lo datos detallados, estarán bajo el control del Responsable de Seguridad. En ningún caso se permitirá la desactivación de los mismos. El período mínimo de conservación es de 2 años. El Responsable de Seguridad se encargará de revisar periódicamente la información de control registrada y elaborará un informe de revisiones realizadas y problemas detectados al menos 1 vez al mes".

- Según el artículo 17.

  > "Los sistemas de información e instalaciones de tratamiento de datos se someterán a una auditoria interna o externa, que verifique el cumplimento del Reglamento, de los procedimientos e instrucciones vigentes en materia de seguridad de datos, al menos cada dos años".

- Según el artículo 5.

  > "En el acceso a datos a través de redes de comunicaciones, el nivel de seguridad debe ser equivalente a los accesos en modo local".

Por otro lado esta la LOPD, cuyo objetivo es garantizar y proteger, en lo que concierne al tratamiento de los datos personales, las libertades públicas y los derechos fundamentales de las personas físicas, y especialmente de su honor e intimidad personal y familiar.

En cuanto a la LOPD, se debe destacar [7]:

- Artículo 4:
  - Los datos de carácter personal sólo se podrán recoger para su tratamiento, así como someterlos a dicho tratamiento, cuando sean adecuados, pertinentes y no excesivos en relación con el ámbito y las finalidades determinadas, explícitas y legítimas para las que se hayan obtenido.
  - Los datos de carácter personal objeto de tratamiento no podrán usarse para finalidades incompatibles con aquellas para las que los datos hubieran sido recogidos. No se considerará incompatible el tratamiento posterior de éstos con fines históricos, estadísticos o científicos.
  - En el caso de que los datos sena inexactos, deberán ser rectificados o cancelados, reconociendo en todo caso la posibilidad de acceso por parte del interesado.
  - Si los datos pierden su finalidad originaria deberán ser cancelados.

- Artículo 5:

  En este artículo se trata la recogida de datos, siendo de especial importancia para las personas que han de confeccionar los cuestionarios de recogida de datos, dado que los interesados deberán ser informados de lo siguiente:

  a) La existencia del fichero o tratamiento de datos, la finalidad de la recogida y los destinatarios de la información.

  b) El carácter optativo u obligatorio de las preguntas.

  c) Las consecuencias sobre la obtención de los datos, así como la negativa a suministrarlos.

  d) La posibilidad de ejercer los derechos de acceso, rectificación, cancelación y oposición.

  e) La identidad y dirección del responsable del tratamiento o su representante.

- Artículo 6:

  En este artículo se trata el consentimiento del afectado a la hora de dar sus datos. No obstante, sin perjuicio de las normas más rigurosas para los datos especialmente protegidos, existen algunas excepciones al principio general, dado que, siempre que no se vulneren los derechos y libertades fundamentales del interesado, no será preciso el consentimiento en los siguientes supuestos:

  a) Cuando los datos se recojan para el ejercicio de las funciones propias de las Administraciones públicas en el ámbito de sus competencias.

  b) Cuando se refieran a las partes en una relación comercial, laboral o administrativa, siempre que sea necesario para el cumplimiento de la relación de que se trate.

  c) Cuando el tratamiento de los datos tenga por finalidad proteger un interés vital del interesado.

  d) Cuando los datos figuren en fuentes accesibles al público y su tratamiento sea necesario para la satisfacción del interés legítimo perseguido.

  De todas formas, en los casos en que no es necesario el consentimiento del afectado, se reconoce a éste el derecho a quedar excluido del tratamiento de los datos, siempre que una ley no disponga lo contrario y existan motivos fundados y legítimos relativos a una concreta situación personal.

- Artículo 7:

  En este artículo se trata los datos especialmente protegidos, como pueden ser los relacionados con ideologías, origen racial, salud, infracciones penales o administrativas, etc.

- Artículo 9:

  En este artículo se trata de la seguridad de los datos:

a) El responsable del fichero, y en su caso, el encargado del tratamiento deberán adoptar las medidas de índole técnica y organizativa necesarias que garanticen la seguridad de los datos y eviten su alteración, perdida o acceso no autorizado.

b) No se registrarán datos de carácter personal en ficheros que no reúnan las condiciones de integridad seguridad necesarias.

- Artículos 11 y 12:

En estos artículos se trata la comunicación de los datos y los accesos de terceras personas a esos datos.

Resaltar lo más importante de la LSSICE sería mucho más extenso. Basta señalar su objetivo y ámbito, lo que da una idea de lo importante que será tenerla en cuenta para la elaboración de cualquier política de seguridad.

El objetivo de la ley es:

"Es objeto de la presente Ley la regulación del régimen jurídico de los servicios de la sociedad de la información y de la contratación por vía electrónica, en lo referente a las obligaciones de los prestadores de servicios, incluidos los que actúan como intermediarios en la transmisión de contenidos por las redes de telecomunicaciones, las comunicaciones comerciales por vía electrónica, la información previa y posterior a la celebración de contratos electrónicos, las condiciones relativas a su validez y eficacia y el régimen sancionador aplicable a los prestadores de servicios de la sociedad de la información".

Es una ley joven y está en el periodo de puesta en macha. Habrá mucho más que decir cuando lleve un tiempo utilizándose y hayan aparecido realmente problemas.

MÉTODOS DE ATAQUE Y SOLUCIONES

3.1 Introducción

En este capítulo se va a tratar de clasificar los distintos tipos de ataques, atendiendo a criterios diversos, pero analizándolos en detalle, desde el punto de vista del atacante "profesional".

3.2 Elementos de la seguridad

Los tres elementos principales a proteger en cualquier sistema informático son:

- El software.
- El hardware.
- Los datos.

Por hardware se entiende el conjunto formado por todos los elementos físicos de un sistema informático, como CPU´s, terminales, cableado, medios de almacenamiento secundario (cintas, CD-ROMs, diskettes…) o tarjetas de red. Por software se entiende el conjunto de programas lógicos que hacen funcionar al hardware, tanto sistemas operativos como aplicaciones, y por datos el conjunto de información lógica que manejan el software y el hardware, como por ejemplo paquetes que circulan por un cable de red o entradas de una base de datos. Aunque generalmente en las auditorias de seguridad se habla de un cuarto elemento a proteger, los fungibles (elementos que se gastan o desgastan con el uso continuo, como papel de impresora, tóners, cintas magnéticas, diskettes…), en este proyecto no se considera la seguridad de estos elementos por ser externos al sistema.

Habitualmente los datos constituyen el principal elemento de los tres a proteger, ya que es el más amenazado y seguramente el más difícil de recuperar, con toda seguridad una máquina Unix está ubicada en un lugar de acceso físico restringido, o al menos controlado, y además en caso de pérdida de una aplicación (o un programa de sistema, o el propio núcleo de Unix) este software se puede restaurar sin problemas desde su medio original. Sin embargo, en caso de pérdida de una base de datos o de un proyecto de un usuario, no tenemos un medio original desde el que restaurar, se ha de pasar obligatoriamente por un sistema de copias de seguridad, y a menos que la política de copias sea muy estricta, es difícil devolver los datos al estado en que se encontraban antes de la pérdida. Contra cualquiera de los tres elementos descritos anteriormente (pero principalmente sobre los datos) se pueden realizar multitud de ataques o, dicho de otra forma, están expuestos a diferentes amenazas.

3.3 Amenazas

3.3.1 Tipos de amenazas.

Generalmente, la taxonomía más elemental de estas amenazas se divide en cuatro grandes grupos:

- Interrupción
- Interceptación
- Modificación

- Fabricación.

Un ataque se clasifica como interrupción si hace que un objeto del sistema se pierda, quede inutilizable o no disponible, éste es un ataque contra la disponibilidad. Se tratara de una interceptación si un elemento no autorizado consigue un acceso a un determinado objeto del sistema, éste es un ataque contra la confidencialidad y de una modificación si además de conseguir el acceso consigue modificar el objeto, algunos autores consideran un caso especial de la modificación: la destrucción, entendiéndola como una modificación que inutiliza al objeto afectado. Por último, se dice que un ataque es una fabricación si se trata de una modificación destinada a conseguir un objeto similar al atacado de forma que sea difícil distinguir entre el objeto original y el fabricado. En la figura 6 se muestran estos tipos de ataque de una forma gráfica.

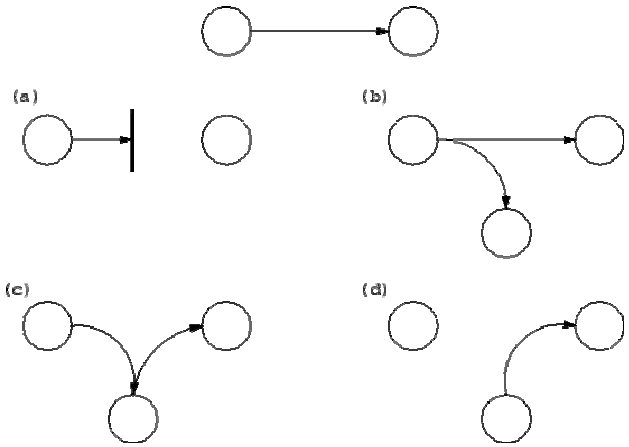

Figura 6: Flujo normal de información entre emisor y receptor y posibles amenazas: (a) interrupción, (b) interceptación, (c) modificación y (d) fabricación.

En la gran mayoría de publicaciones relativas a la seguridad informática en general, y especialmente en las relativas a seguridad en Unix, tarde o temprano se intenta clasificar en grupos a los posibles elementos que pueden atacar nuestro sistema. Con frecuencia, se suele identificar a los atacantes únicamente como personas, esto tiene sentido si hablamos por ejemplo de responsabilidades por un delito informático. Pero en este proyecto es preferible hablar de *elementos* y no de personas, aunque a veces lo olvidemos, nuestro sistema puede verse perjudicado por múltiples entidades aparte de humanos, como por ejemplo programas, catástrofes naturales…

3.3.2 Origen de las amenazas.

A continuación se presenta una relación de los elementos que potencialmente pueden amenazar a nuestro sistema.

3.3.2.1 Personas

La mayoría de ataques al sistema van a provenir en última instancia de personas que, intencionada o inintencionadamente, pueden causar enormes pérdidas. Generalmente se trataría de piratas que intentan conseguir el máximo nivel de privilegio posible aprovechando alguno (o algunos) de los riesgos lógicos de los que hablaremos a continuación, especialmente agujeros del software. Pero con demasiada frecuencia se suele olvidar que los piratas clásicos no son los únicos que amenazan nuestros equipos, es especialmente preocupante que mientras que hoy en día cualquier administrador mínimamente preocupado por la seguridad va a conseguir un sistema relativamente fiable de una forma lógica (permaneciendo atento a vulnerabilidades de su software,

restringiendo servicios, utilizando cifrado de datos…), pocos administradores tienen en cuenta factores como la ingeniería social a la hora de diseñar una política de seguridad.

Aquí se describen brevemente los diferentes tipos de personas que de una u otra forma pueden constituir un riesgo para nuestros sistemas, generalmente se dividen en dos grandes grupos, los atacantes pasivos, aquellos que "se pasean" por el sistema pero no lo modifican o destruyen, y los activos, aquellos que dañan el objetivo atacado, o lo modifican en su favor. Generalmente los curiosos y los crackers realizan ataques pasivos (que se pueden convertir en activos), mientras que los terroristas y ex-empleados realizan ataques activos puros, los intrusos remunerados suelen ser atacantes pasivos si nuestra red o equipo no es su objetivo, y activos en caso contrario, y el personal realiza ambos tipos indistintamente, dependiendo de la situación concreta.

- Personal: Las amenazas a la seguridad de un sistema provenientes del personal de la propia organización rara vez son tomadas en cuenta, se presupone un entorno de confianza donde a veces no existe, por lo que se pasa por alto el hecho de que casi cualquier persona de la organización, incluso el personal ajeno a la infraestructura informática (secretariado, personal de seguridad, personal de limpieza y mantenimiento…) puede comprometer la seguridad de los equipos. Aunque los ataques pueden ser intencionados (en cuyo caso sus efectos son extremadamente dañinos, recordemos que nadie mejor que el propio personal de la organización conoce mejor los sistemas… y sus debilidades), lo normal es que más que de ataques se trate de accidentes causados por un error o por desconocimiento de las normas básicas de seguridad, un empleado de mantenimiento que corta el suministro eléctrico para hacer una reparación puede llegar a ser tan peligroso como el más experto de los administradores que se equivoca al teclear una orden y borra todos los sistemas de ficheros, y en el primer caso, el atacante ni siquiera ha de tener acceso lógico (ni físico) a los equipos, ni conocer nada sobre seguridad en Unix.
- Ex-empleados: Otro gran grupo de personas potencialmente interesadas en atacar nuestro sistema son los antiguos empleados del mismo, especialmente los que no abandonaron el entorno por voluntad propia (y en el caso de redes de empresas, los que pasaron a la competencia). Generalmente, se trata de personas descontentas con la organización que pueden aprovechar debilidades de un sistema que conocen perfectamente para dañarlo como venganza por algún hecho que no consideran justo, pueden insertar troyanos, bombas lógicas, virus… o simplemente conectarse al sistema como si aún trabajaran para la organización (muchas veces se mantienen las cuentas abiertas incluso meses después de abandonar la universidad o empresa), conseguir el privilegio necesario, y dañarlo de la forma que deseen, incluso chantajeando a sus ex-compañeros o ex-jefes.
- Curiosos: Junto con los crackers, los curiosos son los atacantes más habituales de sistemas Unix en redes de I+D. Recordemos que los equipos están trabajando en entornos donde se forma a futuros profesionales de la informática y las telecomunicaciones (gente que a priori tiene interés por las nuevas tecnologías), y recordemos también que las personas suelen ser curiosas por naturaleza, esta combinación produce una avalancha de estudiantes o personal intentando conseguir mayor privilegio del que tienen o intentando acceder a sistemas a los que oficialmente no tienen acceso. Y en la mayoría de ocasiones esto se hace simplemente para leer el correo de un amigo, enterarse de cuánto cobra un compañero, copiar un trabajo o comprobar que es posible romper la seguridad de

un sistema concreto. Aunque en la mayoría de situaciones se trata de ataques no destructivos (a excepción del borrado de huellas para evitar la detección), parece claro que no benefician en absoluto al entorno de fiabilidad que podamos generar en un determinado sistema.

- Crackers: Los entornos de seguridad media son un objetivo típico de los intrusos, ya sea para fisgonear, para utilizarlas como enlace hacia otras redes o simplemente por diversión. Por un lado, son redes generalmente abiertas, y la seguridad no es un factor tenido muy en cuenta en ellas, por otro, el gran número y variedad de sistemas Unix conectados a estas redes provoca, casi por simple probabilidad, que al menos algunos de sus equipos sean vulnerables a problemas conocidos de antemano. De esta forma un atacante sólo ha de utilizar un escáner de seguridad contra el dominio completo y luego atacar mediante un simple exploit los equipos que presentan vulnerabilidades, esto convierte a las redes de I+D, a las de empresas, o a las de ISPs en un objetivo fácil y apetecible para piratas con cualquier nivel de conocimientos, desde los más novatos (y a veces más peligrosos) hasta los expertos, que pueden utilizar toda la red para probar nuevos ataques o como nodo intermedio en un ataque a otros organismos, con el consiguiente deterioro de imagen (y a veces de presupuesto) que supone para una universidad ser, sin desearlo, un apoyo a los piratas que atacan sistemas teóricamente más protegidos, como los militares.
- Terroristas: Bajo esta definición se engloba a cualquier persona que ataca al sistema simplemente por causar algún tipo de daño en él. Por ejemplo, alguien puede intentar borrar las bases de datos de un partido político enemigo o destruir los sistemas de ficheros de un servidor que alberga páginas Web de algún grupo religioso, en el caso de redes de I+D, típicos ataques son la destrucción de sistemas de prácticas o la modificación de páginas Web de algún departamento o de ciertos profesores, generalmente por parte de alumnos descontentos.
- Intrusos remunerados: Este es el grupo de atacantes de un sistema más peligroso, aunque por fortuna el menos habitual en redes normales, suele afectar más a las grandes empresas o a organismos de defensa. Se trata de piratas con gran experiencia en problemas de seguridad y un amplio conocimiento del sistema, que son pagados por una tercera parte generalmente para robar secretos (el nuevo diseño de un procesador, una base de datos de clientes, información confidencial sobre las posiciones de satélites espía…) o simplemente para dañar la imagen de la entidad afectada. Esta tercera parte suele ser una empresa de la competencia o un organismo de inteligencia, es decir, una organización que puede permitirse un gran gasto en el ataque, de ahí su peligrosidad, se suele pagar bien a los mejores piratas, y por si esto fuera poco los atacantes van a tener todos los medios necesarios a su alcance. Aunque como hemos dicho los intrusos remunerados son los menos comunes en la mayoría de situaciones, en ciertas circunstancias pueden aprovechar nuestras redes como plataforma para atacar otros organismos.

3.3.2.2 Amenazas lógicas

Bajo la etiqueta de *amenazas lógicas* encontramos todo tipo de programas que de una forma u otra pueden dañar a nuestro sistema, creados de forma intencionada para ello (software malicioso, también conocido como malware) o simplemente por error (bugs o agujeros) [8].

- Software incorrecto: Las amenazas más habituales a un sistema Unix provienen de errores cometidos de forma involuntaria por los programadores de sistemas o de aplicaciones. Una situación no contemplada a la hora de diseñar el sistema de red del kernel o un error accediendo a memoria en un fichero pueden comprometer local o remotamente a Unix (o a cualquier otro sistema operativo). A estos errores de programación se les denomina bugs, y a los programas utilizados para aprovechar uno de estos fallos y atacar al sistema, exploits. Como se ha dicho, representan la amenaza más común contra Unix, ya que cualquiera puede conseguir un exploit y utilizarlo contra nuestra máquina sin ni siquiera saber como funciona y sin unos conocimientos mínimos de Unix, incluso hay exploits que dañan seriamente la integridad de un sistema (negaciones de servicio o incluso acceso root remoto) y están preparados para ser utilizados desde MS-DOS, con lo que cualquier pirata novato (comúnmente, se les denomina script kiddies) puede utilizarlos contra un servidor y conseguir un control total de una máquina de varios millones de pesetas desde su PC sin saber nada del sistema atacado, incluso hay situaciones en las que se analizan los logs de estos ataques y se descubre que el pirata incluso intenta ejecutar órdenes de MS-DOS.
- Herramientas de seguridad: Cualquier herramienta de seguridad representa un arma de doble filo, de la misma forma que un administrador las utiliza para detectar y solucionar fallos en sus sistemas o en la subred completa, un potencial intruso las puede utilizar para detectar esos mismos fallos y aprovecharlos para atacar los equipos. Herramientas como nessus, saintt o satan pasan de ser útiles a ser peligrosas cuando las utilizan crackers que buscan información sobre las vulnerabilidades de un host o de una red completa. Si como administradores no se utilizan herramientas de seguridad que muestren las debilidades de los sistemas (para corregirlas), hay que estar seguro que un atacante no va a dudar en utilizar tales herramientas (para explotar las debilidades encontradas), por tanto, hay que agradecer a los diseñadores de tales programas el esfuerzo que han realizado (y nos han ahorrado) en pro de sistemas más seguros.
- Puertas traseras: Durante el desarrollo de aplicaciones grandes o de sistemas operativos es habitual entre los programadores insertar “atajos” en los sistemas habituales de autenticación del programa o del núcleo que se está diseñando. A estos atajos se les denomina puertas traseras, y con ellos se consigue mayor velocidad a la hora de detectar y depurar fallos, por ejemplo, los diseñadores de un software de gestión de bases de datos en el que para acceder a una tabla se necesiten cuatro claves diferentes de diez caracteres cada una pueden insertar una rutina para conseguir ese acceso mediante una única clave “especial”, con el objetivo de perder menos tiempo al depurar el sistema. Algunos programadores pueden dejar estos atajos en las versiones definitivas de su software para facilitar un mantenimiento posterior, para garantizar su propio acceso, o simplemente por descuido, la cuestión es que si un atacante descubre una de estas puertas traseras (no nos importa el método que utilice para hacerlo) va a tener un acceso global a datos que no debería poder leer, lo que obviamente supone un grave peligro para la integridad de nuestro sistema.
- Bombas lógicas: Las bombas lógicas son partes de código de ciertos programas que permanecen sin realizar ninguna función hasta que son activadas, en ese punto, la función que realizan no es la original del programa, sino que generalmente se trata de una acción perjudicial. Los activadores más comunes de estas bombas lógicas pueden ser la ausencia o presencia de ciertos ficheros, la ejecución bajo un

determinado UID o la llegada de una fecha concreta, cuando la bomba se activa va a poder realizar cualquier tarea que pueda realizar la persona que ejecuta el programa, si las activa el root, o el programa que contiene la bomba los efectos obviamente pueden ser fatales.

- Canales cubiertos: Los canales cubiertos son canales de comunicación que permiten a un proceso transferir información de forma que viole la política de seguridad del sistema, es decir, un proceso transmite información a otros (locales o remotos) que no están autorizados a leer dicha información. Los canales cubiertos no son una amenaza demasiado habitual en redes de I+D, ya que suele ser mucho más fácil para un atacante aprovechar cualquier otro mecanismo de ataque lógico, sin embargo, es posible su existencia, y en este caso su detección suele ser difícil, algo tan simple como el puerto finger abierto en una máquina puede ser utilizado a modo de covert channel por un pirata con algo de experiencia.
- Virus: Un virus es una secuencia de código que se inserta en un fichero ejecutable (denominado huésped), de forma que cuando el archivo se ejecuta, el virus también lo hace, insertándose a sí mismo en otros programas. Todo el mundo conoce los efectos de los virus en algunos sistemas operativos de sobremesa, sin embargo, en Unix los virus no suelen ser un problema de seguridad grave, ya que lo que pueda hacer un virus lo puede hacer más fácilmente cualquier otro mecanismo lógico. Aunque los virus existentes para entornos Unix son más una curiosidad que una amenaza real, en sistemas sobre plataformas IBM-PC o compatibles (hay muchos sistemas Unix que operan en estas plataformas, como Linux, FreeBSD, NetBSD, Minix, Solaris…) ciertos virus, especialmente los de boot, pueden tener efectos nocivos, como dañar el sector de arranque, aunque se trata de daños menores comparados con los efectos de otras amenazas.
- Gusanos: Un gusano es un programa capaz de ejecutarse y propagarse por sí mismo a través de redes, en ocasiones portando virus o aprovechando bugs de los sistemas a los que conecta para dañarlos. Al ser difíciles de programar su número no es muy elevado, pero el daño que pueden causar es muy grande, el mayor incidente de seguridad en Internet fue precisamente el gusano Word comentado en el capitulo 1. Hemos de pensar que un gusano puede automatizar y ejecutar en unos segundos todos los pasos que seguiría un atacante humano para acceder a nuestro sistema, mientras que una persona, por muchos conocimientos y medios que posea, tardara como mínimo horas en controlar nuestra red completa (un tiempo más que razonable para detectarlo), un gusano puede hacer eso mismo en pocos minutos, de ahí su enorme peligro y sus devastadores efectos.
- Caballos de Troya: Los troyanos o caballos de Troya son instrucciones escondidas en un programa de forma que éste parezca realizar las tareas que un usuario espera de él, pero que realmente ejecute funciones ocultas (generalmente en detrimento de la seguridad) sin el conocimiento del usuario, como el Caballo de Troya de la mitología griega, al que deben su nombre, ocultan su función real bajo la apariencia de un programa inofensivo que a primera vista funciona correctamente. En la práctica totalidad de los ataques a Unix, cuando un intruso consigue el privilegio necesario en el sistema instala troyanos para ocultar su presencia o para asegurarse la entrada en caso de ser descubierto. por ejemplo, es típico utilizar lo que se denomina un rootkit, que no es más que un conjunto de versiones troyanas de ciertas utilidades (netstat, ps, who…), para conseguir que cuando el administrador las ejecute no vea la información relativa al atacante, como sus procesos o su

conexión al sistema, otro programa que se suele suplantar es login, por ejemplo para que al recibir un cierto nombre de usuario y contraseña proporcione acceso al sistema sin necesidad de consultar `/etc/passwd`.

- Programas conejo o bacterias: Bajo este nombre se conoce a los programas que no hacen nada útil, sino que simplemente se dedican a reproducirse hasta que el número de copias acaba con los recursos del sistema (memoria, procesador, disco...), produciendo una negación de servicio. Por sí mismos no hacen ningún daño, sino que lo que realmente perjudica es el gran número de copias suyas en el sistema, que en algunas situaciones pueden llegar a provocar la parada total de la máquina. Hemos de pensar hay ciertos programas que pueden actuar como conejos sin proponérselo, ejemplos se suelen encontrar en los sistemas Unix destinados a prácticas en las que se enseña a programar al alumnado, es muy común que un bucle que por error se convierte en infinito contenga entre sus instrucciones algunas de reserva de memoria, lo que implica que si el sistema no presenta una correcta política de cuotas para procesos de usuario pueda venirse abajo o degradar enormemente sus prestaciones. El hecho de que el autor suela ser fácilmente localizable no debe ser ninguna excusa para descuidar esta política, no podemos culpar a un usuario por un simple error, y además el daño ya se ha producido.

- Técnicas salami: Por técnica salami se conoce al robo automatizado de pequeñas cantidades de bienes (generalmente dinero) de una gran cantidad origen. El hecho de que la cantidad inicial sea grande y la robada pequeña hace extremadamente difícil su detección, si de una cuenta con varios millones de euros se roban unos céntimos, nadie va a darse cuenta de ello, si esto se automatiza para, por ejemplo, descontar un céntimo de cada nómina pagada en la universidad o de cada beca concedida, tras un mes de actividad seguramente se habría robado una enorme cantidad de dinero sin que nadie se haya percatado de este hecho, ya que de cada origen se ha tomado una cantidad mínima. Las técnicas salami no se suelen utilizar para atacar sistemas normales, sino que su uso más habitual es en sistemas bancarios, sin embargo, como en una red con requerimientos de seguridad medios es posible que haya ordenadores dedicados a contabilidad, facturación de un departamento o gestión de nóminas del personal, se comenta esta potencial amenaza contra el software encargado de estas tareas.

Podemos destacar el parecido entre los ataques a redes y servidores y los ataques o amenazas a la seguridad del mundo real. Los ataques dirigidos a los dispositivos de red que se desea proteger están, finalmente, realizados por personas. Se puede pensar en algunos: robos de bancos, malversaciones y fraudes, invasión de la confidencialidad, sabotaje, espionaje, etc. Todo este tipo de problemas existen en las redes de las organizaciones actuales.

Se puede decir que donde hay dinero hay criminales, con lo que asuntos como el sabotaje remoto al control de cajeros automáticos o la pornografía infantil, pasan a ser perfectamente factibles también en las redes.

Pero, desde el punto de vista más técnico si hay diferencias, se podría decir que el ciberespacio cambia las formas de los ataques y, como consecuencia, se deben adecuar las formas de las defensas. La red Internet tiene varias características que hacen que los citados ataques puedan ser mucho peores, en cierto sentido. Esas características son:

- El aspecto automático de los ataques**.** Si en algo son potentes los ordenadores, es en las tareas repetitivas. Se puede dejar una máquina tratando de descifrar contraseñas

de manera automática, mientras el atacante se va a pasar el fin de semana a esquiar, con la tranquilidad de que su trabajador no se va a cansar, ni se va a quejar.

- El aspecto remoto de los ataques**.** Es casi tan fácil conectarse a un ordenador en Paris desde Madrid que desde Tokio. En ese sentido, se dice que Internet no tiene fronteras. Lo grave es que los aspectos legales de todo esto no están, aún, controlados. Como dijo John Gilmore: **"***The Internet tretas censorship as a damage and routes around it"*(Internet trata a la censura como algo dañino y lo rodea)
- La velocidad de propagación de los ataques**.** Cualquier atacante puede conseguir, con mucha facilidad, y en un tiempo record, muchas de las herramientas de ataque que necesitará. Si es un amateur, le bastará con conseguir las aplicaciones y empezar a usarlas. Si se habla de un profesional, seleccionará entre las mejores aplicaciones, buscará, incluso, aquellas de las que se dispone del código fuente, pero, unos y otros, podrán disfrutar de ellas en un tiempo incomparablemente menor del que necesitarían en el mundo no cibernético.

3.4 Mecanismos

Los mecanismos de seguridad se dividen en tres grandes grupos:

- Prevención
- Detección
- Recuperación

Los mecanismos de prevención son aquellos que aumentan la seguridad de un sistema durante el funcionamiento normal de éste, previniendo la ocurrencia de violaciones a la seguridad, por ejemplo, el uso de cifrado en la transmisión de datos se puede considerar un mecanismo de este tipo, ya que evita que un posible atacante escuche las conexiones.

Por mecanismos de detección se conoce a aquellos que se utilizan para detectar violaciones de la seguridad o intentos de violación, ejemplos de estos mecanismos son los programas de auditoria como Tripwire.

Finalmente, los mecanismos de recuperación son aquellos que se aplican cuando una violación del sistema se ha detectado, para retornar a éste a su funcionamiento correcto, ejemplos de estos mecanismos son la utilización de copias de seguridad o el hardware adicional. Dentro de este último grupo de mecanismos de seguridad encontramos un subgrupo denominado mecanismos de análisis forense, cuyo objetivo no es simplemente retornar al sistema a su modo de trabajo normal, sino averiguar el alcance de la violación, las actividades de un intruso en el sistema, y la puerta utilizada para entrar, de esta forma se previenen ataques posteriores y se detectan ataques a otros sistemas de nuestra red.

Parece claro que, aunque los tres tipos de mecanismos son importantes para la seguridad del sistema, se ha de enfatizar en el uso de mecanismos de prevención y de detección, la máxima popular "más vale prevenir que curar" se puede aplicar a la seguridad, para seguridad Web, evitar un ataque, detectar un intento de violación, o detectar una violación exitosa inmediatamente después de que ocurra es mucho más productivo y menos comprometedor para el sistema que restaurar el estado tras una penetración de la máquina. Es más, si se consigue un sistema sin vulnerabilidades y cuya política de seguridad se implementara mediante mecanismos de prevención de una forma completa, no necesitaríamos mecanismos de detección o recuperación. Aunque esto es

imposible de conseguir en la práctica. Los mecanismos de prevención más habituales son los siguientes:

- Mecanismos de autenticación e identificación
- Mecanismos de control de acceso
- Mecanismos de separación
- Mecanismos de seguridad en las comunicaciones

3.5 Taxonomía de los tipos de ataques.

Una buena clasificación no se puede hacer sin un buen criterio. Desde el punto de vista del origen del ataque, se puede decir que los ataques son [9]:

- **Externos.** Cuando el atacante origina su ataque desde el exterior de una organización concreta, se dice que el ataque es externo. Esto puede ser desde un sitio desconocido de Internet o desde una dirección en la que se confía, pero que ha sido suplantada. Esta característica es esencial para cierto tipo de ataques a una red, que realiza el control de acceso basado en direcciones IP.
- **Internos.** Es obvio que estar dentro de la organización facilita mucho el ataque. Tal ataque ha de ser visto, además, desde un punto de vista amplio. Puede haber ataques *"profesionales"* internos, pero, también, por mala aplicación de la política de seguridad, ataques no maliciosos de usuarios internos que prueban herramientas con toda su buena intención.

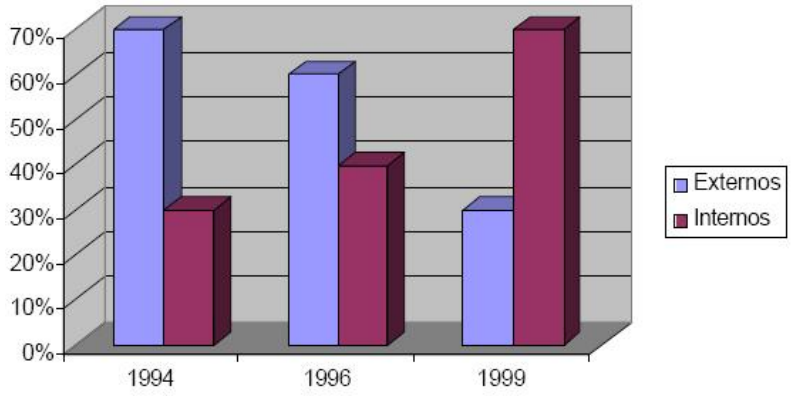

Figura 7: Origen de los ataques.

Otro criterio que conviene utilizar, para una mejor caracterización de cada ataque, es la complejidad. En este sentido se puede hablar de ataques:

- **No estructurados.** Son ataques inocentes, ataques basados en herramientas bastante normales, que pueden tener distintos tipos de objetivos, y ser muy peligrosos, pero, en general, fácilmente reconocibles. Una característica típica de estos ataques es que o bien es una sola herramienta la puesta en marcha o son varias, pero sin ninguna estructuración entre ellas.
- **Estructurados.** Aquí se está ante los más peligrosos. Son ataques que se enfocan como un proyecto. Además tratan de no dejar cabos sueltos, lo que se trata de borrar las huellas y se tienen en cuenta distintos puntos de ataque y de retirada. Aunque sean los más peligrosos, son los menos habituales.

Otro criterio esta relacionado con la personalidad del atacante. Su formación, experiencia y capacidad de comunicación pueden ser fundamentales a la hora de

desarrollar un ataque, especialmente si es uno estructurado y se usa en la fase de obtención de información.

Finalmente si se catalogan por el objetivo que persiguen, una necesidad de clasificación más granular, se puede hablar de los siguientes tipos de ataques:

- Ataques encaminados a la obtención de información sobre los objetos a atacar.
- Ataques orientados a ver, cambiar o borrar información para la que no se está autorizado**,** aprovechándose de la mala administración de las herramientas, máquinas y sistemas de la red.
- Ataques enfocados a ver, cambiar o borrar información para la que no se está autorizado**,** aprovechándose de las vulnerabilidades del software de los sistemas, aplicaciones y protocolos de la red.
- Ataque encauzados a sabotear un servicio, como un servidor Web, un servidor SMTP, el espacio libre en disco de un ordenador, el acceso a (o desde) una red. A todos ellos, se les denomina ataques de denegación de servicio**.**

Si se utiliza este criterio, es muy fácil resaltar los otros criterios citados, con lo cual será este el que usaremos.

3.5.1 Ataques remotos.

3.5.1.1 Escaneo de puertos.

Una de las primeras actividades que un atacante realizará contra su objetivo sería sin duda un escaneo de puertos, un *portscan*, esto le permitirá obtener en primer lugar información básica acerca de qué servicios estamos ofreciendo en nuestras máquinas y, adicionalmente, otros detalles de nuestro entorno como qué sistema operativo tenemos instalados en cada host o ciertas características de la arquitectura de nuestra red. Analizando qué puertos están abiertos en un sistema, el atacante puede buscar agujeros en cada uno de los servicios ofrecidos, cada puerto abierto en una máquina es una puerta de entrada a la misma [10].

Comprobar el estado de un determinado puerto es a priori una tarea muy sencilla, incluso es posible llevarla a cabo desde la línea de órdenes, usando una herramienta tan genérica como telnet. Por ejemplo, para conocer el estado del puerto 5111 en la máquina cuya dirección es www.terra.es, si el telnet ha dicho puerto ofrece una respuesta, entonces está abierto y escuchando peticiones:

```
C:\Documents and Settings\Juanma>telnet www.terra.es 5111
Conectándose a www.terra.es...
No se puede abrir la conexión al host, en puerto 5111:
Error en la conexión
C:\Documents and Settings\Juanma>
```

Si el puerto está protegido por un cortafuegos, lo más probable es que no se obtenga respuesta alguna, el telnet lanzado se quedaría intentando la conexión hasta que se produzca un timeout o hasta que se pare manualmente.

Por lo general, nadie en su sano juicio usará telnet para realizar un escaneo de puertos masivo contra un sistema o contra toda una red, existen herramientas como *strobe* o *nmap* (la más conocida) que pueden realizar esta tarea de una forma más o menos cómoda y automatizable. Evidentemente, ninguno de estos programas se dedica a lanzar

telnet contra los puertos de un sistema, los escaneadores de puertos actuales implementan diferentes técnicas que permiten detectar desde la versión del sistema operativo usado en la máquina atacada, hasta pasar inadvertidos ante diferentes sistemas de detección de intrusos.

Existen diferentes aproximaciones para clasificar los escaneos de puertos, tanto en función de las técnicas seguidas en el ataque como en función de a qué sistemas o puertos concretos va dirigido. Se habla de un escaneo horizontal cuando el atacante busca la disponibilidad de determinado servicio en diferentes máquinas de una red, por ejemplo, si el pirata dispone de un exploit que aprovecha un fallo en la implementación de sendmail, es normal que trate de averiguar qué máquinas aceptan peticiones smtp en un determinado segmento para posteriormente atacar a dichos sistemas. Si por contra el pirata sólo escanea puertos de una máquina, se denomina al ataque escaneo vertical, y suele denotar el interés del atacante en ese host concreto. Si comprueba todos los puertos del sistema al escaneo se le denomina vanilla, mientras que si sólo lo hace contra determinados puertos o rangos, se le denomina strobe. Si nos basamos en las técnicas utilizadas, podemos dividir los escaneos en tres grandes familias:

- Open.
- Half-open
- Stealth

Los escaneos open se basan en el establecimiento de una conexión TCP completa mediante el conocido como protocolo de acuerdo de tres vías o *three-way handshake*, por lo que son muy sencillos de detectar y detener. Utilizan la llamada connect().

La segunda técnica half-open, el pirata finaliza la conexión antes de que se complete el protocolo de acuerdo de tres vías, lo que dificulta la detección del ataque por parte de algunos detectores de intrusos muy simples (casi todos los actuales son capaces de detectarlos). Dentro de esta técnica se encuentra el *SYN Scanning*: cuando el origen - atacante - recibe del destino - máquina escaneada - los *bits* SYN+ACK, envía un *bit* RST (no es necesaria una nueva trama, ya que este *bit* se envía automáticamente a nivel de núcleo) en lugar del ACK correspondiente a un *three-way handshake* completo. Los escaneos SYN son fácilmente detectables y pueden ser bloqueados en cualquier cortafuegos. Existe una variable de esta técnica denominada *dumb scanning* en la que entra en juego una tercera máquina denominada “tonta” (por el poco tráfico que emite y recibe), algo que puede ayudar al pirata a camuflar su origen real. Sin embargo, el *dumb scanning* es más complicado que el *SYN scanning*, por lo que se utiliza mucho menos en la vida real.

En diciembre de 1995 Christopher Klaus proporcionó las pautas de ciertas técnicas de escaneo que permitían al atacante eludir la acción de los sistemas de detección de intrusos de la época y a las que bautizó como *stealth scanning*, actualmente el significado del término ha cambiado, ya que lo que Klaus presentó se denomina hoy en día *half-open scanning*, y por *stealth scanning* se conoce a una familia de técnicas de escaneo que cumplen alguna de las siguientes condiciones:

- Eludir cortafuegos o listas de control de acceso.
- No ser registradas por sistemas de detección de intrusos, ni orientados a red ni en el propio *host* escaneado.
- Simular tráfico normal y real para no levantar sospechas ante un analizador de red.

Una de las técnicas que se encuentran dentro de la familia de los escaneos *stealth* es la conocida como SYN+ACK. La idea es muy simple, y consiste en una violación del *three-way handshake*, el atacante, en lugar de enviar en primer lugar una trama SYN, envía SYN+ACK. Si el puerto está abierto simplemente se ignora, y si está cerrado sabe que no ha recibido previamente un paquete SYN, por lo que lo considera un error y envía una trama RST para finalizar la conexión.

Los escaneos basados en este método se usan poco en la actualidad, debido al elevado número de falsos positivos que pueden generar: sólo debemos pensar en los múltiples motivos que pueden existir para que un sistema no responda ante una petición SYN+ACK, desde listas de control de accesos en los *routers* o cortafuegos hasta simples *timeouts*.

Otra técnica dentro de los escaneos *stealth* es el FIN *scanning*, en este caso, el atacante envía a su objetivo una trama con el *bit* FIN activo, ante lo que este responde con un RST si el puerto está cerrado, o simplemente no responde en caso de estar abierto, como en el caso de los escaneos SYN+ACK este método puede proporcionar muchos falsos positivos, por lo que tampoco se utiliza mucho hoy en día.

Un método de escaneo algo más complejo es el ACK. El atacante envía una trama con este *bit* activo, y si el puerto destino está abierto es muy posible que o bien el campo TTL del paquete de vuelta sea menor que el del resto de las tramas RST recibidas, o que el tamaño de ventana sea mayor que cero, como podemos ver, en este caso no basta con analizar el bit RST sino también la cabecera IP del paquete respuesta. Este método es difícil de registrar por parte de los detectores de intrusos, pero se basa en el código de red de BSD, por lo que es dependiente del operativo escaneado.

Para finalizar con la familia de *stealth scanning* decir que hay dos métodos opuestos entre sí pero que se basan en una misma idea y proporcionan resultados similares, se trata de XMAS y NULL. Los primeros, también denominados escaneos “árbol de navidad”, se basan en enviar al objetivo tramas con todos los *bits* TCP (URG, ACK, PST, RST, SYN y FIN) activos, si el puerto está abierto el núcleo del sistema operativo eliminará la trama, ya que evidentemente la considera una violación del *three-way handshake*, pero si está cerrado devolverá un RST al atacante. Como antes, este método puede generar demasiados falsos positivos, y además sólo es aplicable contra máquinas Unix debido a que está basado en el código de red de BSD. El método opuesto al XMAS se denomina NULL *scanning*, y evidentemente consiste en enviar tramas con todos los *bits* TCP reseteados, el resultado es similar, no se devolverá ningún resultado si el puerto está abierto, y se enviará un RST si está cerrado. Aunque en principio este método sería aplicable a cualquier pila TCP/IP, la implementación incorrecta que hacen algunos sistemas operativos (entre ellos HP/UX o IRIX) de esta, hace que en ocasiones se envíen *bits* RST también desde los puertos abiertos, lo que puede proporcionar demasiados falsos positivos.

Otra técnica que no esta englobada en la anterior clasificación es los escaneos UDP, que al contrario de todos los comentados hasta el momento utiliza este protocolo, y no TCP, para determinar el estado de los puertos de una máquina. Al enviar un datagrama UDP a un puerto abierto este no ofrece respuesta alguna, pero si está cerrado devuelve un mensaje de error ICMP: ICMP_PORT_UNREACHABLE. Evidentemente, estos ataques son muy sencillos de detectar y evitar tanto en un sistema de detección de intrusos como en los núcleos de algunos Unix, y por si esto fuera poco debemos recordar que UDP no es un protocolo orientado a conexión (como lo es TCP), por lo que la pérdida de datagramas puede dar lugar a un elevado número de falsos positivos.

3.5.1.2 Spoofing.

Por *spoofing* se conoce a la creación de tramas TCP/IP utilizando una dirección IP falsa, desde su equipo, un pirata simula la identidad de otra máquina de la red para conseguir acceso a recursos de un tercer sistema que ha establecido algún tipo de confianza basada en el nombre o la dirección IP del *host* suplantado. Y como los anillos de confianza basados en estas características tan fácilmente falsificables son aún demasiado abundantes, el *spoofing* sigue siendo en la actualidad un ataque no trivial, pero factible contra cualquier tipo de organización. En el *spoofing* entran en juego tres máquinas: un atacante, un atacado, y un sistema suplantado que tiene cierta relación con el atacado, para que el pirata pueda conseguir su objetivo necesita por un lado establecer una comunicación falseada con su objetivo, y por otro evitar que el equipo suplantado interfiera en el ataque. Probablemente esto último no le sea muy difícil de conseguir, a pesar de que existen múltiples formas de dejar fuera de juego al sistema suplantado - al menos a los ojos del atacado - que no son triviales (modificar rutas de red, ubicar un filtrado de paquetes entre ambos sistemas...), lo más fácil en la mayoría de ocasiones es simplemente lanzar una negación de servicio contra el sistema en cuestión. Aunque en el punto siguiente hablaremos con más detalle de estos ataques, no suele ser difícil “tumbar”, o al menos bloquear parcialmente, un sistema medio; si a pesar de todo el atacante no lo consigue, simplemente puede esperar a que desconecten de la red a la máquina a la que desea suplantar (por ejemplo, por cuestiones de puro mantenimiento).

El otro punto importante del ataque, la comunicación falseada entre dos equipos, no es tan inmediato como el anterior y es donde reside la principal dificultad del *spoofing*. En un escenario típico del ataque, un pirata envía una trama SYN a su objetivo indicando como dirección origen la de esa tercera máquina que está fuera de servicio y que mantiene algún tipo de relación de confianza con la atacada. El *host* objetivo responde con un SYN+ACK a la tercera máquina, que simplemente lo ignorará por estar fuera de servicio (si no lo hiciera, la conexión se resetearía y el ataque no sería posible), y el atacante enviará ahora una trama ACK a su objetivo, también con la dirección origen de la tercera máquina. Para que la conexión llegue a establecerse, esta última trama deberá enviarse con el número de secuencia adecuado, el pirata ha de predecir correctamente este número, si no lo hace, la trama será descartada, y si lo consigue la conexión se establecerá y podrá comenzar a enviar datos a su objetivo, generalmente para tratar de insertar una puerta trasera que permita una conexión normal entre las dos máquinas.

Podemos comprobar que el *spoofing* no es inmediato, de entrada, el atacante ha de hacerse una idea de cómo son generados e incrementados los números de secuencia TCP, y una vez que lo sepa ha de conseguir engañar a su objetivo utilizando estos números para establecer la comunicación, cuanto más robusta sea esta generación por parte del objetivo, más difícil lo tendrá el pirata para realizar el ataque con éxito. Además el *spoofing* es un ataque ciego, el atacante no ve en ningún momento las respuestas que emite su objetivo, ya que estas van dirigidas a la máquina que previamente ha sido deshabilitada, por lo que debe presuponer qué está sucediendo en cada momento y responder de forma adecuada en base a esas suposiciones.

Para evitar ataques de *spoofing* exitosos contra nuestros sistemas podemos tomar diferentes medidas preventivas, en primer lugar, parece evidente que una gran ayuda es reforzar la secuencia de predicción de números de secuencia TCP. Otra medida sencilla es eliminar las relaciones de confianza basadas en la dirección IP o el nombre de las máquinas, sustituyéndolas por relaciones basadas en claves criptográficas, el cifrado y el

filtrado de las conexiones que pueden aceptar nuestras máquinas también son unas medidas de seguridad importantes de cara a evitar el *spoofing*.

Hay otros ataques de falseamiento aparte del comentado entre los que cabe destacar:

- *DNS Spoofing:* Este ataque hace referencia al falseamiento de una dirección IP ante una consulta de resolución de nombre (esto es, resolver con una dirección falsa un cierto nombre DNS), o viceversa (resolver con un nombre falso una cierta dirección IP). Esto se puede conseguir de diferentes formas, desde modificando las entradas del servidor encargado de resolver una cierta petición para falsear las relaciones dirección-nombre, hasta comprometiendo un servidor que infecte la caché de otro (lo que se conoce como *DNS Poisoning*); incluso sin acceso a un servidor DNS real, un atacante puede enviar datos falseados como respuesta a una petición de su víctima sin más que averiguar los números de secuencia correctos.
- *ARP Spoofing:* El ataque denominado *ARP Spoofing* hace referencia a la construcción de tramas de solicitud y respuesta ARP falseadas, de forma que en una red local se puede forzar a una determinada máquina a que envíe los paquetes a un *host* atacante en lugar de hacerlo a su destino legítimo. La idea es sencilla, y los efectos del ataque pueden ser muy negativos, desde negaciones de servicio hasta interceptación de datos.
- *Web Spoofing:* Este ataque permite a un pirata visualizar y modificar cualquier página *Web* que su víctima solicite a través de un navegador, incluyendo las conexiones seguras vía SSL. Para ello, mediante código malicioso un atacante crea una ventana del navegador correspondiente, de apariencia inofensiva, en la máquina de su víctima, a partir de ahí, enruta todas las páginas dirigidas al equipo atacado, incluyendo las cargadas en nuevas ventanas del navegador, a través de su propia máquina, donde son modificadas para que cualquier evento generado por el cliente sea registrado (esto implica registrar cualquier dato introducido en un formulario, cualquier *clic* en un enlace, etc.).

3.5.1.3 Negaciones de servicio.

El propósito general del ataque es denegar a las víctimas el acceso a un recurso particular, esta caracterizado por un intento, por parte de los agresores, de evitar que los legítimos usuarios puedan utilizar un determinado servicio. Por ejemplo:

- Intento de "inundar" una red, impidiendo el trafico legítimo de esa red.
- Intento de romper la conexión entre dos máquinas, impidiendo el acceso a un servicio determinado.
- Intento de impedir el acceso de un usuario individual a un servicio.
- Intento de romper los servicios a un sistema especifico.

Las negaciones de servicio (Denial of Service, DoS) son ataques dirigidos contra un recurso informático con el objetivo de degradar total o parcialmente los servicios prestados por ese recurso a sus usuarios legítimos, constituyen en muchos casos uno de los ataques más sencillos y contundentes contra todo tipo de servicios, y en entornos donde la disponibilidad es valorada por encima de otros parámetros de la seguridad global puede convertirse en un serio problema, ya que un pirata puede interrumpir constantemente un servicio sin necesidad de grandes conocimientos o recursos, utilizando simplemente sencillos programas y un módem y un PC caseros.

Lo ataques "Denegación de Servicio" vienen dados en una variedad de formas y se dirigen hacia una gran variedad de servicios. Hay tres tipos básicos de ataque:

- Consumo de recursos, limitados o no renovables.
- Destrucción o alteración de información sobre configuración.
- Destrucción física o alteración de componentes de una red.

Todos los sistemas Unix ofrecen mecanismos para evitar que un usuario normal pueda tirar abajo una máquina. El problema real no es que usuarios legítimos de un entorno causen, intencionada o inintencionadamente, negaciones de servicio, el mayor problema es que esas negaciones sean causadas de forma remota por piratas ajenos por completo a nuestra organización, capaces de tumbar un servidor de millones de pesetas con sencillos programas, sin dejar ningún rastro y lo peor, sin ser muchas veces conscientes del daño que están haciendo.

Estos ataques remotos de negación de servicio son considerados negativos incluso por muchos de los propios piratas, ya que realmente no suelen demostrar nada positivo de quien los lanza. En la mayor parte de los casos la negación de servicio tiene éxito porque el objetivo utiliza versiones no actualizadas de demonios o del propio núcleo del sistema operativo, si se detiene o degrada por completo la máquina. Para evitar esto, la norma a seguir es evidente, mantener siempre actualizados nuestros sistemas, tanto en lo referente al nivel de parcheado o versiones del núcleo, como en lo referente a programas críticos encargados de ofrecer un determinado servicio.

Las negaciones de servicio más habituales suelen consistir en la inhabilitación total de un determinado servicio o de un sistema completo, bien porque ha sido realmente bloqueado por el atacante o bien porque está tan degradado que es incapaz de ofrecer un servicio a sus usuarios. En la mayor parte de sistemas, un usuario con acceso shell no tendrá muchas dificultades en causar una negación de servicio que tirara abajo la máquina o la ralentizara enormemente, esto no tiene porqué ser un ataque intencionado, sino que puede deberse a un simple error de programación.

El consumo de recursos están limitados, ordenadores y redes necesitan ciertas cosas para funcionar: ancho de banda para la red, memoria y espacio de disco, tiempo de CPU, estructuras de datos, acceso a otros equipos y redes, y unos ciertos recursos de entorno como son: potencia o aire frío.

1. Conectividad entre redes: Los ataques denegación de servicio se llevan a cabo frecuentemente contra la capacidad de conectitividad entre redes. La intención es evitar que los equipos o las subredes se puedan comunicar con el resto de la red. Un ejemplo de este tipo de ataque es el *SYN flood*[1]. En este tipo de ataque, se comienza por el proceso de establecimiento de conexión de la máquina víctima, la culminación del ataque llega cuando la máquina víctima ha reservado uno de los limitados números de estructuras de datos requeridos para completar la conexión. El resultado es que la legítima conexión se deniega mientras que la máquina víctima espera a que se complete la conexión, cosa que no llega a suceder. Hay que notar que este tipo de ataque no se pretende consumir ancho de banda sino los recursos de memoria y procesador que se consumen con las estructuras de datos usadas para establecer una comunicación. Esto implica que un atacante puede llevar a cabo su ataque desde un módem contra una red mucho más rápida.
2. Utilización de tus propios recursos contra ti: Un intruso puede usar tus propios recursos contra ti de inesperadas maneras.

3. Consumo de ancho de banda: Un intruso también puede ser capaz de consumir todo el ancho de banda disponible en la red generando una gran cantidad de paquetes dirigidos a tu red. Típicamente esos son paquetes ICMP ECHO, pero en principio pueden ser de cualquier otro tipo. El intruso no tiene porque actuar desde una de las máquinas de la subred, puede ser capaz de coordinar varias máquinas de diferentes redes para obtener el mismo resultado.
4. Consumo de otros recursos: Junto con el consumo de ancho de banda, los intrusos pueden ser capaces de consumir otros recursos que el sistema necesita para operar correctamente. Por ejemplo, hay sistemas en los que existen un número limitado de estructuras de datos para proporcionar información sobre procesos (identificadores de procesos, tablas de entradas de procesos, etc.). Un intruso puede ser capaz de consumir esas estructuras de datos escribiendo un script que no haga nada, pero que repetidamente cree copias de sí mismo. Muchos sistemas operativos modernos, aunque no todos, tienen cuotas para proteger contra estos ataques. De todas formas, si la tabla de procesos no está llena, la CPU puede estar ocupada con un gran número de procesos y el correspondiente tiempo para conmutar entre procesos. Un intruso también puede intentar consumir espacio de disco de otras maneras, por ejemplo:
    - Generar excesivo número de mensajes de correo.
    - Generar intencionadamente errores que se almacenen en ficheros log.
    - Colocar ficheros en áreas de ftps anónimos.

En general, cualquier cosa que permita datos en disco, puede ser utilizada para llevar a cabo un ataque denegación de servicio. Por otro lado en algunos lugares, aparece el mensaje "lockout" después de un cierto número de intentos fallidos de acceso a una determinada cuenta, normalmente 3 o 5. Un intruso puede usar este hecho para evitar el acceso a la cuenta a su legítimo usuario. En algunos casos las cuentas con privilegios suelen ser blancos de estos ataques.

Lo mejor es estar seguro de tener un método alternativo para acceder al sistema en caso de emergencia. Un intruso puede hacer que un sistema se venga abajo o que se vuelva inestable enviando datos inesperados a través de la red. Si tu sistema se viene abajo a menudo sin causa aparente, puede que se trate de este tipo de ataque. Hay otras cosas que pueden ser vulnerables para un ataque denegación de servicio y que es conveniente monitorizar:

- impresoras
- cintas de backup
- conexiones a redes
- cualquier recurso importante para las operaciones de tu empresa

Los ataques por denegación de servicio pueden suponer una importante pérdida de tiempo y dinero para la mayoría de las organizaciones. Recomendamos considerar las siguientes opciones dependiendo de las necesidades personales:

- Implementar routers que filtren. Esto disminuirá la exposición a ciertos ataques. Adicionalmente, esto puede prevenir que usuarios de tu red lancen ciertos ataques.

- Si es posible, instalar parches para protegerse contra ataques TCP SYN flooding. Esto puede reducir substancialmente la exposición a estos ataques, aunque no elimina el riesgo por completo.
- Desactivar cualquier servicio de red que no sea necesario. Esto puede limitar la capacidad del intruso de utilizar estos servicios para realizar su ataque.
- Activar mecanismos de cuota en el sistema operativo si es posible. Por ejemplo si el sistema operativo soporta cuotas de disco, activarlas para todas las cuentas, especialmente para cuentas que operan con servicios de red. Además si el sistema operativo soporta particionado de discos, es conveniente considerar particionar sistemas de ficheros para separar funciones críticas de otras actividades.
- Observar el rendimiento y estabilidad de las líneas básicas de la actividad cotidiana. Observar niveles inusuales de actividad de disco, uso de CPU o tráfico de red.
- Examinar rutinariamente la seguridad física. Considerar servidores, routers, terminales, puntos de acceso a la red y otros componentes del sistema.
- Utilizar Tripwire o una herramienta similar para detectar cambios en la información sobre configuración o de otros ficheros.
- Colocar y mantener "hot spares", máquinas que pueden ser puestas en servicio rápidamente en el momento que una máquina se ha desactivado.
- Activar en la red configuraciones redundantes y tolerantes a fallos.
- Establecer y mantener calendarios y políticas de backup, sobre todo para información importante sobre configuración.
- Establecer y mantener políticas de claves apropiadas, especialmente para acceso a cuentas con privilegios como es la de *root*. Muchas organizaciones pueden sufrir perdidas financieras como resultado de estos ataques y pueden decidir presentar cargos contra el intruso.

De un tiempo a esta parte se ha popularizado mucho el término negación de servicio distribuida (Distributed Denial of Service, DDoS), en este ataque un pirata compromete en primer lugar un determinado número de máquinas y, en un determinado momento, hace que todas ellas ataquen masiva y simultáneamente al objetivo u objetivos reales enviándoles diferentes tipos de paquetes, por muy grandes que sean los recursos de la víctima, el gran número de tramas que reciben hará que tarde o temprano dichos recursos sean incapaces de ofrecer un servicio, con lo que el ataque habrá sido exitoso.

Si en lugar de cientos o miles de equipos atacando a la vez lo hiciera uno sólo las posibilidades de éxito serían casi inexistentes, pero es justamente el elevado número de pequeños atacantes lo que hace muy difícil evitar este tipo de negaciones de servicio. Según el CERT los ataques de negación de servicio distribuidos más habituales consisten en el envío de un gran número de paquetes a un determinado objetivo por parte de múltiples hots, lo que se conoce como packet ooding. Defenderse de este tipo de ataques es difícil, en primer lugar, uno piensa en bloquear de alguna forma (probablemente en un cortafuegos o en un router) todo el tráfico proveniente de los atacantes, pero ¿qué sucede cuando tenemos miles de ordenadores atacando desde un gran número de redes diferentes? ¿Los bloqueamos uno a uno? Esto supondría un gran esfuerzo que difícilmente ayudará, ya que lo más probable es que en el tiempo que nos cueste bloquear el tráfico de una

determinada máquina, dos o tres nuevas nos comiencen a atacar. Entonces, ¿bloqueamos todo el tráfico dirigido hacia el objetivo? Si hacemos esto, estamos justamente ayudando al atacante, ya que somos nosotros mismos los que causamos una negación en el servicio a los usuarios legítimos de nuestro sistema. La defensa ante una negación de servicio distribuida no es inmediata, en cualquier caso, podemos tomar ciertas medidas preventivas que nos ayudarán a limitar el alcance de uno de estos ataques. Un correcto filtrado del tráfico dirigido a nuestras máquinas es vital para garantizar nuestra seguridad, no hay que responder a pings externos a nuestra red, es necesario activar el antispoofing en nuestros cortafuegos y en los elementos de electrónica de red que lo permitan, etc. Establecer correctamente límites a la utilización de nuestros recursos es también una importante medida preventiva, es posible limitar el ancho de banda dedicado a una determinada aplicación o a un protocolo, de forma que las utilizaciones por encima del margen son negadas. También podemos limitar los recursos del sistema (CPU, memoria, disco…) que puede consumir en global una determinada aplicación servidora (por ejemplo, un demonio sirviendo páginas Web), además de restringir sus recursos por cliente simultáneo. A pesar de las dificultades con las que nos podemos encontrar a la hora de prevenir ataques de negación de servicio, una serie de medidas sencillas pueden ayudarnos de forma relativa en esa tarea, las negaciones de servicio son por desgracia cada día más frecuentes, y ninguna organización está a salvo de las mismas. Especialmente en los ataques distribuidos, la seguridad de cualquier usuario conectado a Internet es un eslabón importante en la seguridad global de la red, ya que esos usuarios se convierten muchas veces sin saberlo en satélites que colaboran en un ataque masivo contra organizaciones de cualquier tipo. Cuanto más difícil se lo pongamos cada uno de nosotros a los piratas, mucho mejor para todos.

3.5.1.4 Interceptación

La interceptación lógica de datos más conocida y extendida es el sniffing, consiste en capturar tramas que circulan por la red mediante un programa ejecutándose en una máquina conectada a ella o bien mediante un dispositivo que se engancha directamente el cableado.

En las redes de difusión, cuando una máquina envía una trama a otra indica en un campo reservado la dirección del host destino, todas las máquinas del dominio de colisión ven esa trama, pero sólo su receptora legítima la captura y elimina de la red. Este es el funcionamiento normal de TCP/IP, sin embargo, es necesario insistir en un aspecto, todas las máquinas ven la trama, y si no leen todos sus campos es porque no quieren. Existe un modo de funcionamiento de las interfaces de red denominado modo promiscuo, en el cual la tarjeta lee todas las tramas que circulan por la red, tanto dirigidas a ella como a otras máquinas, el leerlas no implica el eliminarlas de la red, por lo que el host destino legítimo la recibirá y eliminará sin notar nada extraño.

Para hacer funcionar un interfaz de red en modo promiscuo es necesario tener un control total del sistema, es decir, ser root en la máquina, esto ayuda un poco en la defensa, pero no soluciona el problema, no podemos permitir que cualquiera que sea superusuario de un sistema pueda capturar todo el tráfico que pasa por el mismo (incluyendo claves, correo electrónico, y cientos de datos privados). Por si esto fuera poco, en los sistemas donde todos los usuarios tienen un control total de la máquina (por ejemplo, en toda la familia Windows 9x) ni siquiera hace falta ese privilegio, cualquiera que se siente en un PC puede ejecutar un sniffer y capturar todo el tráfico de la red.

Hay programas para que actúan como sniffer entre ellos cabe destacar, *dsniff* y su familia, capaces hasta de capturar los correos electrónicos directamente en formato smtp o

cargar de forma automática en un navegador las mismas páginas que visita la víctima del ataque, otro programa es el *snoop* de Solaris, que vuelca paquetes en un formato por defecto casi ilegible, pasando por los clásicos *tcpdump* o *sniffit*. Para evitar que programas de este tipo capturen nuestra información existen diferentes aproximaciones más o menos efectivas, como sustituir los HUBs de nuestra red por switches que aíslan dominios de colisión (esto dificulta el ataque pero no lo imposibilita) o implantar redes privadas virtuales. Pero la más barata y sencilla es el uso de protocolos cifrados siempre que nos sea posible (que lo suele ser casi siempre), sustituir telnet y rlogin por ssh y ftp por scp o sftp es muy sencillo, y nos proporciona un incremento de seguridad abismal en nuestro entorno. Implantar SSL o túneles seguros quizás es algo más costoso, pero también en la mayoría de ocasiones es algo que vale la pena hacer, en todo momento hemos de tener presente que el sniffing es un peligro real, que no necesita de grandes medios y, lo que es peor, indetectable en la mayor parte de casos, a pesar de que existen métodos para tratar de detectar sistemas con un interfaz en modo promiscuo, no suelen ser todo lo efectivos que uno podría esperar, ya que detectar una máquina en este estado no es ni de lejos inmediato.

El sniffing es el ataque de interceptación más conocido y utilizado, pero no es el único que se puede poner en práctica contra un sistema determinado, Unix o no. En algunas versiones de Linux existe un programa denominado ttysnoop capaz de registrar en tiempo real todo lo que un usuario teclea en una terminal, tanto física como virtual. Aunque el resultado es en muchos aspectos similar al sniffing, técnicamente poco tiene que ver con este, en ningún momento se capturan datos que circulan por la red, la tarjeta no trabaja en modo promiscuo (es mas, ni siquiera es necesario un interfaz de red), etc, simplemente, la información que un usuario introduce en una terminal es clonada en otra, permitiendo tanto la entrada como la salida de datos a través de ambas. Aunque Linux sea el sistema Unix nativo de ttysnoop existen versiones también para otros entornos.

Otro ataque de interceptación, menos utilizado que los anteriores pero igual de peligroso, es el keylogging, el registro de las teclas pulsadas por un usuario en una sesión, otro ejemplo de un programa que capture esta información puede ser una mula de troya clásica.

3.5.2 Ataques orientados a la obtención de información sobre el objetivo.

Para obtener información sobre el objetivo a atacar hay 2 metodologías: la ingeniería social y las técnicas informáticas de obtención de información.

Para la ingeniería social describimos dos casos reales:

1. En 1994, un hacker francés, llamado Anthony Zboralski, llama a la oficina del FBI en Washington, haciéndose pasar por un representante del FBI, trabajando en la oficina de EE.UU. en París. Persuade a la persona al otro lado del teléfono de que le explique cómo conectarse al sistema de conferencias telefónicas del FBI. Después se gasta 250000 dólares en 7 meses.
2. En 1999, todos los clientes de AOL, en EE.UU, recibieron varias veces mensajes como éste:

> "SYSTEM ADMINISTRATOR: A database error has deleted the information for over 25000 accounts, and tours is one. In order for us to access the backup data, we do need your password. Without your password, we will NOT be able to

> allow you to sign onto America Online within the next 24 hours after your opening of this letter".

Es una táctica común llamar por teléfono a empleados de una compañía, haciéndose pasar por un administrador de red, oficial de seguridad o personal de mantenimiento. Si el atacante sabe suficiente de la red de la compañía como para sonar convincentemente, puede obtener contraseñas, nombres de cuentas, etc. Muchas veces, todo consiste en preguntar. La gente quiere colaborar y suele fiarse de otra gente que sabe de qué habla, más si le da datos de su lugar de trabajo.

En realidad, el término ingeniería social es equivalente a engañar, mentir, consiguiendo que otra persona haga cosas que el atacante quiere que se hagan. Es difícil de parar, al menos técnicamente, pues no usa métodos informáticos. Va al eslabón más débil de la cadena de seguridad: usa del factor humano.

A la hora de tenerlo en cuenta en la política de seguridad, habrá que hacerlo en la parte de formación general para todo el mundo. Aunque hay alguna cosa que se puede hacer para frenar este tipo de ataques, nunca se conseguirá pararlos del todo con herramientas informáticas, pues se aprovechan de la buena voluntad de la gente.

En cuanto a la otra opción, las técnicas informáticas de obtención de información, se habla, generalmente, de cómo usar comandos y herramientas, no diseñadas específicamente, para un ataque, como medios de obtención de información.

3.5.3 Ataques vía Web.

Durante los últimos años los servidores Web [11] se han convertido en una excelente fuente de diversión para piratas, cualquier empresa que se precie, desde las más pequeñas a las grandes multinacionales, tiene una página Web en las que al menos trata de vender su imagen corporativa. Si hace unos años un pirata que quisiera atacar a una empresa (y no a todas, ya que muy pocas tenían representación en la red) tenía que agenciárselas para obtener primero información de la misma y después buscar errores de configuración más o menos comunes de sus sistemas, hoy en día le basta con teclear el nombre de su objetivo en un navegador y añadir la coletilla *.com* detrás del mismo para contactar con al menos una de sus máquinas, su servidor Web.

Los ataques a las páginas Web de una organización son casi siempre los más vistosos que la misma puede sufrir, en cuestión de minutos piratas de todo el mundo se enteran de cualquier problema en la página Web principal de una empresa más o menos grande pueda estar sufriendo, y si se trata de una modificación de la misma incluso existen recopilatorios de páginas hackeadas. Por supuesto, la noticia de la modificación salta inmediatamente a los medios, que gracias a ella pueden rellenar alguna cabecera sensacionalista sobre *los piratas de la red*, y así se consigue que la imagen de la empresa atacada caiga notablemente y la del grupo de piratas suba entre la comunidad *underground* nacional o internacional.

La mayor parte de estos ataques tiene éxito gracias a una configuración incorrecta del servidor o a errores de diseño del mismo, si se trata de grandes empresas, los servidores Web suelen ser bastante complejos (alta disponibilidad, balanceo de carga, sistemas propietarios de actualización de contenidos…) y difíciles de administrar correctamente, mientras que si la empresa es pequeña es muy posible que haya elegido un servidor Web simple en su instalación y administración pero en el cual es casi imposible garantizar una mínima seguridad. Sea por el motivo que sea, la cuestión es que cada día es más sencillo para un pirata ejecutar órdenes de forma remota en una máquina, o al menos modificar

contenidos de forma no autorizada, gracias a los servidores Web que un sistema pueda albergar.

Cualquier analizador de vulnerabilidades que podamos ejecutar contra nuestros sistemas (nessus, ISS Security Scanner, NAI CyberCop Scanner…) es capaz de revelar información que nos va a resultar útil a la hora de reforzar la seguridad de nuestros servidores Web, incluso existen analizadores que están diseñados para auditar únicamente este servicio, como whisker, el cual nos proporciona excesiva información sobre la configuración del servidor(versión, módulos, soporte SSL…), y que la herramienta ha obtenido algunos archivos y directorios que pueden resultar interesantes para un atacante, en el caso de los CGI no tiene más que acercarse a alguna base de datos de vulnerabilidades e introducir en el buscador correspondiente el nombre del archivo para obtener información sobre los posibles problemas de seguridad que pueda presentar. El caso de los directorios es algo diferente, pero típicamente se suele tratar de nombres habituales en los servidores que contienen información que también puede resultarle útil a un potencial atacante.

¿Cómo evitar estos problemas de seguridad de los que estamos hablando? Una medida elemental es eliminar del servidor cualquier directorio o CGI de ejemplo que se instale por defecto, aunque generalmente los directorios (documentación, ejemplos…) no son especialmente críticos, el caso de los CGIs es bastante alarmante, muchos servidores incorporan programas que no son ni siquiera necesarios para el correcto funcionamiento del software, y que en ciertos casos abren enormes agujeros de seguridad, como el acceso al código fuente de algunos archivos, la lectura de ficheros fuera del DocumentRoot, o incluso la ejecución remota de comandos bajo la identidad del usuario con que se ejecuta el demonio servidor.

Otra medida de seguridad básica es deshabilitar el Directory Indexing que por defecto muchos servidores incorporan, la capacidad de obtener el listado de un directorio cuando no existe un fichero index.html o similar en el mismo, se trata de una medida extremadamente útil y sobre todo sencilla de implantar, ya que en muchos casos un simple *chmod -r* sobre el directorio en cuestión es suficiente para evitar este problema. A primera vista esta medida de protección nos puede resultar curiosa, a fin de cuentas, a priori todo lo que haya bajo el Document Root del servidor ha de ser público, ya que para eso se ubica ahí. Evidentemente la teoría es una cosa y la práctica otra muy diferente, entre los ficheros de cualquier servidor no es extraño encontrar desde archivos de *log* hasta paquetes *tar* con el contenido de subdirectorios completos. Por supuesto, la mejor defensa contra estos ataques es evitar de alguna forma la presencia de estos archivos bajo el Document Root, pero en cualquier caso esto no es siempre posible, y si un atacante sabe de su existencia puede descargarlos, obteniendo en muchos casos información realmente útil para atacar al servidor (como el código de ficheros jsp, php, asp… o simplemente rutas absolutas en la máquina), y una excelente forma de saber que uno de estos ficheros está ahí es justamente el Directory Indexing, por si esto no queda del todo claro, no tenemos más que ir a un buscador cualquiera y buscar la cadena *Index of /admin*, por poner un ejemplo sencillo, para hacernos una idea de la peligrosidad de este error de configuración.

Además, en cualquier servidor Web es muy importante el usuario bajo cuya identidad se ejecuta el demonio *httpd*, ese usuario no debe ser nunca el root del sistema, pero tampoco un usuario genérico como *nobody*, se ha de tratar siempre de un usuario dedicado y sin acceso real al sistema. Por supuesto, las páginas html (los ficheros planos, para entendernos) nunca deben ser de su propiedad, y mucho menos ese usuario ha de tener permiso de escritura sobre los mismos, con un acceso de lectura (y ejecución, en caso de

CGIs) es más que suficiente en la mayoría de los casos. Hemos de tener en cuenta que si el usuario que ejecuta el servidor puede escribir en las páginas Web, y un pirata consigue ejecutar órdenes bajo la identidad de dicho usuario, podría modificar las páginas Web sin ningún problema (que no olvidemos, es lo que perseguirá la mayoría de atacantes de nuestro servidor Web).

Igual de importante que evitar estos problemas es detectar cuando alguien trata de aprovecharlos intentando romper la seguridad de nuestros servidores, para conseguirlo no tenemos más que aplicar las técnicas de detección de intrusos que veremos más adelante. Una característica importante de los patrones de detección de ataques vía Web es que no suelen generar muchos falsos positivos, por lo que la configuración de la base de datos inicial es rápida y sencilla, al menos en comparación con la detección de escaneos de puertos o la de tramas con alguna característica especial en su cabecera.

3.5.4 Ataques a los sistemas.

El número de ataques de hackers a sistemas conectados a Internet está teniendo un crecimiento exponencial en los últimos tiempos. Asimismo, el número de servicios que la Administración Pública proporciona a los ciudadanos a través de Internet es cada vez mayor, siendo en consecuencia también mayor el riesgo de ataques a los Sistemas de Información de las entidades públicas.

3.5.4.1 Troyanos.

Un troyano es un programa malicioso insertado en un PC sin consentimiento de su dueño que permite el control de ese PC por parte de una persona no autorizada, pudiéndose incluso considerar un tipo de virus, ya que el PC atacado se “infecta” con él. Tiene unas características propias que le confieren un carácter malicioso:

- Se aprovecha frecuentemente de bugs y backdoors de los sistemas informáticos, como el Back Orifice o el BackDoor.
- Sólo una de las dos partes del programa (un troyano se instala en ambos PCs) tiene capacidad de controlar (cliente del troyano) mientras que la otra sirve de conexión con el PC controlado (servidor del troyano).
- El usuario del PC que actúa como servidor (el PC controlado) no tiene conciencia o conocimiento de estar comunicado con aquel otro PC. Ni tan siquiera es consciente de haber instalado el troyano en su PC.
- El servidor del troyano se oculta intentando pasar desapercibido para actuar clandestinamente.

Dado que la infección de nuestro PC por parte de un troyano es en realidad una instalación de una aplicación, nuestro sistema contará a partir de entonces con elementos nuevos dentro de nuestro disco duro, e incluso de la memoria. La instalación de un troyano en un PC viene acompañada de la creación de nuevas librerías DLL, de archivos ejecutables, de nuevas entradas en el registro de Windows y de nuevas líneas de comando en archivos tipo win.ini, system.ini,...

Otra forma algo menos complicada de llevar a cabo estas comprobaciones consistiría en la utilización de un monitor de sistema que localice más fácilmente los nuevos archivos creados.

También se podría usar el comando netstat (estado de la red) del propio Windows. Para ello abriríamos una ventana DOS en nuestro PC y teclearíamos “netstat –an”,

mostrándose las conexiones de nuestro PC que se están llevando a cabo en ese preciso momento y nos fijaríamos en si hay alguna que resulte sospechosa. Por ejemplo:

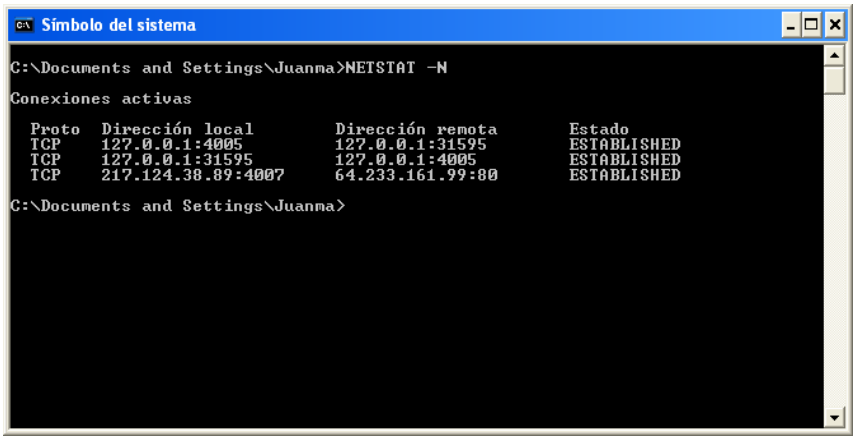

Figura 8: Conexiones del PC.

3.5.4.2 Hackers.

Hacker es una de las denominaciones de mayor prestigio en el sentido del conocimiento tecnológico, sin embargo los hackers más famosos son quienes cometieron delitos informáticos. Por lo tanto podría decirse que existen dos tipos de hackers, los que colaboran en el crecimiento de la tecnología y los que se aprovechan de sus conocimientos para llevar a cabo operaciones ilegales. Por otro lado, los crackers y script kiddies son en realidad quienes efectúan daños en las máquinas. Un hacker investiga, trata de solucionar problemas y es precisamente un cracker, y en menor medida un script kidd, quien se dedica a ocasionarlos.

3.5.4.3 Virus.

Los virus informáticos y gusanos se aprovechan de las vulnerabilidades de las aplicaciones para infectar el PC y reproducirse, siendo indetectable a la visión del usuario hasta que se lleva a cabo el ataque. Existen distintos tipos de virus, según sus fines, pero las dos características que los unen son: reproducirse y/o sobrevivir, y cumplir sus misiones. La mejor solución para proteger el PC de los virus es tener instalado un buen antivirus y mantenerlo actualizado.

3.5.4.4 Hoax.

Básicamente, un Hoax es una falsedad, que muchas personas creen y difunden, creada con ese fin. La definición más extendida la encontramos en el ámbito del mundo antivirus, y que considera a los Hoax como mensajes enviados por lo general mediante correo electrónico, aunque pueden aparecer muchas veces en papel fotocopiado en el ambiente empresarial, en los cuales se avisa en tono alarmante de la existencia de nuevos y potentes virus, muy peligrosos y destructivos. Estos supuestos virus en realidad no existen, o son archivos inofensivos sobre los que recae la elección de los creadores del Hoax. La finalidad de tal creación sería la de crear alarma entre los cibernautas y conseguir el reenvío masivo de dichos mensajes alarmistas. Entonces quizás la definición más acertada sería aquella que contemple a los Hoax como un mensaje cuyo contenido está basado en mentiras, engaños o falsedades, en ocasiones relativas a falsos virus, creado para su reenvío masivo. Dicho mensaje estaría creado bien para causar algún tipo de perjuicio o bien para diversión del/los creadores del mismo. El objetivo de estos avisos alarmantes es

difícil de establecer, ya que no se conocen los motivos que impulsan a alguien a crear un elemento perjudicial de ese tipo. Estas serían las motivaciones:

- Crear una "alarma social" entre los cibernautas. Se supone que ello resultaría divertido para el iniciador del hoax.
- Recopilar gran cantidad de direcciones de correo.
- Difundir información falsa en perjuicio de terceras personas u organismos.
- Incitar al receptor del mensaje a causar daños en su propio ordenador.
- Una mera competición en busca del mayor número de mensajes.
- La notoriedad que otorgaría en el círculo del creador del hoax una gran difusión del mismo.

3.5.4.5 Spam.

Spam es el recibo de correo electrónico no solicitado. El correo no deseado es el mayor problema que enfrenta hoy Internet. El spam es responsable de colapsar redes, provocar pérdidas millonarias y enfurecer al usuario hogareño saturando sus buzones. Quienes hacen spam justifican sus actos diciendo que realizan marketing electrónico, sin embargo no tienen en cuenta la situación límite a la que se ha llegado.

3.5.4.6 Dialers.

Sin lugar a dudas, una de las más recientes e impactantes amenazas de Internet. Lo que hace un dialer es conectar nuestro PC a Internet por medio de un número más costoso de lo normal. El problema reside en que los responsables no siempre advierten de la instalación de dicha modalidad en nuestro equipo, usualmente mientras visitamos una página Web. De esta manera, el usuario sólo advierte la trampa al recibir su factura telefónica. Los dialers afectan principalmente a los usuarios de España, en donde se han tomado medidas al respecto (denegar el uso de estos números telefónicos a usuarios que lo soliciten, y reducir los costos de llamada). Los dialers no se eliminan porque una gran parte de los sitios Web viven de ello, y no está mal siempre y cuando el usuario sea conciente de la modalidad que se le impone para acceder al contenido de una página.

3.6 Sistemas de detección de intrusos.

Existen numerosas medidas de seguridad para proteger los recursos informáticos de una empresa, pero aunque se sigan todas las recomendaciones de los expertos, no estaremos libres de posibles ataques con éxito. Esto se debe a que conseguir un sistema virtualmente invulnerable es sumamente costoso, además de que las medidas de control reducirían la productividad de la empresa [12].

Llamaremos intrusión a un conjunto de acciones que intentan comprometer la integridad, confidencialidad o disponibilidad de un recurso, podemos darnos cuenta de que una intrusión no tiene por qué consistir en un acceso no autorizado a una máquina, también puede ser una negación de servicio. A los sistemas utilizados para detectar las intrusiones o los intentos de intrusión se les denomina sistemas de detección de intrusiones Intrusión Detection Systems, IDS) o, más habitualmente sistemas de detección de intrusos, cualquier mecanismo de seguridad con este propósito puede ser considerado un IDS, pero generalmente sólo se aplica esta denominación a los sistemas automáticos (software o hardware).

Una de las primeras cosas que deberíamos plantearnos a la hora de hablar de IDSs es si realmente necesitamos uno de ellos en nuestro entorno de trabajo, a fin de cuentas, debemos tener ya un sistema de protección perimetral basado en cortafuegos, y por si nuestro firewall fallara, cada sistema habría de estar configurado de una manera correcta, de forma que incluso sin cortafuegos cualquier máquina pudiera seguirse considerando relativamente segura. La respuesta es, sin duda, sí, debemos esperar que en cualquier momento alguien consiga romper la seguridad de nuestro entorno informático, y por tanto hemos de ser capaces de detectar ese problema tan pronto como sea posible (incluso antes de que se produzca, cuando el potencial atacante se limite a probar suerte contra nuestras máquinas). Ningún sistema informático puede considerarse completamente seguro, pero incluso aunque nadie consiga violar nuestras políticas de seguridad, los sistemas de detección de intrusos se encargarían de mostrarnos todos los intentos de multitud de piratas para penetrar en nuestro entorno, no dejándonos caer en ninguna falsa sensación de seguridad, si somos conscientes de que a diario hay gente que trata de romper nuestros sistemas, no caeremos en la tentación de pensar que nuestras máquinas están seguras porque nadie sabe de su existencia o porque no son interesantes para un pirata.

Los sistemas de detección de intrusos no son precisamente nuevos, el primer trabajo sobre esta materia data de 1980, no obstante, este es uno de los campos más en auge desde hace ya unos años dentro de la seguridad informática. Y no es extraño, la capacidad para detectar y responder ante los intentos de ataque contra nuestros sistemas es realmente muy interesante. Durante estos veinte años, cientos de investigadores de todo el mundo han desarrollado, con mayor o menor éxito, sistemas de detección de todo tipo, desde simples procesadores de logs hasta complejos sistemas distribuidos, especialmente vigentes con el auge de las redes de computadores en los últimos años.

Dentro de las soluciones tecnológicas que en la actualidad están disponibles para reforzar la seguridad de una red encontramos los firewalls los cuales son muy populares. Un firewall es un sistema encargado del cumplimiento de las políticas de control de acceso a la red, lo cual se hace a través de reglas. Un firewall actúa como guardia perimetral de una red protege una red de ataques que provengan del exterior de ésta. Pero el escenario se puede complicar de la siguiente forma:

- Un atacante puede lograr pasar el firewall, dejando la red a su merced.
- Un firewall protege de los accesos no autorizadas hacia la red interna, pero no protege a las máquinas ubicadas en la red perimetral como servidores web, servidores de correo, servidores FTP, en otras palabras, a las bases funcionales de Internet.
- Un firewall no protege contra ataques desde adentro.

En estos casos lo que nos queda detectar el ataque o la intrusión lo antes posible para que cause el menor daño en el sistema. Antes de continuar vamos a definir qué se entiende normalmente por intrusión. Normalmente un intruso intenta:

- Acceder a una determinada información.
- Manipular cierta información.
- Hacer que el sistema se no funcione de forma segura o inutilizarlo.

Una intrusión es cualquier conjunto de acciones que puede comprometer la integridad, confidencialidad o disponibilidad de una información o un recurso informático. Los intrusos pueden utilizar debilidades y brechas en la arquitectura de los sistemas y el

conocimiento interno del sistema operativo para superar el proceso normal de autenticación.

La detección de intrusos se puede detectar a partir de la caracterización anómala del comportamiento y del uso que hacen de los recursos del sistema. Este tipo de detección pretende cuantificar el comportamiento normal de un usuario. Para una correcta distinción hay que tener en cuenta las tres distintas posibilidades que existen en un ataque, atendiendo a quién es el que lo lleva a cabo:

- Penetración externa. Que se define como la intrusión que se lleva a cabo a partir un usuario o un sistema de computadores no autorizado desde otra red.
- Penetraciones internas. Son aquellas que llevan a cabo por usuarios internos que no están autorizados al acceso.
- Abuso de recursos. Se define como el abuso que un usuario lleva a cabo sobre unos datos o recursos de un sistema al que está autorizado su acceso.

La idea central de este tipo de detección es el hecho de que la actividad intrusiva es un subconjunto de las actividades anómalas. Esto puede parecer razonable por el hecho de que si alguien consigue entrar de forma ilegal en el sistema, no actuará como un usuario normal. Sin embargo en la mayoría de las ocasiones una actividad intrusiva resulta del agregado de otras actividades individuales que por sí solas no constituyen un comportamiento intrusivo de ningún tipo. Idealmente el conjunto de actividades anómalas es el mismo del conjunto de actividades intrusivas, de todas formas esto no siempre es así:

- Intrusivas pero no anómalas. Se les denomina falsos negativos y en este caso la actividad es intrusiva pero como no es anómala y no se consigue detectarla. Se denominan falsos negativos porque el sistema erróneamente indica ausencia de intrusión.
- No intrusivas pero anómalas. Se denominan falsos positivos y en este caso la actividad es no intrusiva, pero como es anómala el sistema decide que es intrusiva. Se denominan falsos positivos, porque el sistema erróneamente indica la existencia de intrusión.
- Ni intrusiva ni anómala. Son negativos verdaderos, la actividad es no intrusiva y se indica como tal.
- Intrusiva y anómala. Se denominan positivos verdaderos, la actividad es intrusiva y es detectada.

Los primeros no son deseables, porque dan una falsa sensación de seguridad del sistema y el intruso en este caso puede operar libremente en el sistema. Los falsos positivos se deben de minimizar, en caso contrario lo que puede pasar es que se ignoren los avisos del sistema de seguridad, incluso cuando sean acertados. Los detectores de intrusiones anómalas requieren mucho gasto computacional, porque se siguen normalmente varias métricas para determinar cuánto se aleja el usuario de lo que se considera comportamiento normal.

Hoy día existen en el mercado una buena cantidad de productos conocidos como IDS (Sistemas de Detección de Intrusos) o en inglés IDS (Intrusión Detection System). Estos sistemas basan su funcionamiento en la recolección y análisis de información de diferentes fuentes, que luego utilizan para determinar la posible existencia de un ataque o penetración de intrusos. En caso de que exista la suficiente certeza de la detección de un incidente, el IDS tiene como función principal alertar al administrador o personal de

seguridad, para que tome acciones al respecto. Otras implementaciones más complejas son capaces de ir más allá de la notificación de un posible ataque, es decir pueden ejecutar acciones automáticas que impidan el desarrollo de éste.

3.6.1 Clasificación de los IDS.

Existen varios tipos de IDS, clasificados según el tipo de situación física, del tipo de detección que posee o de su naturaleza y reacción cuando detecta un posible ataque.

3.6.1.1 Clasificación por situación.

Se pueden clasificar como [13]:

- HIDS (HostIDS): un IDS vigilando un único ordenador y por tanto su interfaz corre en modo no promiscuo. La ventaja es que la carga de procesado es mucho menor.
- NIDS (NetworkIDS): un IDS basado en red, detectando ataques a todo el segmento de la red. Su interfaz debe funcionar en modo promiscuo capturando así todo el tráfico de la red.
- DIDS (DistributedIDS): sistema basado en la arquitectura cliente/servidor compuesto por una serie de NIDS (IDS de redes) que actúan como sensores centralizando la información de posibles ataques en una unidad central que puede almacenar o recuperar los datos de una base de datos centralizada. La ventaja es que en cada NIDS se puede fijar unas reglas de control especializándose para cada segmento de red. Estructura habitual en redes VPN (Redes Privadas Virtuales).

Los NIDS analizan el tráfico de la red completa, examinando los paquetes individualmente, comprendiendo todas las diferentes opciones que pueden coexistir dentro de un paquete de red y detectando paquetes armados maliciosamente y diseñados para no ser detectados por los cortafuegos. Pueden buscar cual es el programa en particular del servidor de Web al que se está accediendo y con que opciones y producir alertas cuando un atacante intenta explotar algún fallo en este programa. Los NIDS tienen dos componentes:

- Un sensor: situado en un segmento de la red, la monitoriza en busca de tráfico sospechoso.
- Una Consola: recibe las alarmas del sensor o sensores y dependiendo de la configuración reacciona a las alarmas recibidas.

Las principales ventajas del NIDS son:

- Detectan accesos no deseados a la red.
- No necesitan instalar software adicional en los servidores en producción.
- Fácil instalación y actualización por que se ejecutan en un sistema dedicado.

Como principales desventajas se encuentran:

- Examinan el tráfico de la red en el segmento en el cual se conecta, pero no puede detectar un ataque en diferentes segmentos de la red. La solución más sencilla es colocar diversos sensores.
- Pueden generar tráfico en la red.
- Ataques con sesiones encriptadas son difíciles de detectar.

En cambio, los HIDS analizan el tráfico sobre un servidor o un PC, se preocupan de lo que está sucediendo en cada host y son capaces de detectar situaciones como los intentos fallidos de acceso o modificaciones en archivos considerados críticos.

Las ventajas que aporta el HIDS son:

- Herramienta potente, registra comandos utilizados, ficheros abiertos,...
- Tiende a tener menor número de falsos-positivos que los NIDS, entendiendo falsos-positivos a los paquetes etiquetados como posibles ataques cuando no lo son.
- Menor riesgo en las respuestas activas que los IDS de red.

Los inconvenientes son:

- Requiere instalación en la máquina local que se quiere proteger, lo que supone una carga adicional para el sistema.
- Tienden a confiar en las capacidades de auditoria y logging de la máquina en sí.

Hay autores que dividen el grupo HIDS (HostIDS) en tres subcategorías:

- Verificadores de integridad del sistema (SIV). Un verificador de integridad no es más que un mecanismo encargado de monitorizar archivos de una máquina en busca de posibles modificaciones no autorizadas, por norma general backdoors dejadas por un intruso (por ejemplo, una entrada adicional en el fichero de contraseñas o un */bin/login* que permite el acceso ante cierto nombre de usuario no registrado).
- Monitores de registros (LFM). Estos sistemas monitorizan los archivos de log generados por los programas (generalmente demonios de red) de una máquina en busca de patrones que puedan indicar un ataque o una intrusión. Un ejemplo de monitor puede ser swatch, pero más habituales que él son los pequeños shelllscripts que casi todos los administradores realizan para comprobar periódicamente sus archivos de log en busca de entradas sospechosas (por ejemplo, conexiones rechazadas en varios puertos provenientes de un determinado host, intentos de entrada remota como root…).
- Sistemas de decepción. Los sistemas de decepción o tarros de miel (honeypots), como Deception Toolkit (DTK), son mecanismos encargados de simular servicios con problemas de seguridad de forma que un pirata piense que realmente el problema se puede aprovechar para acceder a un sistema, cuando realmente se está aprovechando para registrar todas sus actividades. Se trata de un mecanismo útil en muchas ocasiones (por ejemplo, para conseguir *entretener* al atacante mientras se tracea su conexión) pero que puede resultar peligroso, por ejemplo ¿qué sucede si el propio sistema de decepción tiene un bug que desconocemos, y el atacante lo aprovecha para acceder realmente a nuestra máquina?

3.6.1.2 Clasificación según los modelos de detecciones.

Los dos tipos de detecciones que pueden realizar los IDS son:

- Detección del mal uso.
- Detección del uso anómalo.

La detección del mal uso involucra la verificación sobre tipos ilegales de tráfico de red, por ejemplo, combinaciones dentro de un paquete que no se podrían dar

legítimamente. Este tipo de detección puede incluir los intentos de un usuario por ejecutar programas sin permiso (por ejemplo, "sniffers"). Los modelos de detección basados en el mal uso se implementan observando como se pueden explotar los puntos débiles de los sistemas, describiéndolos mediante unos patrones o una secuencia de eventos o datos ("firma") que serán interpretados por el IDS.

La detección de actividades anómalas se apoya en estadísticas tras comprender cual es el tráfico "normal" en la red del que no lo es. Un claro ejemplo de actividad anómala sería la detección de tráfico fuera de horario de oficina o el acceso repetitivo desde una máquina remota (rastreo de puertos). Este modelo de detección se realiza detectando cambios en los patrones de utilización o comportamiento del sistema. Esto se consigue realizando un modelo estadístico que contenga una métrica definida y compararlo con los datos reales analizados en busca de desviaciones estadísticas significantes. Este tipo de detección es bastante compleja, debido a que la cuantificación de los parámetros a observar no es sencilla y a raíz de esto, se pueden presentar los siguientes inconvenientes:

- Pueden generarse falsas alarmas si el ambiente cambia repentinamente, por ejemplo, cambio en el horario de trabajo.
- Un atacante puede ir cambiando lentamente su comportamiento para así engañar al sistema.

Los inconvenientes antes mencionados pueden ser controlados mediante una implementación robusta y minuciosa.

3.6.1.3 Clasificación según su naturaleza.

Un tercer y último tipo básico de clasificación sería respecto a la reacción del IDS frente a un posible ataque:

- Pasivos.
- Reactivos.

Los IDS pasivos detectan una posible violación de la seguridad, registran la información y genera una alerta.

Los IDS reactivos están diseñados para responder ante una actividad ilegal, por ejemplo, sacando al usuario del sistema o mediante la reprogramación de los cortafuegos para impedir el tráfico desde una fuente hostil.

3.6.2 Topologías de los IDS.

Existen muchas formas de añadir las herramientas IDS a nuestra red, cada una de ellas tiene su ventaja y su desventaja. La mejor opción debería ser un compendio entre coste económico y propiedades deseadas, manteniendo un alto nivel de ventajas y un número controlado de desventajas, todo ello de acuerdo con las necesidades de la organización.

Por este motivo, las posiciones de los IDS dentro de una red son varias y aportan diferentes características. A continuación vamos a ver diferentes posibilidades en una misma red. Imaginemos que tenemos una red dónde el cortafuegos nos divide Internet de la zona desmilitarizada (DMZ – Demilitarized Zone), y otro que divide la DMZ de la intranet de la organización como se muestra en la figura 7. Por zona desmilitarizada entendemos la zona que debemos mostrar al exterior, la zona desde la cual mostramos nuestros servicios o productos:

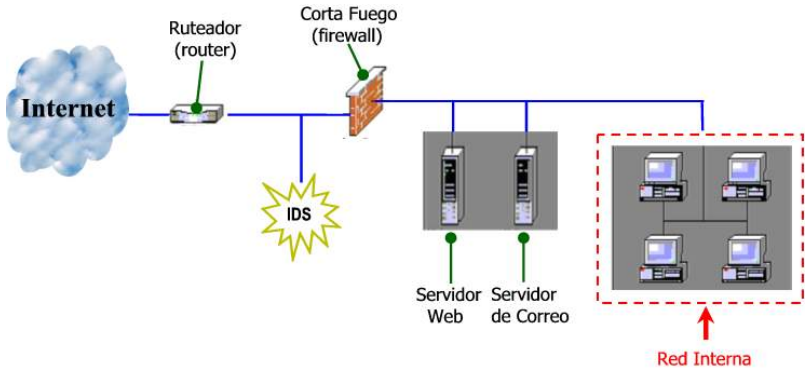

Figura 9: Red con IDS simple

Si situamos un IDS antes del cortafuegos exterior permitiría detectar el rastreo de puertos de reconocimiento que señala el comienzo de una actividad hacking, y obtendríamos como ventaja un aviso prematuro. Sin embargo, si los rastreos no son seguidos por un ataque real, se generará un numeroso número de alertas innecesarias con el peligro de comenzar a ignorarlas.

Si optamos por colocar el IDS en la zona desmilitarizada (DMZ) tendríamos como ventaja la posibilidad de adecuar la base de datos de atacantes del NIDS para considerar aquellos ataques dirigidos a los sistemas que están en la DMZ (servidor Web y servidor de correo) y configurar el cortafuegos para bloquear ese tráfico.

Así mismo, un NIDS dentro de la red, por ejemplo, de Recursos Humanos podría monitorear todo el tráfico para fuera y dentro de esa red. Este NIDS no debería ser tan poderoso como los comentados anteriormente, puesto que el volumen y el tipo de tráfico es más reducido. El resultado lo podemos visualizar en la siguiente figura:

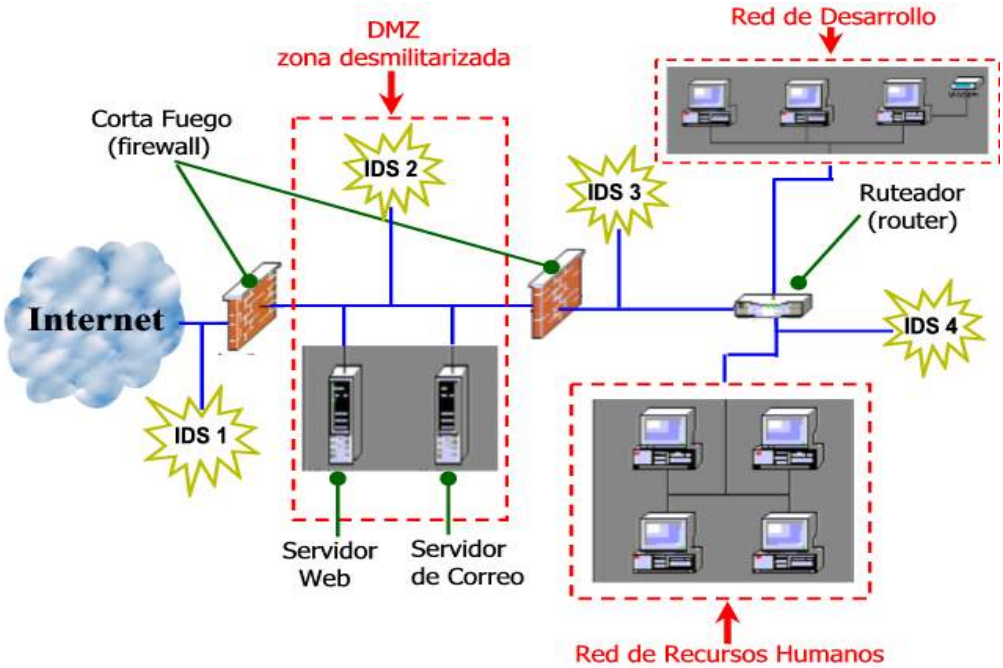

Figura 10: Red completa con IDS.

El IDS1 se encargaría de avisar del rastreo de puertos, y si es reactivo podría enviar un "aviso" tanto al que esta rastreando (por ejemplo un ping a la dirección que emite el paquete) como al encargado de la seguridad de la organización. El IDS2 se encargaría de vigilar la zona desmilitarizada y analizar el tráfico que reciben tanto el servidor Web como el servidor de correo. Los otros dos IDS se encargarían de la red interna, el IDS3 de la totalidad de la red, y el IDS4 de una subred, en este caso la de RRHH. Estos dos NIDS internos (el IDS3 y el IDS4) podrían ser sensores que recogiesen la información y lo enviasen a una consola dónde se realizarían los cálculos.

3.6.3 Arquitecturas basadas en IDS.

Las arquitecturas que implementan sistemas de detección de intrusos han ido modificándose y mejorando con el paso del tiempo y con la experiencia. Los primeros IDS eran herramientas software rígidos, diseñados y realizados bajo la supervisión de un experto en seguridad de redes y basándose en la experiencia personal. Eran verdaderos sistemas rígidos, dónde la actualización ante nuevos tipos de ataques requería verdaderos esfuerzos de análisis y programación, además de contar con un solo programa que realizaba todo el trabajo, sin pensar en sistemas distribuidos.

Actualmente, las arquitecturas empleadas para el desarrollo de herramientas IDS, se basan fundamentalmente en dos principios básicos, la utilización de Agentes autónomos que recogen información por separado, para luego analizar una parte de esta información ellos y la otra una entidad central coordinadora, y las arquitecturas basadas en la exploración de los datos en tiempo real. Y como ocurre en el mundo de la informática, se dan arquitecturas híbridas en busca de la mejor solución posible.

3.6.3.1 Arquitecturas basadas en agentes autónomos.

Basa su desarrollo en las carencias y limitaciones que presentan los IDS existentes. La principal carencia de los viejos (y primeros) IDS es que los datos son recogidos por un solo host, además, algunos de los que intentan solucionar esta desventaja utilizan una colección de datos distribuida, pero analizada por una consola central.

Los principales problemas de utilizar estas arquitecturas son:

- La consola central es un solo punto de fallo, la red entera esta sin protección si esta falla.
- La escalabilidad esta limitada. Procesar toda la información en un host implica un límite en el tamaño de la red que puede monitorizar.
- Dificultad en reconfigurar o añadir capacidades al IDS.
- El análisis de los datos de la red puede ser defectuoso.

Basándose en estas limitaciones, se ha introducido el término de Agente Autónomo definido como una entidad software capaz de analizar actividades de una manera flexible e inteligente, capaz de funcionar continuamente y de aprender por su experiencia y la de otros agentes, además de comunicarse y cooperar con ellos. Los agentes son autónomos porque son entidades que se ejecutan independientemente y no necesitan información de otros agentes para realizar su trabajo.

Las principales ventajas que puede aportar una arquitectura basada en Agentes Autónomos residen en que si un agente para de trabajar por cualquier motivo, solamente sus resultados se pierden, sin afectar a otros agentes autónomos. Además, la posible organización de los agentes en estructuras jerárquicas con diferentes capas hace un sistema

escalable. Por estos motivos, un agente autónomo puede cruzar la barrera de los HIDS y los NIDS y realizar ambas funciones, así como estar implementado en el lenguaje que mejor le convenga para cada caso.

Los componentes de la arquitectura basada en Agentes Autónomos se basa en tres componentes esenciales: agentes, transceivers y monitores.

Un agente autónomo puede ser distribuido sobre cualquier número de hosts en una red. Cada host puede contener un número de agentes que monitoriza una cierta característica (cada uno). Todos los agentes de un host envían sus resultados a un transceiver y estos envían sus resultados globales del host a uno o más monitores.

Una posible topología de esta red sería la descrita a continuación:

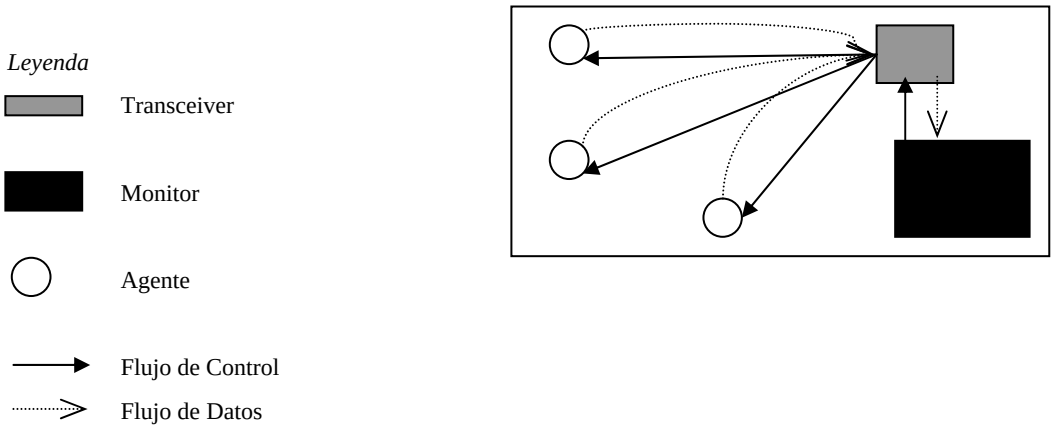

Figura 11: Arquitectura de un sistema autónomo.

Un agente se encarga de monitorizar un aspecto del sistema total, por ejemplo, un agente puede estar destinado a monitorizar y vigilar las conexiones telnet de un host protegido. Este agente genera un informe y lo envía a su respectivo transceiver. El agente no puede generar una alarma por sí mismo, solo informa.

Un transceiver es la interfaz externa de comunicación de cada host. Tiene varios agentes a su servicio de los que recibe informes y construye un mapa de estado del host al que pertenece. Tiene dos funciones principales, una de control y otra de procesamiento de datos. Como funciones de control, debe inicializar o parar los agentes de su host y ejecutar los comandos que le envía el monitor al que pertenece. Como funciones de procesamiento, recibe informes de los agentes de su host, los procesa y la información obtenida la envía al monitor o lo reparte con los agentes.

El monitor es la entidad de más alto nivel. Tiene funciones parecidas a las del transceiver, la principal diferencia es que un monitor puede controlar entidades que están ejecutándose en diferentes hosts, además, la información que recibe de los transceivers es limitada y solamente un monitor es capaz de obtener conclusiones más refinadas de esta información.

Otras entidades opcionales que se pueden implementar en las arquitecturas basadas en Agentes Autónomos son los agentes SNMP o los auditores de routers.

La principal desventaja de esta arquitectura es que si un monitor detiene su ejecución, todos los transceivers que controla paran de producir información.

3.6.3.2 Arquitectura de exploración de datos en tiempo real.

El principal punto negativo de estas arquitecturas es que necesitan un gran paquete de datos de entrenamiento más complicados que los utilizados en los sistemas tradicionales.

Las principales características que intentan aportar las arquitecturas basadas en análisis de datos en tiempo real son la exactitud o veracidad, la eficiencia (en cuanto a coste computacional) y la facilidad de uso:

- veracidad: medido según el ratio de detección (ataques que puede detectar el sistema) y el ratio de falsos positivos (porcentaje de datos normales que el sistema determina como intruso cuando no lo es). En la práctica, solo un ratio falso-positivo bajo puede tolerarse.
- Eficiencia: un IDS en tiempo real debe detectar las intrusiones lo más pronto posible. Ante grandes flujos de datos a analizar, el tiempo tomado en procesar cada registro y calcular las estadísticas de sus características es demasiado elevado. Este "retraso" en el cálculo puede propiciar un ataque que tardaría en detectarse. La principal característica de la eficiencia es el coste computacional, dividido en 4 niveles:
    - Nivel 1: características que pueden ser computados con el primer paquete recibido
    - Nivel 2: características que pueden ser analizadas en cualquier momento de la conexión.
    - Nivel 3: características analizadas al final de la conexión.
    - Nivel 4: estadísticas computadas al final de la conexión.
- Usabilidad: entendido como facilidad de crear el paquete de entrenamiento y de actualizar los patrones utilizados por el IDS.

Para detectar los intrusos, debemos tener un paquete de características cuyos valores en los campos de los paquetes analizados difieran de los valores de los campos de los paquetes intrusos. Para poder llevar a cabo esta tarea, el flujo de datos (en binario) es analizado y procesado en registros que contienen un número básico de características para más tarde ser analizados por programas especializados que comparan estos registros con los patrones de la base de datos en búsqueda de variaciones o combinaciones peligrosas.

Estos patrones pueden ser entrenados para comprobar su correcto funcionamiento generando anomalías de forma artificial, utilizando una heurística para cambiar el valor de un campo dejando los demás como estaban. Para a continuación introducir este flujo de datos en una conexión normal.

El IDS ejecuta un conjunto de reglas, con coste computacional creciente o con combinaciones de los niveles coste computacional para obtener una buena eficiencia y exactitud en tiempo real acorde con las características deseadas. Cada modelo ejecuta un conjunto de reglas.

Los componentes de la arquitectura de IDS basada en tiempo real consisten en sensores, detectores, almacén de datos y modelos de generación de características.

El sistema esta diseñado para ser independiente del formato de datos de los sensores y del modelo utilizado. Este modelo puede estar representado como una red neuronal, un

conjunto de reglas o un modelo probabilístico. La codificación XML permitiría intercambiar datos entre los diferentes modelos.

Si todos los componentes residen en la misma área local, la carga de trabajo puede distribuirse sobre todos los componentes, si están distribuidos por diferentes redes, pueden colaborar con otros IDS en Internet, intercambiando información de nuevos ataques.

Este es el diseño de una arquitectura basada en tiempo real:

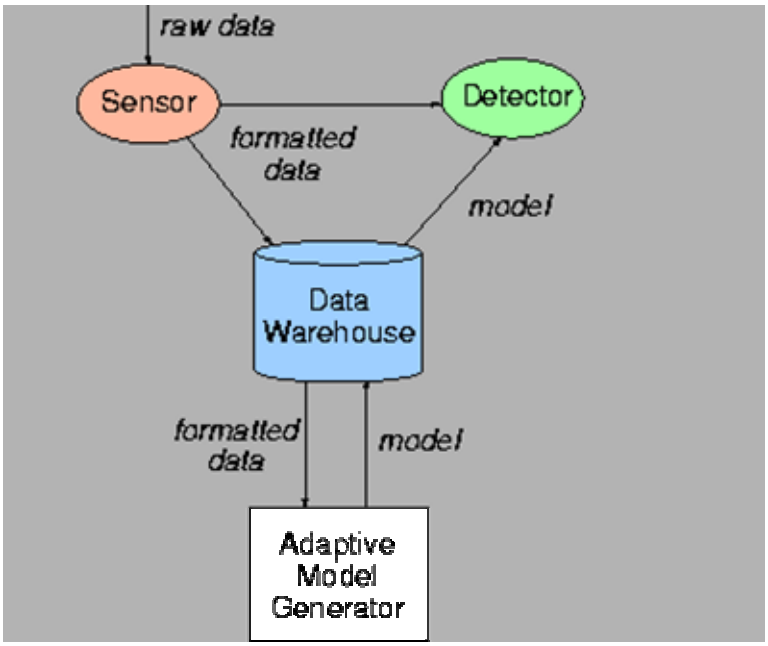

Figura 12: Arquitectura de un IDS basado en tiempo real.

Los sensores observan los bits en un sistema monitorizado y analiza sus características para utilizarlas en el modelo.

Los detectores procesan los datos de los sensores y utilizan un modelo de detección para evaluar los datos y determinar si es un ataque. Además, envía el resultado a un almacén de datos (típicamente una base de datos) para otro posterior análisis y para generar informes. Pueden coexistir detectores monitorizando el mismo sistema (en paralelo). Los detectores "back-end" emplearían modelos con coste computacional complejo mientras que los detectores "front-end" realizarían una detección de intrusión sencilla y rápida.

Por último, la centralización del almacenamiento de los datos y de los modelos aportaría varias ventajas:

- Eficiencia: varios componentes podrían manejar el mismo tipo de datos.
- Operatividad: los datos de los sensores podrían ser consultados con una simple SQL.
- Rendimiento: la detección de complicados ataques y ataques a gran escala podría ser posible mediante el conocimiento y el manejo de los datos por todos los detectores.

3.6.4 Características deseables de un IDS.

- Debe ejecutarse continuamente sin intervención o supervisión de un operador humano.

- Debe ser confiable, lo suficiente como para ejecutarse en background, pero no debe ser una caja negra, es decir, que su funcionamiento interno pueda ser examinado.
- Debe ser capaz de tolerar fallas, en el sentido de que pueda sobrevivir a una caída del sistema, sin tener que reconstruir su base de datos de conocimientos al reiniciarse.
- El sistema debe estar en capacidad de automonitorearse para asegurar su correcto funcionamiento.
- Debe ser ligero, es decir su ejecución no debe cargar al sistema de una manera tal que le impida ejecutar otras tareas con relativa normalidad
- Debe observar desviaciones del comportamiento estándar.
- Debe poder adaptarse al comportamiento cambiante del sistema, es decir, si la configuración del sistema cambia, el IDS se adaptará.
- Debe ser difícil de engañar.

Sin importar qué sistemas vigile o su forma de trabajar, cualquier sistema de detección de intrusos ha de cumplir algunas propiedades para poder desarrollar su trabajo correctamente. En primer lugar, y quizás como característica más importante, el IDS ha de ejecutarse continuamente sin nadie que esté obligado a supervisarlo, independientemente de que al detectar un problema se informe a un operador o se lance una respuesta automática, el funcionamiento habitual no debe implicar interacción con un humano. Podemos fijarnos en que esto parece algo evidente, muy pocas empresas estarían dispuestas a contratar a una o varias personas simplemente para analizar logs o controlar los patrones del tráfico de una red. Sin entrar a juzgar la superioridad de los humanos frente a las máquinas (¿puede un algoritmo determinar perfectamente si un uso del sistema está correctamente autorizado?) o viceversa (¿sería capaz una persona de analizar en tiempo real todo el tráfico que llega a un servidor Web mediano?), hemos de tener presente que los sistemas de detección son mecanismos automatizados que se instalan y configuran de forma que su trabajo habitual sea transparente a los operadores del entorno informático.

Otra propiedad, y también como una característica a tener siempre en cuenta, es la aceptabilidad o grado de aceptación del IDS, al igual que sucedía con cualquier modelo de autenticación, los mecanismos de detección de intrusos han de ser aceptables para las personas que trabajan habitualmente en el entorno. Por ejemplo, no ha de introducir una sobrecarga considerable en el sistema (si un IDS ralentiza demasiado una máquina, simplemente no se utilizará) ni generar una cantidad elevada de falsos positivos (detección de intrusiones que realmente no lo son) o de logs, ya que entonces llegará un momento en que nadie se preocupe de comprobar las alertas emitidas por el detector. Por supuesto (y esto puede parecer una tontería, pero es algo que se hace más a menudo de lo que podamos imaginar), si para evitar problemas con las intrusiones simplemente apagamos el equipo o lo desconectamos de la red, tenemos un sistema bastante seguro… pero inaceptable.

Una tercera característica a evaluar a la hora de hablar de sistemas de detección de intrusos es la adaptabilidad del mismo a cambios en el entorno de trabajo. Como sabemos, ningún sistema informático puede considerarse estático, desde la aplicación más pequeña hasta el propio kernel de Unix, pasando por supuesto por la forma de trabajar de los usuarios (¿quién nos asegura que ese engorroso procedimiento desde una “desfasada” línea de órdenes mañana no se realizará desde una aplicación gráfica, que realmente hace el mismo trabajo pero que genera unos patrones completamente diferentes en nuestro sistema?), todo cambia con una periodicidad más o menos elevada. Si nuestros

mecanismos de detección de intrusos no son capaces de adaptarse rápidamente a esos cambios, están condenados al fracaso.

Todo IDS debe además presentar cierta tolerancia a fallos o capacidad de respuesta ante situaciones inesperadas, insistiendo en lo que comentábamos antes sobre el carácter altamente dinámico de un entorno informático, algunos de los cambios que se pueden producir en dicho entorno no son graduales sino bruscos, y un IDS ha de ser capaz de responder siempre adecuadamente ante los mismos. Podemos contemplar, por ejemplo, un reinicio inesperado de varias máquinas o un intento de engaño hacia el IDS, esto último es especialmente crítico, sólo hemos de pararnos a pensar que si un atacante consigue modificar el comportamiento del sistema de detección y el propio sistema no se da cuenta de ello, la intrusión nunca será notificada, con los dos graves problemas que eso implica, aparte de la intrusión en sí, la falsa sensación de seguridad que produce un IDS que no genera ninguna alarma es un grave inconveniente de cara a lograr sistemas seguros.

3.6.5 Metodología para la detección de intrusos y para la selección e implantación de sistemas IDS.

La labor de un administrador o de la persona encargada de la seguridad de un sistema informático puede ser realmente frustrante. Sobre todo cuando el sistema a sido invadido por un intruso o hacker. En principio, si se ha configurado correctamente un servidor y se está al día en materia de seguridad, así como de fallas (bugs) que van surgiendo, no habrá problemas de que un intruso entre en el sistema. Realmente con un poco de esfuerzo se puede tener un servidor altamente seguro que evitará alrededor del 85% de los intentos de acceso no autorizados alo sistema. Pero en muchas ocasiones el peligro viene de los propios usuarios internos del sistema, los cuales presentan un gran riesgo debido a que ya tiene acceso al sistema.

Pasos a seguir para detectar a un intruso. Lo primero que debemos hacer es seguir una serie de pasos los cuales nos ayudarán a descubrir si realmente ha entrado un intruso, ya que en muchas ocasiones pensamos que ha entrado alguien, pero no es cierto. Realmente en muchas ocasiones es fácil detectar a un intruso en ambiente Unix, ya que suelen seguir un patrón detectable, el cual podría ser el mostrado en la figura. Este esquema representa básicamente los pasos que sigue de un intruso, Primero entra al sistema, y si sólo tiene acceso como usuario, explotará alguna debilidad o falla del sistema para así obtener ID 0 (o lo que es lo mismo, privilegios de root). En caso de entrar como root u obtenerlo de alguna otra manera, se dedicará a controlar el sistema, dejando algún mecanismo para volver cuando quiera. Seguramente copiará el archivo /etc/passwd y el /etc/shadow (en caso de que el sistema use "shadow"), luego le dará rienda suelta a su imaginación, como por ejemplo, instalar un sniffer, troyanos, leer mails ajenos, etc. Y en caso de ser un pirata malicioso puede causar desastres en el sistema, como sería modificar paginas Web, borrar archivos o mails, producir un DoS (Denial of Service), cambiar passwords de usuarios legítimos, etc.

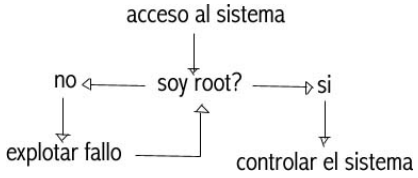

Figura 13: Pasos de un intruso.

A continuación se exponen los diferentes pasos a seguir de acuerdo a los expertos en seguridad como son el CERT, ISS, etc. [14]

1. Examinar los archivos log como el “last” log, contabilidad, syslog, y los C2 log buscando conexiones no usuales o cosas sospechosas en el sistema. Aunque hay que tener especial cuidado en guiarnos por los logs, ya que muchos intrusos utilizaran diversas herramientas para borrar sus huellas.
2. Buscar por el sistema archivos ocultos o no usuales (archivos que empiezan por un “.” (punto), no salen con un simple “ls”), ya que pueden ser usado para esconder herramientas para violar la seguridad del sistema, por ejemplo un crackeador o incluso contener el /etc/passwd del sistema o de otros sistemas al cual ha entrado nuestro intruso. Muchos piratas suelen crear directorios ocultos utilizando nombres como “...” (punto-punto-punto), “..” (punto-punto), “..^g” (punto-punto control+G). En algunos casos un pirata ha utilizado nombres como “.x” o “.hacker” o incluso “.mail”.
3. Buscar archivos SET-UID por el sistema. Ya que en muchas ocasiones los piratas suelen copiar y dejar escondido copias del /bin/sh para obtener root. Podemos utilizar el comando “find” para buscar este tipo de archivos por el sistema (el comando “find” puede ser sustituido por un troyano para esconder archivos del pirata, por lo que no es totalmente confiable), para ello ejecutamos la siguiente línea: # find / -user root -perm -4000 –print
4. Revisar los archivos binarios del sistema para comprobar que no han sido sustituidos por un troyano, como por ejemplo los programas “su”, “login”, “telnet” y otros programas vitales del sistema. (Existen varias herramientas conocidas como “RootKit” que permite a un pirata cambiar los binarios del sistema por troyanos que son copias exactas de los originales). Lo recomendado es comparar con las copias de seguridad aunque puede que las copias de seguridad también hayan sido sustituidas por un troyano.
5. Examinar todos los archivos que son ejecutados por “cron” y “at”. Ya que algunos piratas depositan puertas traseras que le permiten volver al sistema aunque los hayamos echado del sistema. Asegurarse que todos los archivos son nuestros y no tienen permiso de escritura.
6. Examinar el archivo /etc/inetd.conf en busca de cambios, en especial aquellas entradas que ejecuten un shell (por ejemplo: /bin/sh o /bin/csh) y comprobar que todos los programas son legítimos del sistema y no troyanos.
7. Examinar los archivos del sistema y de configuración en busca de alteraciones. En particular, buscar entradas con el signo “+” o “host names” no apropiados en archivos como /etc/hosts.equiv, /etc/hosts.lpd y en todos los archivos .rhost del sistema, con especial interés los de “root”, “uucp”, “ftp” y otras cuentas del sistema. Estos archivos no deberían tener atributo de escritura.
8. Examinar cuidadosamente todos los computadores de nuestra red local en busca de indicios que nuestra red ha sido comprometida. En particular, aquellos sistemas que compartan NIS+ o NFS, o aquellos sistemas listados en el /etc/hosts.equiv. Lógicamente también revisar los sistemas informáticos que los usuarios comparten mediante el acceso del .rhost.
9. Examinar el archivo /etc/passwd, en busca de alteraciones en las cuentas de los usuarios o la creación de cuentas nuevas, especialmente aquellas cuentas con ID 0, las que no tienen password, etc.

Estos nueve puntos son los pasos a seguir recomendados por el CERT, los cuales se quedan un poco cortos de soluciones prácticas para el administrador. Para ello explicaremos los métodos de los piratas y como combatirlos [15]:

- Los archivos Log:

messages: Este archivo contiene bastante información, por lo que debemos buscar sucesos poco usuales.

xferlog: Si el sistema comprometido tiene servicio FTP, este archivo contiene la bitácora de todos los procesos del FTP. Podemos examinar qué tipo de herramientas ha subido el pirata y qué archivos ha bajado de nuestro servidor.

utmp: Este archivo contiene información en binario de todos los usuarios conectados al sistema en el momento. Por lo que puede ser muy útil para determinar quién esta conectado al sistema en este momento. Para ello ejecutaremos el comando “who” o “w”.

wtmp: Cada vez que un usuario entra al servidor y sale del mismo, la máquina modifica este archivo. Al igual que el anterior, este archivo está en binario, por lo que tendremos que usar alguna herramienta especial para ver el contenido de este archivo. El mismo contiene la información en formato: usuario, hora de conexión, e IP origen del usuario, por lo que podemos averiguar de dónde provino el pirata. Pero aunque contemos con esta información, puede que haya sido falseada por el pirata utilizando alguna técnica para ocultar su IP original o haya borrado su entrada.

secure: Algunos sistemas Unix loggean mensajes en archivo secure, ya que utilizan algún software de seguridad para ello, como el TCP Wrapper.

Muchos piratas intentaran borrar sus huellas utilizando programas conocidos como “Zapper's” o “Zap”, entre otros. Podemos detectar este tipo de programas para ello debemos buscar cuidadosamente archivos SETUID o SETGID (especialmente aquellos archivos SETUID de root). Esto lo podemos realizar con el comando “find”, escribimos lo siguiente:

- # find / -group kmen -perm -2000 –print
- # find / -user root -perm -4000 -print –xdev
- # ncheck -s /dev/rsd0g

Este ultimo comando “ncheck” nos permitirá buscar archivos SETUID por las particiones. Debemos buscar troyanos en nuestros archivos binarios, ya que suele ser una de las tareas principales de un pirata cuando ha comprometido la seguridad de un servidor. Una lista de posibles binarios que un pirata puede sustituir, es la siguiente:

- Login
- Su
- telnet
- netstat
- ifconfig
- ls
- find
- du

- df
- libc
- sync

Así como los binarios listados en /etc/inetd.conf. Hay varias utilidades ampliamente disponibles para detectar estos troyanos. Otras de las principales tareas de un pirata consiste en la utilización de *sniffers*, para capturar información confidencial. Los más usados son los siguientes:

- linsniff666.c
- esniff.c
- solsniff.c
- sunsniff.c
- sniffit

3.6.6 Detección de "problemas".

Desde que en 1980 James P. Anderson propusiera la detección de anomalías como un método válido para detectar intrusiones en sistemas informáticos [16], la línea de investigación más activa es la denominada Anomaly Detection IDS, IDSs basados en la detección de anomalías. La idea es a priori muy interesante, estos modelos de detección conocen lo que es "normal" en nuestra red o nuestras máquinas a lo largo del tiempo, desarrollando y actualizando conjuntos de patrones contra los que comparar los eventos que se producen en los sistemas. Si uno de esos eventos (por ejemplo, una trama procedente de una máquina desconocida) se sale del conjunto de normalidad, automáticamente se cataloga como sospechoso.

Los IDSs basados en detección de anomalías se basan en la premisa de que cualquier ataque o intento de ataque implica un uso anormal de los sistemas. Pero, ¿cómo puede un sistema conocer lo que es y lo que no es "normal" en nuestro entorno de trabajo? Para conseguirlo, existen dos grandes aproximaciones [17], o es el sistema el que es capaz de aprenderlo por sí mismo (basándose por ejemplo en el comportamiento de los usuarios, de sus procesos, del tráfico de nuestra red…) o bien se le especifica al sistema dicho comportamiento mediante un conjunto de reglas. La primera de estas aproximaciones utiliza básicamente métodos estadísticos (medias, varianzas…), aunque también existen modelos en los que se aplican algoritmos de aprendizaje automático, la segunda aproximación consiste en especificar mediante un conjunto de reglas los perfiles de comportamiento habitual basándose en determinados parámetros de los sistemas (con la dificultad añadida de decidir cuáles de esos parámetros que con mayor precisión delimitan los comportamientos intrusivos).

En el primero de los casos (el basado en métodos estadísticos), el detector observa las actividades de los elementos del sistema, activos (sujetos), pasivos (objetos) o ambos, y genera para cada uno de ellos un perfil que define su comportamiento, dicho perfil es almacenado en el sistema, y se actualiza con determinada frecuencia envejeciendo la información más antigua y priorizando la más fresca. El comportamiento del usuario en un determinado momento se guarda temporalmente en otro perfil, denominado "perfil actual" (current profile), y a intervalos regulares se compara con el almacenado previamente en busca de desviaciones que puedan indicar una anomalía.

Existen diferentes tipos de datos o medidas que pueden ser útiles en la elaboración de estos perfiles:

1. Intensidad de la actividad. Reflejan el ratio de progreso de la actividad en el sistema, para lo cual recogen datos a intervalos muy pequeños (típicamente entre un minuto y una hora). Estas medidas detectan ráfagas de comportamiento (por ejemplo, una excesiva generación de peticiones de entrada/salida en un cierto intervalo) que en espacios de tiempo más amplios no podrán ser detectadas.
2. Numéricas. Se trata de medidas de la actividad cuyo resultado se puede representar en forma de valor numérico, como el número de ficheros leídos por cierto usuario en una sesión o la cantidad de veces que ese usuario se ha equivocado al teclear su contraseña de acceso al sistema.
3. Categóricas. Las medidas categóricas son aquellas cuyo resultado es una categoría individual, y miden la frecuencia relativa o la distribución de una actividad determinada con respecto a otras actividades o categorías, por ejemplo, cual es la relación entre la frecuencia de acceso a un determinado directorio del sistema en comparación con la de acceso a otro. Seguramente la palabra "categoría" no es la más afortunada (por lo menos, no la más clara), ya que bajo este término se pueden englobar tanto a objetos (por ejemplo, ficheros) como a eventos (por ejemplo, llamadas a la función crypt()) del sistema, esta definición genérica puede resultar más sencilla si distinguimos entre categorías globales e individuales, en castellano plano, podemos entender las categorías globales como acciones muy genéricas dentro de un entorno, mientras que las categorías individuales serían la particularización para un elemento determinado del sistema. Así, una categoría global puede ser la formada por el conjunto de accesos remotos a la máquina, mientras que una individual sería la formada por los accesos desde una determinada ubicación física.
4. Distribución de registros de auditoria. Esta medida analiza la distribución de las actividades generadas en un pasado reciente basándose en los logs generados por las mismas, dicho análisis se realiza de forma ponderada, teniendo más peso las actividades más recientes, y es comparado con un perfil de actividades "habituales" previamente almacenado, de forma que permite detectar si en un pasado reciente se han generado eventos inusuales.

La segunda aproximación a la que antes hemos hecho referencia era la consistente en indicar mediante un conjunto de reglas el comportamiento habitual del sistema, suele ser denominada detección de anomalías basada en especificaciones (specification-based anomaly detection), y fue propuesta y desarrollada inicialmente por Calvin Cheuk Wang Ko y otros investigadores de la Universidad de California en Davis, durante la segunda mitad de los noventa. La idea en la que se sustentan los sistemas de detección de anomalías basados en especificaciones es que se puede describir el comportamiento "deseable" (entendiendo por "deseable" el comportamiento "normal") de cualquier programa cuya seguridad sea crítica, esta descripción se realiza en base a una especificación de seguridad mediante gramáticas, y se considera una violación de la seguridad (al menos en principio) a las ejecuciones de dichos programas que violen su respectiva especificación.

Para ver más claramente el concepto de la detección de anomalías basada en especificaciones, podemos pensar en la ejecución de un programa que se puede considerar crítico, por ser privilegiado: /bin/passwd. Si conseguimos diferenciar las diferentes ejecuciones de esta orden que se pueden considerar "habituales" (por ejemplo, cuando un usuario cambia su contraseña sin problemas, cuando se equivoca, cuando el sistema no le

deja cambiarla por los motivos típicos (es débil, hace poco que la cambió...), podríamos especificar formalmente cada una de estas secuencias de operación. De esta forma, cada vez que un usuario invoque a /bin/passwd, el sistema de detección monitorizará las operaciones que esa llamada genere, y considerará una intrusión a cualquiera que no sea "habitual".

La idea de los sistemas de detección de intrusos basados en la detección de anomalías es realmente atractiva, no obstante, existen numerosos problemas a los que estos mecanismos tratan de hacer frente. En primer lugar podemos pararnos a pensar en las dificultades que existen a la hora de "aprender" o simplemente especificar lo habitual, si alguien piensa que por ejemplo obtener un patrón de tráfico "normal" en una red es fácil, se equivoca, quizás establecer un conjunto de procesos habituales en una única máquina resulte menos complicado, pero tampoco se trata de una tarea trivial. Además, conforme aumentan las dimensiones de los sistemas (redes con un gran número de máquinas interconectadas, equipos con miles de usuarios…) estos se hacen cada vez más aleatorios e impredecibles.

Otro gran problema de los sistemas basados en detección de anomalías radica en la política de aprendizaje que éstos sigan [18], si se trata de esquemas donde el aprendizaje es rápido, un intruso puede generar eventos para conseguir un modelo distorsionado de lo "normal" antes de que el responsable de los sistemas se percate de ello, de forma que el IDS no llegue a detectar un ataque porque lo considera algo "habitual". Si por el contrario el aprendizaje es lento, el IDS considerará cualquier evento que se aleje mínimamente de sus patrones como algo anómalo, generando un gran número de falsos positivos (falsas alarmas), que a la larga harían que los responsables de los sistemas ignoren cualquier información proveniente del IDS, con los evidentes riesgos que esto implica.

3.6.7 Detección de usos indebidos.

Dentro de la clasificación de los sistemas de detección de intrusos en base a su forma de actuar, la segunda gran familia de modelos es la formada por los basados en la detección de usos indebidos. Este esquema se basa en especificar de una forma más o menos formal las potenciales intrusiones que amenazan a un sistema y simplemente esperar a que alguna de ellas ocurra, para conseguirlo existen cuatro grandes aproximaciones [19], los sistemas expertos, los análisis de transición entre estados, las reglas de comparación y emparejamiento de patrones (pattern matching) y la detección basada en modelos.

Los primeros sistemas de detección de usos indebidos, como NIDES, se basaban en los sistemas expertos para realizar su trabajo, en ellos las intrusiones se codifican como reglas de la base de conocimiento del sistema experto, de la forma genérica if-then (if condición then acción). Cada una de estas reglas puede detectar eventos únicos o secuencias de eventos que denotan una potencial intrusión, y se basan en el análisis de los registros de auditoria proporcionados por cualquier sistema Unix, esta es una de las principales ventajas de los sistemas expertos, el hecho de que el mecanismo de registro dentro de Unix venga proporcionado "de serie", ya que de esta forma el sistema de detección trabaja siempre por encima del espacio del sistema operativo, algo que facilita enormemente su integración dentro de diferentes clones de Unix.

Podemos pensar en un caso real que nos ayude a comprender el funcionamiento de los sistemas expertos a la hora de detectar intrusiones, el típico ejemplo es la detección de un mismo usuario conectado simultáneamente desde dos direcciones diferentes. Cada vez

que un usuario se autentica correctamente en el sistema, cualquier Unix genera una línea de registro que se guarda en el fichero de log correspondiente.

La segunda implementación de los sistemas de detección de usos indebidos era la basada en los análisis de transición entre estados, bajo este esquema, una intrusión se puede contemplar como una secuencia de eventos que conducen al atacante desde un conjunto de estados inicial a un estado determinado, representando este último una violación consumada de nuestra seguridad. Cada uno de esos estados no es más que una imagen de diferentes parámetros del sistema en un momento determinado, siendo el estado inicial el inmediatamente posterior al inicio de la intrusión, y el último de ellos el resultante de la completitud del ataque, la idea es que si conseguimos identificar los estados intermedios entre ambos, seremos capaces de detener la intrusión antes de que se haga efectiva.

Sin duda el sistema de detección basado en el análisis de transición entre estados más conocido es ustat [20]. Este modelo fue desarrollado inicialmente sobre SunOS 4.1.1 (en la actualidad está portado a Solaris 2), y utiliza los registros de auditoria generados por el C2 Basic Security Module de este operativo. En ustat, estos registros del c2-bsm son transformados a otros de la forma <s, a, o>, representando cada uno de ellos un evento de la forma “el sujeto S realiza la acción A sobre el objeto O”, a su vez, cada elemento de la terna anterior está formado por diferentes campos que permiten identificar unívocamente el evento representado. El sistema de detección utiliza además una base de datos (realmente, se trata de simples ficheros planos) formada principalmente por dos tablas, una donde se almacenan las descripciones de los diferentes estados (SDT, State Description Table) y otra en la que se almacenan las transiciones entre estados que denotan un potencial ataque (SAT, Signature Action Table). Cuando ustat registre una sucesión determinada de eventos que representen un ataque entrará en juego el motor de decisiones, que emprenderá la acción que se le haya especificado (desde un simple mensaje en consola informando de la situación hasta acciones de respuesta automática capaces de interferir en tiempo real con la intrusión).

La tercera implementación que habíamos comentado era la basada en el uso de reglas de comparación y emparejamiento de patrones o pattern matching [21], en ella, el detector se basa en la premisa de que el sistema llega a un estado comprometido cuando recibe como entrada el patrón de la intrusión, sin importar el estado en que se encuentre en ese momento. Dicho de otra forma, simplemente especificando patrones que denoten intentos de intrusión el sistema puede ser capaz de detectar los ataques que sufre, sin importar el estado inicial en que esté cuando se produzca dicha detección, lo cual suele representar una ventaja con respecto a otros modelos de los que hemos comentado.

Actualmente muchos de los sistemas de detección de intrusos más conocidos (por poner un ejemplo, podemos citar a snort o RealSecure) están basados en el pattern matching. Utilizando una base de datos de patrones que denotan ataques, estos programas se dedican a examinar todo el tráfico que ven en su segmento de red y a comparar ciertas propiedades de cada trama observada con las registradas en su base de datos como potenciales ataques, si alguna de las tramas empareja con un patrón sospechoso, automáticamente se genera una alarma en el registro del sistema.

Por último, tenemos que hablar de los sistemas de detección de intrusos basados en modelos, se trata de una aproximación conceptualmente muy similar a la basada en la transición entre estados, en el sentido que contempla los ataques como un conjunto de estados y objetivos, pero ahora se representa a los mismos como escenarios en lugar de hacerlo como transiciones entre estados. En este caso se combina la detección de usos

indebidos con una deducción o un razonamiento que concluye la existencia o inexistencia de una intrusión, para ello, el sistema utiliza una base de datos de escenarios de ataques, cada uno de los cuales está formado por una secuencia de eventos que conforman el ataque. En cada momento existe un subconjunto de esos escenarios, denominado de escenarios activos, que representa los ataques que se pueden estar presentando en el entorno, un proceso denominado anticipador analiza los registros de auditoría generados por el sistema y obtiene los eventos a verificar en dichos registros para determinar si la intrusión se está o no produciendo (realmente, al ser esos registros dependientes de cada sistema Unix, el anticipador pasa su información a otro proceso denominado planner, que los traduce al formato de auditoría utilizado en cada sistema). El anticipador también actualiza constantemente el conjunto de escenarios activos, de manera que este estará siempre formado por los escenarios que representan ataques posibles en un determinado momento y no por la base de datos completa.

Hasta hace poco tiempo no existían sistemas de detección de intrusos basados en modelos funcionando en sistemas reales, por lo que era difícil determinar aspectos como su eficiencia a la hora de detectar ataques. En 1998 se presentó nstat, una extensión de ustat (del cual hemos hablado antes) a entornos distribuidos y redes de computadores, y en el que se combina el análisis de transición entre estados con la detección de intrusos basada en modelos. A pesar de todo, este modelo es el menos utilizado a la hora de detectar ataques y efectuar respuestas automáticas ante los mismos, especialmente si lo comparamos con los basados en la comparación y emparejamiento de patrones.

Los IDSs basados en la detección de usos indebidos son en principio más robustos que los basados en la detección de anomalías, al conocer la forma de los ataques, es teóricamente extraño que generen falsos positivos (a no ser que se trate de un evento autorizado pero muy similar al patrón de un ataque), es necesario recalcar el matiz "teóricamente", porque como veremos más adelante, la generación de falsos positivos es un problema a la hora de implantar cualquier sistema de detección. No obstante, en este mismo hecho radica su debilidad, sólo son capaces de detectar lo que conocen, de forma que si alguien nos lanza un ataque desconocido para el IDS éste no nos notificará ningún problema, como ya dijimos, es algo similar a los programas antivirus, y de igual manera que cada cierto tiempo es conveniente (en MS-DOS y derivados) actualizar la versión del antivirus usado, también es conveniente mantener al día la base de datos de los IDSes basados en detección de usos indebidos. Aún así, seremos vulnerables a nuevos ataques.

Otro grave problema de los IDSes basados en la detección de usos indebidos es la incapacidad para detectar patrones de ataque convenientemente camuflados. Volviendo al ejemplo de los antivirus, pensemos en un antivirus que base su funcionamiento en la búsqueda de cadenas virales, lo que básicamente hará ese programa será buscar cadenas de código hexadecimal pertenecientes a determinados virus en cada uno de los archivos a analizar, de forma que si encuentra alguna de esas cadenas el software asumirá que el fichero está contaminado. Y de la misma forma que un virus puede ocultar su presencia simplemente cifrando esas cadenas (por ejemplo de forma semialeatoria utilizando eventos del sistema, como el reloj), un atacante puede evitar al sistema de detección de intrusos sin más que insertar espacios en blanco o rotaciones de bits en ciertos patrones del ataque, aunque algunos IDSes son capaces de identificar estas transformaciones en un patrón, otros muchos no lo hacen.

AUTENTIFICACIÓN Y CERTIFICADOS DIGITALES

4.1 Introducción

En la actualidad, y especialmente para el tráfico que atraviesa redes inseguras, como Internet, es normal que se busque asegurar, más aún que la privacidad, la autentificación, la integridad y otras propiedades como el no repudio.

En este capítulo se empezará por analizar los problemas típicos que se plantean al intentar imponer tales propiedades. Se continuará con la firma digital, explicando la importancia que tiene, hoy en día, para casi cualquier sistema de autentificación segura. Se analizarán distintos esquemas de firma digital, haciendo énfasis especial en el más utilizado, que es el basado en el algoritmo de criptografía asimétrica RSA[1]. El cual, es un algoritmo de clave pública entre los participantes, siendo necesario estándares como el X.509 donde cada participante tiene el suyo propio que certifica su identidad, es decir, su clave pública, pero para garantizar que este certificado es válido aparecen lo que llamamos Autoridades de certificación.

De una u otra forma, las estructuras y algoritmos de este capítulo se están usando, y modificándose parcialmente todos los días, se utilizan en los sistemas criptográficos utilizados en redes de ordenadores, tanto en el caso de seguridad aplicada a redes internas de las organizaciones, como al tráfico comercial por redes inseguras, especialmente Internet.

4.2 Autentificación.

La autentificación es cualquier proceso mediante el cual se verifica que alguien es quien dice ser [22], se le permite estar donde quiere ir, o tener la información que quiere tener. Este concepto tiene diversas aceptaciones dependiendo de dónde y cómo se aplique, como por ejemplo:

- Un dispositivo de red, por ejemplo un encaminador, demostrando a otro encaminador en Internet, que realmente su identidad es la que dice ser.
- Un servidor de comercio electrónico, que usa SSL, demostrando a un posible cliente, mediante lo que se conoce como “conexión segura”, que realmente es el servidor que dice ser.
- Un servidor de oficina de banca por Internet, que necesita asegurarse de que un mensaje de cargo a una cuenta corriente es de quien dice ser. El mensaje suele venir cifrado con una contraseña, que puede estar autentificada.

A parte de la autentificación podemos hablar de integridad, que es la propiedad que permite garantizar que una serie de datos no han sido modificados, pero hay otra propiedad que, cada vez más, resulta importante para la seguridad, especialmente en entornos comerciales, se trata del no repudio y es una propiedad intrínsecamente ligada con la autentificación. El no repudio es la necesidad de asociar un mensaje con su creador.

[1]RSA: Rivest Shamir Addleman. Nombre de los tres inventores del algoritmo RSA, de criptografía asimétrica. También, nombre de la empresa que crearon ellos mismos para el desarrollo de todo tipo de productos y sistemas criptográficos.

En la vida no digital se han usado toda una variedad de métodos para reducir la capacidad de la gente de negar que hayan sido los creadores de documentos de papel. Tales métodos y los aplicados a sus homólogos digitales son bastante parecidos y están muy unidos a los problemas de autenticación.

En ambos casos, se aplica una firma al mensaje. En el caso de firmas en papel, se trabaja con una autoridad de confianza, por ejemplo un notario, para validar los mensajes. En el campo digital es típico echar mano de una autoridad de certificación, aunque no es la única solución. En estos casos tal autoridad de certificación emite un documento denominado certificado digital.

Para conseguir todas estas propiedades se utilizan varias combinaciones de algoritmos criptográficos asimétricos y funciones de una sola vía y casi todas ellas pasan por la utilización de firmas digitales.

En una solución en el cual no se usa la firma digital, sino que se basa en criptografía simétrica, se trata de introducir la figura de una tercera figura, a la que podemos llamar Fiador, que mantiene una clave privada, dentro de un sistema de criptografía simétrica, para cada participante del sistema. Así, si Alicia y Juanma son dos de tales participantes, el Fiador mantiene una clave, ka, para la comunicación segura con Alicia y una clave, kb, para la comunicación segura con Juanma.

En el caso de que Alicia desee enviar un mensaje a Juanma los pasos que se siguen son los siguientes:

- Alicia encripta el mensaje M utilizando la clave ka, obteniendo C(ka(M)), y lo envía al Fiador.
- El fiador comprueba la integridad de Alicia, desencripta el mensaje, obteniendo M y, encriptándolo con kb, envía a Juanma tanto el mensaje M, la identidad de Alicia y la firma C(ka(M)), C(kb(M, A, C(ka(M)))).

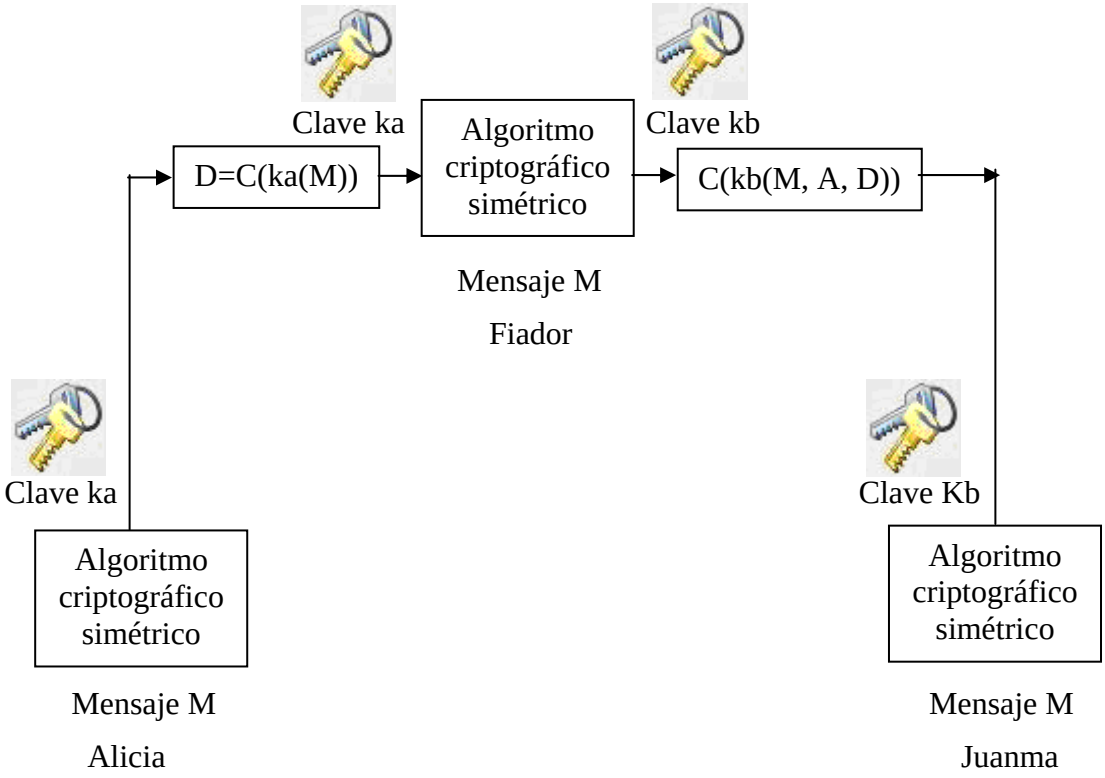

Figura 14: Autentificación y no repudio mediante clave privada

Como tanto Alicia como Juanma confían en el Fiador, el procedimiento es seguir y, en caso de dudas, el Fiador puede comprobar la identidad de Alicia, descifrando el mensaje recibido de ella. En este caso, la propia cifra de datos está haciendo el papel de autentificador.

En el caso de que la clave del algoritmo simétrico sea segura, se puede afirmar que, además de la privacidad que se consigue con tal cifra, se obtiene a la vez la integridad y autenticidad del mensaje, ya que es únicamente el usuario emisor el puede crear ese mensaje.

Con este método no se puede alcanzar la doble autentificación de mensaje y emisor, se sigue teniendo la inseguridad de la transmisión de claves.

Otra aproximación a este problema es hacer que el Fiador sea un KDC, *Key Distribuition Center*, una especie de distribuidor de claves de confianza, que, además, debe compartir una clave, que suele denominarse maestra, con todos los participantes de este sistema.

La tarea del KDC es distribuir una clave de sesión, que será utilizada, para la conexión entre dos participantes. Tal clave de sesión está protegida, siguiendo el modelo, por la clave maestra citada. El sistema kerberos se basa en este modelo, servicio de autentificación, auditoria y contabilidad, desarrollado en el MIT (Massachussets Institute of Technology).

4.3 Autentificación HTTP basada en php.

Las características de autentificación HTTP en PHP solo están disponibles cuando se está ejecutando como módulo en Apache. De esta manera se puede usar la función header() para enviar un mensaje de “Autentificación requerida” al navegador cliente haciendo que muestre una ventana de entrada emergente con nombre de usuario y contraseña. Una vez que el usuario ha rellenado el nombre y la contraseña, la URL que contiene el script PHP vuelve a ser llamada con las variables $PHP_AUTH_USER, $PHP_AUTH_PW y $PHP_AUTH_TYPE rellenas con el nombre de usuario, la contraseña y el tipo de autentificación respectivamente. Se puede configurar Apache de forma que para acceder a cierto directorio (y subdirectorios) sea necesario introducir un usuario y una clave. Dentro de esta posibilidad vamos a ver dos métodos:

- Basic: cuando en el cliente se introduce el usuario y la clave, estos viajan al servidor sin cifrar.
- Digest: cuando el usuario y la clave van cifrados del cliente al servidor.

Estos métodos tienen la ventaja de que es Apache quien se encarga de comprobar las correctas credenciales del cliente, por lo que no tenemos que hacer ningún tipo de comprobación en nuestra aplicación.

Hay que tener en cuenta que estos dos métodos sólo sirven para autenticar a un usuario cuando intenta acceder a un determinado recurso. Es decir, Apache identifica si se trata de un usuario válido, y en tal caso le deja acceder al recurso. Pero los datos que posteriormente se envíen de cliente al servidor, o viceversa, no tienen ningún tipo de cifrado. Estos métodos sólo sirven para controlar el acceso, no para proteger los datos una vez se ha comprobado que el acceso es válido.

4.3.1 Autentificación HTTP Basic.

Para este modo de autenticación se utiliza el módulo mod_auth. Consiste en crear un fichero de texto en el que se almacenará un nombre de usuario y una contraseña, una vez creado se configura el servidor para que pida autentificación cuando se acceda a un determinado recurso.

Este método tiene la ventaja de que lo soportan todos los navegadores, pero tiene la desventaja de que el nombre de usuario y la clave no van cifrados del cliente al servidor. Esto hace que no sea un método recomendado para entornos donde la seguridad es clave.

4.3.1.1 Configuración de httpd.conf

En el fichero /usr/local/apache2/conf/httpd.conf tenemos que añadir las siguientes líneas:

```
<Directory "/usr/local/apache2/htdocs/protegido">
    AuthName "privatefiles"
    AuthType Basic
    AuthUserFile /usr/local/apache2/conf/passwd_basic
    Require valid-user
</Directory>
```

El significado de cada una de estas directivas es:

- Directory: estamos diciendo a Apache que las directivas que vienen a continuación tiene efecto sobre el directorio /usr/local/apache2/htdocs/private, y sus subdirectorios. Es decir, que vamos a proteger este directorio y sus subdirectorios.
- AuthName: es el nombre del dominio de autenticación. También es el texto que aparecerá en la ventana que pide el usuario y la clave.
- AuthType: el tipo de autenticación.
- AuthUserFile: el fichero donde están los usuarios y las claves.
- Require: con esta directiva indicamos que usuarios tienen acceso, tenemos varias posibilidades:
    - valid-user: cualquier usuario que esté en el fichero de claves.
    - user <lista de usuarios>: lista de usuarios, separados por espacios, que pueden acceder.
    - group <lista de grupos>: lista de grupos, separados por espacios, que pueden acceder. Ojo, porque si usamos esta opción también necesitamos usar la directiva AuthGroupFile para indicar donde se encuentra el fichero con la definición de los grupos.

Utilizando la configuración anterior, al acceder al directorio *private* el mensaje que nos muestra es el siguiente:

Figura 15: Autentificación Basic.

4.3.1.2 Creación de un usuario.

Para crear los usuarios para el método de autenticación “Basic”, se usa la aplicación htpasswd:

- `cd /usr/local/apache2/bin`
- `./htpasswd /usr/local/apache2/conf/passwd_basic admin`

De esta forma se añade el usuario “admin.” al fichero de claves /usr/local/apache2/conf/passwd_basic. El programa pedirá la clave y luego la vuelve a preguntar para confirmarla.

Si el fichero no existe (para la primera vez), es necesario lanzar el comando con la opción -c:

```
./htpasswd -c /usr/local/apache2/conf/passwd_basic admin
```

También se puede especificar la clave en la línea de comandos, a continuación del nombre. En este caso se tendrá que utilizar la opción -b:

```
./htpasswd -b /usr/local/apache2/conf/passwd_basic admin laclave
```

Si se ejecuta el comando sin argumentos, obtendremos la ayuda de las posibles opciones.

4.3.1.3 Creación de un grupo.

Para especificar los grupos se debe crear un fichero de texto con el siguiente formato para cada línea:

- `nombreGrupo: user1 user2 user3 ...`

Seguidamente se usa la directiva AuthGroupFile para indicar la ruta completa donde se encuentra el fichero que se ha creado con la definición de los grupos:

```
<Directory "/usr/local/apache2/htdocs/private">
```

```
    AuthName "privatefiles"
    AuthType Basic
    AuthUserFile /usr/local/apache2/conf/passwd_basic
    AuthGroupFile /usr/local/apache2/conf/ficherogrupos
    Require group nombregrupo
</Directory>
```

Cada usuario del grupo se añadirá al fichero de claves tal y como se describió en el apartado anterior.

4.3.2 Autentificación HTTP Digest.

Para este modo de autenticación se utiliza el módulo mod_auth_digest. Este método tiene la gran ventaja de que el usuario y la clave van cifradas del cliente al servidor. Pero tiene el inconveniente de que se puede encontrar con versiones antiguas de navegadores, que no lo soporten.

4.3.2.1 Configuración de httpd.conf

En el fichero /usr/local/apache2/conf/httpd.conf basta con añadir las siguientes líneas:

```
<Directory "/usr/local/apache2/htdocs/protegido">
    AuthName "privatefiles"
    AuthType Digest
    AuthDigestFile /usr/local/apache2/conf/passwd_digest
    Require valid-user
</Directory>
```

Se puede apreciar que es prácticamente igual que cuando se usa el método Basic. Ha cambiado:

AuthName: esta vez se indica que el método a usar es “Digest”.

AuthDigestFile: esta directiva sustituye a AuthUserFile, pero tiene la misma finalidad: especificar donde se encuentra el fichero de claves. Hay que tener presente que el fichero de claves tiene que ser distinto que el que se usa con el método Basic, ya que tiene formatos diferentes.

4.3.2.2 Creación de un usuario.

Para crear los usuarios para el método de autenticación “Digest”, se usa la aplicación htdigest. Por ejemplo:

- `cd /usr/local/apache2/bin`
- `./htdigest /usr/local/apache2/conf/passwd_digest privatefiles admin`

De esta forma se añade el usuario “admin.” al fichero de claves /usr/local/apache2/conf/passwd_digest. El programa pedirá la clave y luego la volverá a preguntar para confirmarla.

Se puede ver que en la línea de comandos se ha introducido como segundo parámetro “privatefiles”. Este parámetro debe coincidir exactamente con el nombre del dominio de

autenticación que se ha especificado con la directiva AuthName. Es decir, cuando se añade un usuario, se hace a un dominio de autenticación concreto. De forma que si con la directiva *Require* se especifica "valid-user", se consideran sólo usuarios válidos los que pertenecen al dominio de autenticación especificado en AuthName.

Si el fichero no existe (para la primera vez), es necesario lanzar el comando con la opción -c:

- `./htpasswd -c /usr/local/apache2/conf/passwd_digest privatefiles admin`

Si se ejecuta el comando sin argumentos, se obtendrá la ayuda de las posibles opciones.

4.4 Firma digital.

Dos matemáticos de la Universidad de Standford y otros estudiosos del Instituto Tecnológico de Massachussets, descubrieron que aplicando conceptos matemáticos era posible autentificar la información digital. A este conjunto de fórmulas se le denominó "Criptografía de Llave Pública" y con ello cobraban especial importancia términos como la confidencialidad, referente a la capacidad de mantener accesible un documento electrónico a determinadas personas, y la autenticidad, que determina el compromiso de un individuo sobre el contenido del documento electrónico.

Ambos conceptos, por tanto, se solucionaban mediante una firma escrita en un documento tradicional. Sin embargo, con la aparición del documento electrónico, se hacía necesario garantizar la confidencialidad y la autenticidad mediante técnicas de "firma digital".

La firma digital es una herramienta tecnológica que permite garantizar la autoría e integridad de los documentos digitales, posibilitando que éstos gocen de una característica que únicamente era propia de los documentos en papel, no implica asegurar la confidencialidad del mensaje, un documento firmado digitalmente puede ser visualizado por otras personas, al igual que cuando se firma de forma escrita. Posee características técnicas y normativas, esto significa que existen procedimientos técnicos que permiten la creación y verificación de firmas digitales, y existen documentos normativos que respaldan el valor legal.

Desde el punto de vista criptográfico, los requisitos que debe cumplir la firma digital son los siguientes:

- Debe ser fácil de generar.
- Debería ser irrevocable, no rechazable por el firmante.
- Debería de ser única, sólo el emisor debería ser capaz de generarla.
- Debe ser fácil de reconocer por cualquier receptor de un mensaje.
- Debe depender del mensaje y del emisor.

La firma digital está formada por una serie de caracteres elaborados por un sistema informático, que se basa en algoritmos matemáticos. La firma digital emplea un método de encriptación que adjudica una clave pública y otra privada a cada sujeto.

La clave pública es conocida por los usuarios de cada operación. Para firmar electrónicamente un documento, el software del firmante aplica un algoritmo llamado hash sobre el texto. Así, se obtiene un extracto específico para ese mensaje. El extracto

conseguido se cifra por medio de la clave privada del autor. Y se obtiene el mensaje final cifrado con la firma digital.

El receptor del mensaje sólo tiene que aplicar la clave pública para descifrar el mensaje.

4.4.1 Funcionamiento de la firma digital.

El fundamento de las firmas electrónicas es la criptografía, disciplina matemática que no sólo se encarga del cifrado de textos para lograr su confidencialidad, protegiéndolos de ojos indiscretos, sino que también proporciona mecanismos para asegurar la integridad de los datos y la identidad de los participantes en una transacción.

El cifrado consiste en transformar un texto en claro (inteligible por todos) mediante un algoritmo en un texto cifrado, gracias a una información secreta o clave de cifrado, que resulta ininteligible para todos excepto el legítimo destinatario del mismo. Se distinguen dos métodos generales de cifrado:

- Cifrado simétrico: cuando se emplea la misma clave en las operaciones de cifrado y descifrado, se dice que el sistema es simétrico o de clave secreta. Estos sistemas son mucho más rápidos que los de clave pública, y resultan apropiados para el cifrado de grandes volúmenes de datos. Ésta es la opción utilizada para cifrar el cuerpo de los mensajes en el correo electrónico o los datos intercambiados en las comunicaciones digitales. Para ello se emplean algoritmos como IDEA, RC5, DES, TRIPLE DES, etc.
- Cifrado asimétrico: cuando se utiliza una pareja de claves para separar los procesos de cifrado y descifrado, se dice que el sistema es asimétrico o de clave pública. Una clave, la privada, se mantiene secreta, mientras que la segunda clave, la pública, es conocida por todos. De forma general, las claves públicas se utilizan para cifrar y las privadas, para descifrar. El sistema posee la propiedad de que a partir del conocimiento de la clave pública no es posible determinar la clave privada ni descifrar el texto con ella cifrado. Los sistemas de clave pública, aunque más lentos que los simétricos, resultan adecuados para los servicios de autenticación, distribución de claves de sesión y firmas digitales. Se utilizan los algoritmos de RSA, Diffie-Hellman, etc.

En general, el cifrado asimétrico se emplea para cifrar las claves de sesión utilizadas en el documento, de modo que puedan ser transmitidas sin peligro a través de la red junto con el documento cifrado, para que en recepción éste pueda ser descifrado. La clave de sesión se cifra con la clave pública del destinatario del mensaje, que aparecerá normalmente en una libreta de claves públicas. El cifrado asimétrico se emplea también para firmar documentos y autentificar entidades. En principio, bastaría con cifrar un documento con la clave privada para obtener una firma digital segura, puesto que nadie excepto el poseedor de la clave privada puede hacerlo. Posteriormente, cualquier persona podría descifrarlo con su clave pública, demostrándose así la identidad del firmante.

4.4.2 Autoridades de certificación.

- Fábrica Nacional de Moneda y Timbre [23] La FNMT se constituye como la autoridad publica de Certificación Española. Sus certificados son utilizados para la relación con la administración pública. La Agencia Tributaria es un importante exponente de la utilidad del certificado emitido por la FNMT: la declaración de la renta es posible presentarla telemáticamente gracias a esta iniciativa. El número de

administraciones que utilizan los certificados de la Fábrica para las relaciones con sus administrados aumentan permanentemente [24].

- ACE (Agencia de Certificación electrónica) [25]. ACE centra su actividad en proporcionar servicios de confianza que garanticen la seguridad de las transacciones electrónicas, con los subsiguientes ahorros de costes, disminución de tiempo y de recursos utilizados en tales gestiones. Su orientación se dirige al sector empresarial.
- FESTE [26]. El Patronato de FESTE está formado por el Consejo General del Notariado, el Consejo General de la Abogacía y la Universidad de Zaragoza. Los distintos certificados emitidos por FESTE van desde los certificados notariales, a los certificados WEB o los certificados corporativos para su uso por instituciones privadas.
- IPSCA [27]. Centra sus actividades en los servicios de Certificación Digital en los que se incluye la emisión de certificados (Servidor, WAP, Firma de Código,...), formación, soporte y consultoría. Siendo su área de actuación más destacada, la emisión de Certificados de Servidor Seguro para comercio electrónico, actuando como una tercera parte que verifica, autentica y certifica identidades sobre Internet. Por medio de la promoción anunciada por Microsoft, ipsCA pone gratuitamente a disposición de los usuarios de Office XP un certificado personal del tipo B1 para su utilización en el entorno de firma proporcionado por MS Office XP. El objetivo de Microsoft con esta iniciativa es facilitar el acceso a este tipo de tecnologías, acercándolas al usuario final.
- CAMERFIRMA. Camerfirma es la autoridad de certificación digital de las Cámaras de Comercio españolas además de tener experiencia en servicios de outsourcing de Entidades de Certificación. Actualmente, Camerfirma está participado por el Consejo Superior de Cámaras, y por tres importantes entidades financieras como son Banesto, Bancaja o Caixa Galicia.

La autoridad de certificación más importante en España es la Casa de la Moneda.

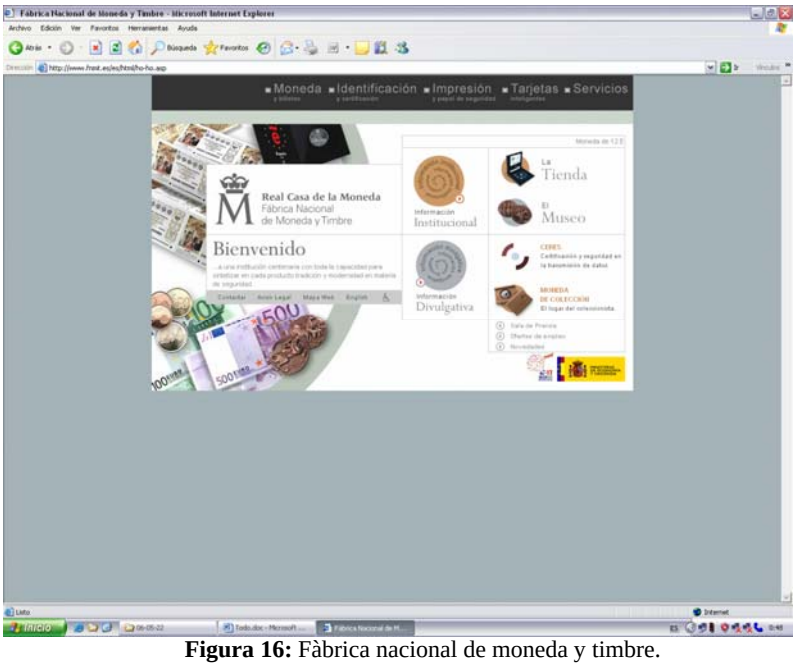

Figura 16: Fàbrica nacional de moneda y timbre.

4.5 Certificados digitales.

Los Certificados electrónicos son documentos digitales que sirven para asegurar la veracidad de la Clave Pública perteneciente al propietario del certificado ó de la entidad [28], con la que se firman digitalmente documentos que puedan proporcionar las más absolutas garantías de seguridad respecto a cuatro elementos fundamentales:

- La autentificación del usuario/entidad (es quien asegura ser).
- La confidencialidad del mensaje (que sólo lo podrá leer el destinatario).
- La integridad del documento (nadie los ha modificado).
- El no repudio (el mensaje una vez aceptado, no puede ser rechazado por el emisor).

Es, por tanto, muy importante estar realmente seguros de que la Clave Pública que manejamos para verificar una firma o cifrar un texto, pertenece realmente a quien creemos que pertenece.

Sería nefasto cifrar un texto confidencial con una Clave Pública de alguien, que no es nuestro intencionado receptor. Si se hiciera, la persona a quién pertenece la clave pública con la que se ha cifrado, podría conocer perfectamente el contenido de este, si tuviera acceso al texto cifrado. De la misma forma, si se manejara una clave pública de alguien que se hace pasar por otro, sin poderlo detectar, podríamos tomar una firma fraudulenta por válida y creer que ha sido realizada por alguien que realmente no es quien dice ser.

Otro dato a tener en cuenta, es que un certificado no puede falsificarse ya que van firmados por la Autoridad de Certificación. Si algún dato se modificase la firma no correspondería con el resumen (hash) que se obtendría de los datos modificados. Por tanto al utilizarlo, el software que los gestiona daría un menaje de invalidez.

Un certificado electrónico contiene una clave pública, y una firma digital**.** Para su correcto funcionamiento, los certificados contienen además la siguiente información:

- Un identificador del propietario del certificado, que consta de su nombre, sus apellidos, su dirección e-mail, datos de su empresa como el nombre de la organización, departamento, localidad, provincia y país, etc.
- Otro identificador de quién asegura su validez, que será una Autoridad de Certificación.
- Dos fechas, una de inicio y otra de fin del período de validez del certificado, es decir, cuándo un certificado empieza a ser válido y cuándo deja de serlo, fecha a partir de la cual la clave pública que se incluye en él, no debe utilizarse para cifrar o firmar.
- Un identificador del certificado o número de serie, que será único para cada certificado emitido por una misma Autoridad de Certificación. Esto es, identificará inequívocamente a un certificado frente a todos los certificados de esa Autoridad de Certificación.
- Firma de la Autoridad de Certificación de todos los campos del certificado que asegura la autenticidad del mismo. Los navegadores actuales gestionan y almacenan las Claves Públicas de los certificados que permiten al emisor de mensajes firmarlos y encriptarlos utilizando las claves públicas de los destinatarios. Para estar completamente seguros en cualquier transacción es necesario utilizar, al menos dos tipos de certificados, uno general para comunicaciones seguras (X.509) y otro específico para transacciones económicas (SET). Además de servir como

mecanismo confiable y seguro de identificación en la red, su certificado de identidad digital le permite disfrutar de otra serie de beneficios, puede enviar y recibir información confidencial, asegurándose que sólo el remitente pueda leer el mensaje enviado, puede acceder a sitios Web de manera segura con su identidad digital, sin tener que usar el peligroso mecanismo de passwords; puede firmar digitalmente documentos, garantizando la integridad del contenido y autoría del documento, y todas aquellas aplicaciones en que se necesiten mecanismos seguros para garantizar la identidad de las partes y confidencialidad e integridad de la información intercambiada, como comercio electrónico, declaración de impuestos, pagos provisionales, uso en la banca, etc.

- Aplicaciones de Internet como navegadores o programas para correo electrónico, ya traen incorporados los elementos que les permiten utilizar los certificados de identidad digital, por lo que los usuarios no necesitan instalar ningún software adicional.

La *Autoridad de Certificación (CA)*, es quien firma digitalmente los certificados, asegurando su integridad y certificando la relación existente entre la Clave Pública contenida y la identidad del propietario. La firma de la CA es la que garantiza la validez de los certificados.

La confianza de los usuarios la Autoridad de Certificación es fundamental para el buen funcionamiento del servicio. El entorno de seguridad (control de acceso, cifrado, etc.) de la CA ha de ser muy fuerte, en particular en lo que respecta a la protección de la Clave Privada que utiliza para firmar sus emisiones. Si este secreto se viera comprometido, toda la infraestructura de Clave Pública (PKI) se vendría abajo.

Las autoridades de certificación pueden realizan las siguientes tareas:

- Emisión de los certificados de usuarios registrados y validados por la Autoridad de Registro (RA).
- Revocación de los certificados que ya no sean válidos (CRL - lista de certificados revocados). Un certificado puede ser revocado por que los datos han dejado de ser válidos, la clave privada ha sido comprometida o el certificado ha dejado de tener validez dentro del contexto para el que había sido emitido.
- Renovación de certificados.
- Publicar certificados en el directorio repositorio de certificados.

La emisión de certificados y la creación de claves privadas para firmas digitales acostumbra a depender de una pluralidad de entidades que están jerarquizadas de una manera que las de nivel inferior obtienen su capacidad de certificación de otras entidades de nivel superior. Finalmente, en la cúspide de la pirámide suele hallarse una autoridad certificadora, que puede pertenecer al Estado.

Los certificados indican la autoridad certificadora que lo ha emitido, identifican al firmante del mensaje o transacción, contienen la clave pública del firmante, y contienen a su vez la firma digital de la autoridad certificadora que lo ha emitido.

De esta manera, las partes que intervienen en una transacción aportan como credencial los certificados de su correspondiente entidad certificadora. Para llegar a ser una entidad certificadora deberá mediar una solicitud a una autoridad certificadora de nivel superior, que podrá denegar la licencia si el solicitante no ofrece la fiabilidad o los conocimientos necesarios, ni cumple los requisitos establecidos en la ley.

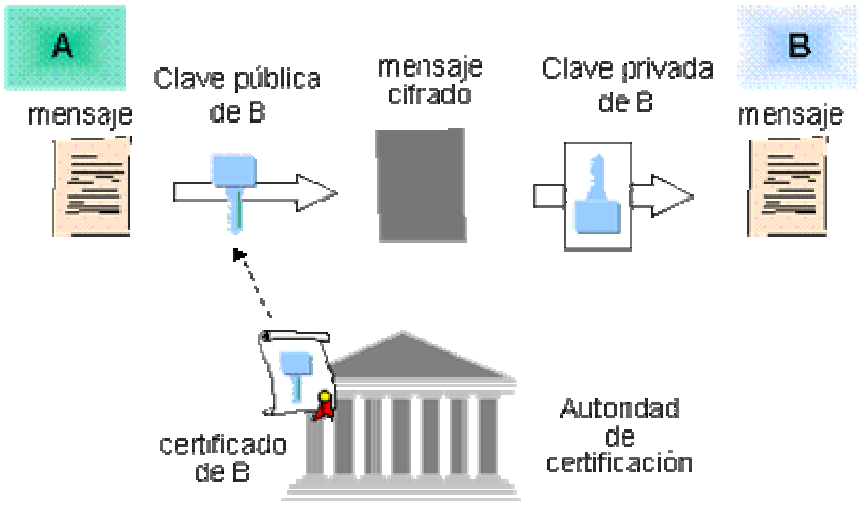

Figura 17: Certificados digitales.

INTRODUCCIÓN A LOS SERVIDORES Web: EL CASO APACHE.

5.1 Servidores Web.

Un servidor Web es un programa que implementa el *protocolo HTTP* (hypertext transfer protocol). Este protocolo está diseñado para transferir lo que llamamos hipertextos, páginas Web o páginas HTML (hypertext markup language): textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de sonidos [29].

Sin embargo, el hecho de que HTTP y HTML estén íntimamente ligados no debe dar lugar a confundir ambos términos. HTML es un formato de archivo y HTTP es un protocolo.

Cabe destacar el hecho de que la palabra servidor identifica tanto al programa como a la máquina en la que dicho programa se ejecuta. Existe, por tanto, cierta ambigüedad en el término, aunque no será difícil diferenciar a cuál de los dos nos referimos en cada caso.

Un servidor Web se encarga de mantenerse a la espera de peticiones HTTP llevada a cabo por un cliente HTTP que solemos conocer como navegador. El navegador realiza una petición al servidor y éste le responde con el contenido que el cliente solicita. A modo de ejemplo, al teclear http://juanmabultaco.blogspot.com en nuestro navegador, éste realiza una petición HTTP al servidor de dicha dirección. El servidor responde al cliente enviando el código HTML de la página, el cliente, una vez recibido el código, lo interpreta y lo muestra en pantalla. Como vemos con este ejemplo, el cliente es el encargado de interpretar el código HTML, es decir, de mostrar las fuentes, los colores y la disposición de los textos y objetos de la página, el servidor tan sólo se limita a transferir el código de la página sin llevar a cabo ninguna interpretación de la misma.

Sobre el servicio Web clásico podemos disponer de aplicaciones Web. Éstas son fragmentos de código que se ejecutan cuando se realizan ciertas peticiones o respuestas HTTP. Hay que distinguir entre:

- Aplicaciones en el lado del cliente: el cliente Web es el encargado de ejecutarlas en la máquina del usuario. Son las aplicaciones tipo Java o Javascript, el servidor proporciona el código de las aplicaciones al cliente y éste, mediante el navegador, las ejecuta. Es necesario, por tanto, que el cliente disponga de un navegador con capacidad para ejecutar aplicaciones (también llamadas scripts).
- Aplicaciones en el lado del servidor: el servidor Web ejecuta la aplicación, ésta, una vez ejecutada, genera cierto código HTML, el servidor toma este código recién creado y lo envía al cliente por medio del protocolo HTTP.

Las aplicaciones de servidor suelen ser la opción por la que se opta en la mayoría de las ocasiones para realizar aplicaciones Web. La razón es que, al ejecutarse ésta en el servidor y no en la máquina del cliente, éste no necesita ninguna capacidad adicional, como sí ocurre en el caso de querer ejecutar aplicaciones javascript o java. Así pues, cualquier cliente dotado de un navegador Web básico puede utilizar este tipo de aplicaciones.

Este proyecto en un principio se empezó realizando bajo Windows pero a medida que se fueron viendo temas sobre seguridad, criptografía… se apreció que lo más idóneo

para este proyecto era trabajar bajo Linux, por eso se cambió la base de datos, apache... Entre las características más relevantes de este cambio cabe destacar que en la actualidad la mayor parte de los usuarios de un computador utilizan el sistema operativo Windows en cualquiera de sus versiones como base, esto sucede en realidad porque la mayoría de las máquinas que compramos vienen con este tipo de software, sin embargo dependiendo del área en el que se desempeñe el profesional serán las necesidades de seguridad, desempeño y herramientas que debe tener, por ejemplo para un diseñador gráfico será mejor utilizar un ambiente basado en *mac* y para una secretaria que solamente utiliza la máquina como procesador de texto puede tener única y exclusivamente un sistema basado incluso en *Windows 3.1 o windows9x,* sin embargo para un desarrollador de software o alguna empresa que busque seguridad en su red será mejor tener alguna distribución de *Unix* , como puede ser *Linux, RedHat, PPP, Mandrake* , hace algunos años eran muy pocos los usuarios de otras plataformas distintas a Windows, y aunque la mayor parte todavía utiliza como *default* Windows, cada vez son más los usuarios y empresas que migran a otras plataformas.

Los sistemas operativos se definen como el conjunto de procedimientos que permiten el control de los recursos de una instalación para el procesamiento de datos. Dichos recursos comprenden el equipo, los programas, los datos y el o los operadores. Sin embargo, por lo general se define un sistema operativo como a un programa que supervisa la operación de otros programas. Estos programas son de tres tipos básicos: control de tareas, sistema de control de entrada-salida y programa de procesamiento.

Linux esta basado en Software Libre (Free Software) y, más recientemente Software de fuentes abiertas (Open Source Software), es decir, en vez de que el código de el sistema o de cada uno de los programas sea un secreto celosamente guardado por la empresa que lo produce, éste es puesto a disposición del público, para que puedan modificar, mejorar o corregir los programas, logrando un tiempo de desarrollo, respuesta a incidentes y soporte que ninguna compañía comercial ha logrado igualar. Bajo este esquema están desarrollados tanto Linux como la mayor parte de los programas que con él podemos correr, así como otros sistemas operativos.

La siguiente tabla nos muestra una comparativa de la seguridad entre Linux y Windows [30]:

LINUX	**WINDOWS**
Seguridad en los archivos: Cada archivo tiene definida la seguridad para Dueño, Grupo y Otros. Cada uno de ellos tiene permisos de lectura, escritura y ejecución (*rwxrwxrwx*). Sabiendo manejar este sistema, presenta una gran flexibilidad. Además de esto, tiene la característica del *SUID bit*, que permite que un archivo se ejecute con la identidad de un usuario determinado diferente del usuario que lo ejecuta.	Seguridad en los archivos: A cada archivo se puede asignar varios grupos de atributos basados en usuarios o grupos, con permisos de creación, lectura, escritura, remoción, y ejecución, creando fácilmente listas de control de acceso (*ACLs*).
Maneja a todos los niveles del sistema un sistema verdadero de multiusuario, permitiendo nativamente que se puedan conectar simultáneamente diferentes usuarios, y manteniendo los recursos que ocupan cada	No existe realmente el concepto de multiusuario, aunque ha habido intentos de lograrlo, nunca han sido exitosos. Esto es, en buena parte, por una deficiente protección de memoria y recursos, y por utilizar un

uno de ellos perfectamente aislados de los procesos de otros usuarios.	diseño de sistema operativo como servidor de archivos únicamente, no como servidor de aplicaciones.
TCP/IP fue desarrollado sobre UNIX, por tanto su implementación es la más segura y ampliamente probada. Si una operación ilegal llega a bloquear el subsistema de TCP, este típicamente se reestablece tan pronto esta operación termina.	El subsistema de TCP/IP para Windows fue creado para seguir, en la medida de lo posible, los estándares, sin embargo, en caso de haber operaciones ilegales, es muy raro que el sistema pueda continuar operando, pues casi siempre el resultado es que Windows cae en la "pantalla azul de la muerte", situación tras la cual hay que reiniciar forzosamente el sistema a mano.
Al ser Linux software libre, no pasan normalmente más de un par de horas entre que es encontrado un error y que este es corregido y la corrección publicada. Esto hace que el impacto de cualquier problema de seguridad sea mínimo. Desde 1996, cuando fue liberado el Kernel (núcleo) 2.0.0, se liberaron 38 revisiones a este. El Kernel 2.2.0 fue liberado a principios de marzo de 1999, y (a fines de 1999) ya va en su decimosegunda revisión.	Windows está basado en el esquema de "seguridad a través de la oscuridad": El usuario no tiene acceso al código, por tanto no le es tan fácil encontrar errores. Sin embargo, cuando estos llegan a ser encontrados no aparece un parche sino hasta meses después, con el “service pack” correspondiente. Desde 1996, fecha en que apareció Windows NT 4.0, sólo han sido publicados 5 *service packs* - el último de ellos midiendo más de 30MB. Un caso muy notorio de los problemas que puede causar la seguridad a través de la oscuridad es el que se dio a conocer el 14 de abril del 2000, tras más de dos años de existencia - En todos los servidores Windows NT con extensiones de Frontpage 98 viene una puerta trasera secreta, con la contraseña *Netscape enigeers are weenies*, afectando a millones de servidores en todo el mundo.

Tabla 1: Comparativa Seguridad.

También cabe destacar que Windows es más fácil de manejar que Linux, pero de hay que haya distintas versiones de Linux que hacen esta tarea algo más fácil, entre estas cabe destacar: Linuxconf, GNOME, KDE, o GNUstep.

Estas son algunas razones de porque se ha elegido Linux, pero también hay otras como son el rendimiento, el costo, la confiabilidad, la funcionalidad, fiabilidad… pero que no se van a mencionar ya que este proyecto esta enfocado en temas de seguridad y las demás características no son relevantes para este tema.

5.2 Servidor Web: caso Apache.

El tráfico WWW es uno de los mayores componentes del uso de Internet hoy en día. Existen una variedad de servidores WWW bastantes populares, siendo el más famoso Apache con más del 50% del mercado [31].

La seguridad, ya sea en Apache o en cualquier otro servidor, es un tema bastante amplio y muy serio. Por defecto Apache se ejecuta como el usuario “nobody”, lo cual le da muy poco acceso al sistema. En general, la mayoría de los servidores WWW simplemente toman datos del sistema y los envían fuera, los mayores peligros no vienen del Apache sino de programas descuidados que se ejecutan vía Apache (CGI’s, server side includes, etc.).

El servidor Apache es un software que esta estructurado en módulos. La configuración de cada módulo se hace mediante la configuración de las directivas que están contenidas dentro del mismo. Los módulos del Apache se pueden clasificar en tres categorías:

- Módulos Base: Módulo con las funciones básicas del Apache
- Módulos Multiproceso: son los responsables de la unión con los puertos de la máquina, aceptando las peticiones y enviando a los hijos a atender a las peticiones.
- Módulos Adicionales: Cualquier otro módulo que le añada una funcionalidad al servidor.

Las funcionalidades más elementales se encuentran en el módulo base, siendo necesario un módulo multiproceso para manejar las peticiones. Se han diseñado varios módulos multiproceso para cada uno de los sistemas operativos sobre los que se ejecuta el Apache, optimizando el rendimiento y rapidez del código.

El resto de funcionalidades del servidor se consiguen por medio de módulos adicionales que se pueden cargar. Para añadir un conjunto de utilidades al servidor, simplemente hay que añadirle un módulo, de forma que no es necesario volver a instalar el software.

5.2.1 Comprobación del entorno.

Antes de empezar se debe comprobar que el servidor Apache está compilado con los módulos de seguridad que vamos a utilizar.

Para ver listar los módulos ejecutamos:

```
# cd /usr/local/apache2/bin
# ./httpd -l
```

Módulos compilados:

- core.c: Funciones básicas del Apache que están siempre disponibles.
- mod_access.c: proporciona control de acceso basándose en el nombre del host del cliente, su dirección IP u otras características de la petición del cliente.
- mod_auth.c: autentificación de usuario utilizando ficheros de texto.
- mod_auth_digest.c: autentificación de usuario utilizando MD5.
- mod_include.c: Documentos HTML generados por el servidor (Server Side Includes).
- mod_log_config.c: registro de las peticiones hechas al servidor.
- mod_env.c: modificación del entorno que se envía a los scripts CGI y las páginas SSI.

- mod_setenvif.c: permite la configuración de las variables de entorno basándose en las características de la petición.
- mod_ssl.c: criptografía avanzada utilizando los protocolos Secure Sockets Layer y Transport Layer Security.
- prefork.c: Implementa un servidor sin hilos.
- http_core.c
- mod_mime.c: asocia las extensiones de peticiones de los ficheros con el comportamiento del fichero (manejadores y filtros) y contenido (tipos mime, idioma, juego de caracteres y codificación).
- mod_status.c: proporciona información en la actividad y rendimiento del servidor.
- mod_autoindex.c: muestra los contenidos de un directorio automáticamente, parecido al comando ls de Unix.
- mod_asis.c: envío de ficheros que tienen sus propias cabeceras http.
- mod_cgi.c: Ejecución de Scritps CGI.
- mod_negotiation.c: se proporciona para la negociación del contenido.
- mod_dir.c: Proporcionado para redirecciones y para servir los ficheros de listado de directorios.
- mod_imap.c: proceso de imágenes en el lado del servidor.
- mod_actions.c: este módulo se utiliza para ejecutar Scripts CGI, basándose en el tipo de medio o el método de petición.
- mod_userdir.c: directorios específicos para usuarios.
- mod_alias.c: proporcionado para mapear diferentes partes del sistema de ficheros del servidor en el árbol de documentos del servidor, y para redirección de URL's.
- mod_so.c: carga del código ejecutable y los módulos al iniciar o reiniciar el servidor.

En la lista deben aparecer: mod_auth.c, mod_auth_digest.c, mod_ssl.c, mod_auth.c. Si los módulos mod_auth.c y mod_auth_digest.c no aparecen tendremos que recompilar Apache.

5.2.2 Instalación de apache en Linux.

Para la instalación del servidor Apache se deben seguir los siguientes pasos:

- Se comprueba que están instalados los siguientes paquetes:
    - flex
    - libxml2
    - libxml2-devel
    - ncurses
    - ncurses-devel
    - mysql-devel

  - gcc-c++

- Se baja el Apache de http://httpd.apache.org/download.cgi, para este proyecto se ha bajado el Apache httpd-2.0.52.tar.gz.
- Se descomprime el paquete en una carpeta con el comando:

  ```
  tar -zxvf httpd-2.0.52.tar.gz
  ```

- Se entra como root a la carpeta donde se ha descomprimido apache y se configura el código fuente para que apache sea instalado en el directorio /usr/local/apache2, para ello:

  ```
  ./configure --prefix=/usr/local/apache2
  ```

  prefix indica la ruta donde se va a instalar el servidor Apache.
- Una vez terminada la configuración del código fuente se pasa a crear los binarios, si da algún error es por que falta alguna librería pero si se añaden todas las librerías anteriores no habrá ningún problema. Para crear estos binarios se ejecuta:

  ```
  make
  ```

- Cuando terminan de crearse los binarios, se deben instalar. Estos binarios se instalarán en la ruta que se ha dado a prefix. Se utiliza la función:

  ```
  make install
  ```

- Seguidamente se moverá todo el código fuente de Apache 2 (httpd-2.0.52, carpeta temporal) al directorio de instalación (/usr/local/apache2), y se renombrará el directorio a fuente o source, esto es esencial cuando se instalan Módulos en Apache. Una vez cambiado el nombre se copia con:

  ```
  cp -r fuente /usr/local/apache2/
  ```

- Llegado a este punto ya tenemos el servidor Apache instalado:
  - Para arrancarlo se utiliza:

    ```
    /usr/local/apache2/bin/apachectl start
    ```

  - Para pararlo se utiliza:

    ```
    /usr/local/apache2/bin/apachectl stop
    ```

  - Para reiniciarlo se utiliza:

    ```
    /usr/local/apache2/bin/apachectl restart
    ```

- Ahora queda modificar el archivo httpd.conf, el cual se encuentra en /usr/local/apache2/conf/ y se pone lo siguiente:

  ```
  ServerName localhost
  ```

  `DocumentRoot "/usr/local/apache2/htdocs"`: Directorio donde se aloja la Web.

  `<Directory "/usr/local/apache2/htdocs">`: Se pone lo mismo que en DocumentRoot.

  `DirectoryIndex index.html, index.htm, index.php, home.html`: indica el documento que se carga al acceder al mismo.

- Para comprobar si el servidor funciona se pone en el navegador 127.0.0.1 o localhost y aparecerá la página que se ha puesto en el DocumentRoot, sino hemos colocado ninguna página aparecerá un mensaje de bienvenida.

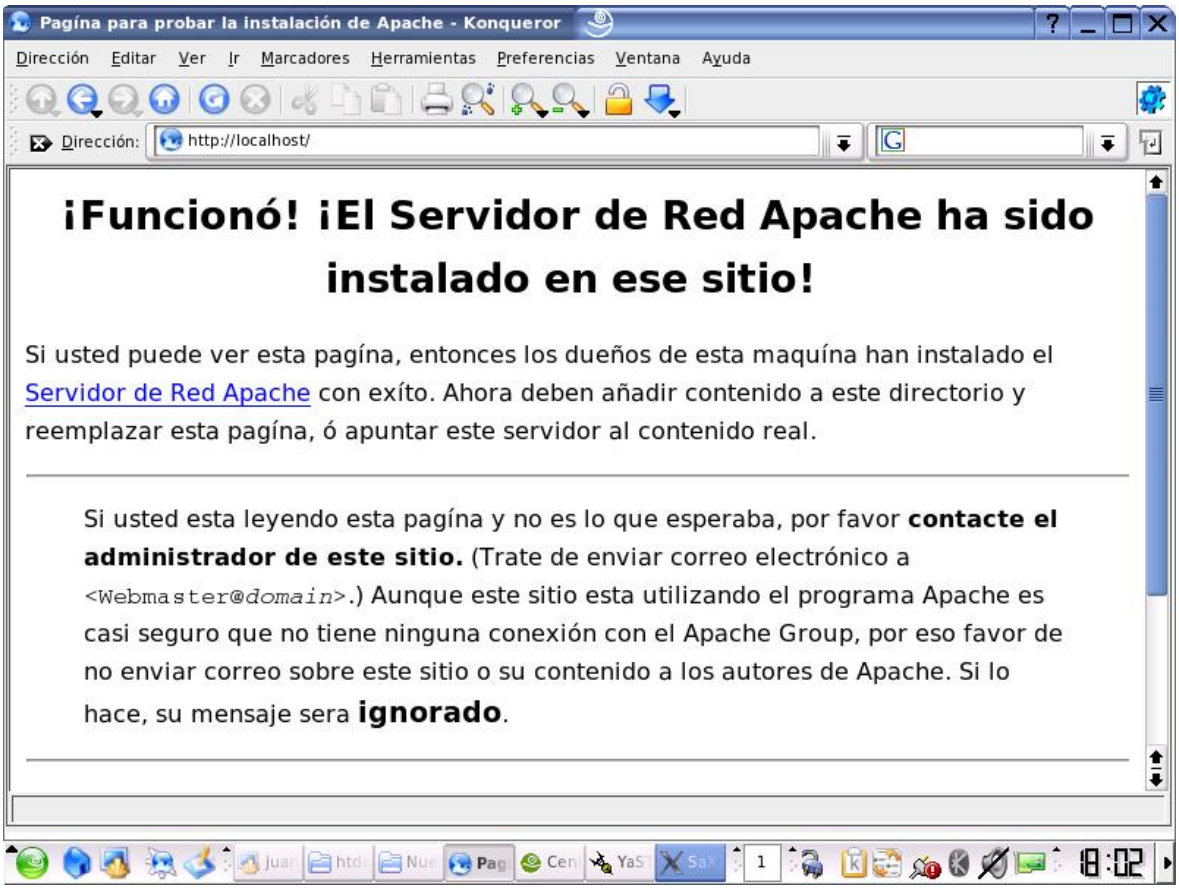

Figura 18: Mensaje del funcionamiento de apache.

5.2.3 Configurar apache para seguridad

Para activar el servidor seguro, necesita, como mínimo, tener instalados los siguientes paquetes:

- apache: El paquete apache contiene el demonio httpd y utilidades relacionadas, ficheros de configuración, iconos, módulos de Apache, páginas man y otros ficheros usados por el servidor Web Apache.
- mod_ssl: El paquete mod_ssl incluye el módulo mod_ssl, que proporciona criptografía fuerte para el servidor Web Apache a través de los protocolos SSL (Secure Sockets Layer) y TLS (Transport Layer Security).
- openssl: El paquete openssl contiene el kit de herramientas OpenSSL. Dicho kit implementa los protocolos SSL y TLS y también incluye una librería criptográfica de propósito general.
- mm: El paquete mm contiene la librería MM, la cual permite múltiples instancias del demonio httpd para compartir información de estado.

La siguiente tabla muestra la localización de los paquetes de servidor seguro y paquetes adicionales relativos a la seguridad. También nos dice si cada paquete es opcional o no para la instalación de un servidor Web seguro:

Nombre del paquete	**Localizado en el Grupo**	**¿Opcional?**
Apache	Entorno de Sistema/Demonio	No
Mod_ssl	Entorno de Sistema/Demonio	No
openssl	Entorno de Sistema/Librerías	No
mm	Entorno de Sistema/Librerías	No
Apache-devel	Desarrollo/Librerías	Sí
Apache-manual	Documentación	Sí
openssh	Aplicaciones/Internet	Sí
Openssh-askpass	Aplicaciones/Internet	Sí
Openssh-askpass-gnome	Aplicaciones/Internet	Sí
Openssh-clients	Aplicaciones/Internet	Sí
Openssh-server	Entorno de Sistema/Demonio	Sí
Openssl-devel	Desarrollo/Librerías	Sí
stunnel	Aplicaciones/Internet	Sí

Tabla 2: Paquetes del servidor seguro.

5.2.3.1 El protocolo SSL.

El protocolo SSL proporciona sus servicios de seguridad cifrando los datos intercambiados entre el servidor y el cliente con un algoritmo de cifrado simétrico [32], típicamente el RC4 o IDEA, y cifrando la clave de sesión de RC4 o IDEA mediante un algoritmo de cifrado de clave pública, típicamente el RSA. La clave de sesión es la que se utiliza para cifrar los datos que vienen y van al servidor seguro. Se genera una clave de sesión distinta para cada transacción, lo cual permite que aunque sea reventada por un atacante en una transacción dada, no sirva para descifrar futuras transacciones. MD5 se usa como algoritmo de hash.

Antes de hablar más sobre SSL se definen algunos términos:

- RSA Private Keys: fichero digital que se puede usar para descifrar mensajes que se mandan. Tiene una parte pública (que se distribuye con el certificado), que permite a los usuarios cifrar los mensajes que manda. Este mecanismo de clave asimétrica asegura que los mensajes cifrados con la clave pública (que se distribuyen a muchos usuarios) sólo pueden ser descifrados con la clave privada.
- Certificate Signing Request (CSR): es un fichero digital que contiene la clave pública y el nombre.
- Certification Authority (CA): entidad de confianza encargada de firmar certificados (CSR).
- Certificate (CRT): Una vez la CA ha firmado el CSR, obtenemos un CRT. Este fichero contiene la clave pública, el nombre, el nombre de la CA, y está firmado digitalmente por la CA. De esta forma otras entidades pueden verificar esta firma para comprobar la veracidad del certificado. Es decir, si se obtiene un certificado que esta firmado por una CA que se considera de confianza, se puede confiar también en la autenticidad del certificado.

Se puede ver que hay varias posibilidades para configurar SSL. Por ejemplo:

Cualquier cliente puede conectarse a una URL determinada, usando https. En este caso el servidor enviará su certificado al cliente para que este pueda descifrar la información que le llega del servidor y cifrar la que envía hacia él.

También se puede hacer que sólo los clientes que tengan un determinado certificado puedan conectarse a una determinada URL.

Otra posibilidad (la que se lleva a cabo en este proyecto) es combinar el primer ejemplo con las técnicas de autenticación que se han visto antes. De forma que cuando se intente acceder a una determinada URL usando https se trendrá que autentificar primero.

5.2.3.2 Creación de nuestra propia CA.

Existen varias CAs que, previo pago, pueden firmar el CSR. Estas CAs son mundialmente conocidas de forma que cualquier cliente podrá conectarse con confianza a el servidor.

Un ejemplo de este tipo de entidades de certificación puede ser VISA o VeriSign.

En este caso, como se está probando, se crea una CA. Para facilitar este tipo de tareas el paquete openssl proporciona el script CA.sh (o CA.pl en Perl). Se ejecuta:

- `cd /usr/local/apache2`
- `/usr/lib/ssl/misc/CA.sh -newca`

Se hará las siguientes preguntas:

Nombre del certificado de la CA o que se pulse “enter” para crearlo. En este caso se pulsa “enter” para crear uno nuevo.

Una “pass phrase”, que se vuelve a preguntar para confirmarla. Esta será la clave para acceder a la clave privada (la clave privada se guarda cifrada). Se debería poner más de una letra.

Cierta información para añadir al certificado: código del país, provincia, localidad, nombre de la organización, de la unidad...

Una vez finaliza la ejecución del script se puede ver que en el directorio /usr/local/apache2 ha aparecido un nuevo directorio: demoCA. Este tiene la siguiente estructura:

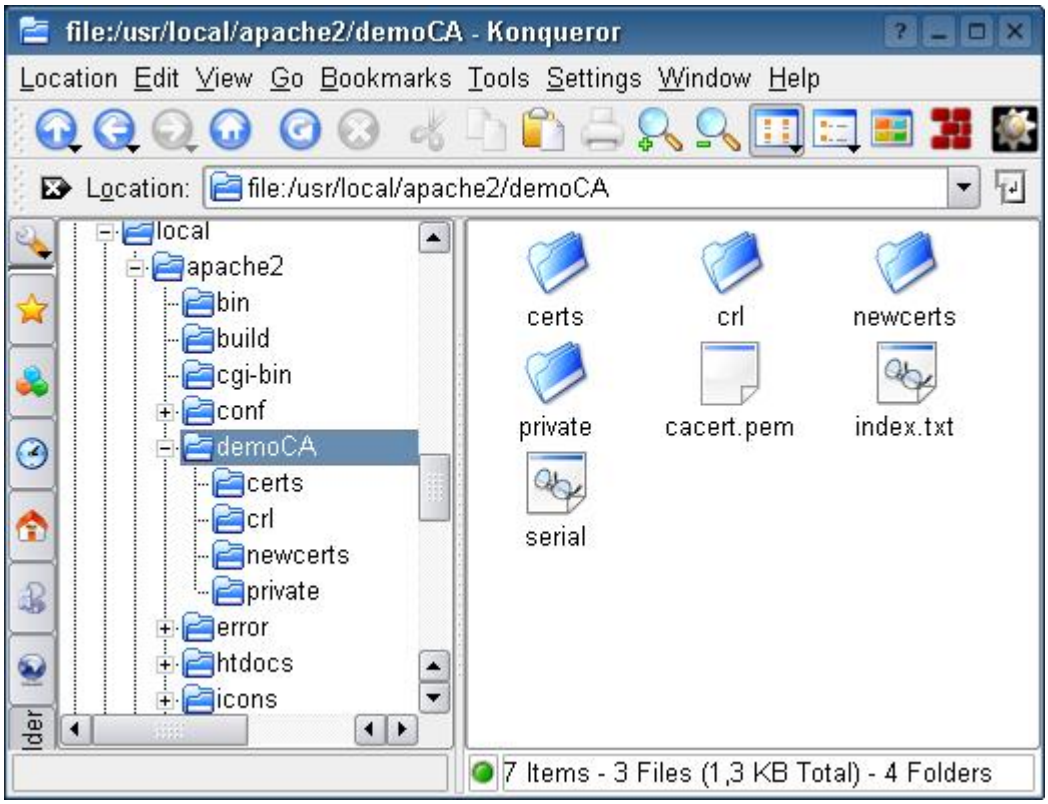

Figura 19: Estructura del directorio demoCA.

El fichero cacer.pem es el certificado de la CA, que luego se usará para firmar el certificado.

Los subdirectorios están vacíos, a excepción de private, donde se encuentra el fichero cakey.pem. Este fichero es la clave de la CA.

5.2.3.3 Creación de nuestro CSR.

Seguidamente se crea el CSR que debería firmar una auténtica CA [33], en este proyecto se firmará con la CA que se ha creado en el punto anterior. Para esto se siguen los siguientes pasos:

- Se crea la clave para el servidor Apache (la clave será triple-DES y en formato PEM):

  ```
  openssl genrsa -des3 -out server.key 1024
  ```

  El comando pedirá un “pass phrase” y la volverá a preguntar para confirmarla.

- Se crea el CSR (estará en formato PEM), usando la clave generada en el punto anterior:

  ```
  openssl req -new -key server.key -out server.csr
  ```

Lo primero que hará será pedir la “pass phrase” que se le puso a la clave en el punto anterior. Luego pedirá los datos que se quieren añadir al certificado. Aquí es importante tener en cuenta que cuando pregunte por el “Common Name” se debe poner el nombre completo del dominio del servidor, por ejemplo, si se va a acceder usando https://www.foo.dom/, se tendrá que poner www.foo.dom.

Con esto se consigue generar el fichero server.csr.

5.2.3.4 Creación de un CRT.

El último paso es firmar el CSR para conseguir el CRT. Para esto se vuelve a usar el script CA.sh. El problema que tiene este script es que trabaja con unos nombres fijos de ficheros, así que antes de ejecutarlo se tiene que renombrar el fichero server.csr.

```
ln -s server.csr newreq.pem
CA.sh -signreq
```

Se pedirá la "pass phrase" de la clave que se ha usado para generar el certificado de la CA.

Se mostrará la información del certificado, y preguntará si se ha firmar el mismo. Por supuesto, se contestará que sí.

Preguntará si se quiere hacer "comit" de los certificados firmados, es decir si se confirma la operación. De nuevo, se contestará que sí.

Habrá generado el fichero newcert.pem. Este fichero se renombrará por server.crt.

```
mv newcert.pem server.crt
```

5.2.3.5 Configuración apache.

Para configurar apache lo primero que se debe hacer es poner los ficheros donde corresponde.

```
mkdir conf/ssl.key
mkdir conf/ssl.crt
mv server.key conf/ssl.key/
mv server.crt conf/ssl.crt/
```

Posteriormente se añade al fichero /usr/local/apache2/conf/httpd.conf:

```
<Directory "/usr/local/apache2/htdocs/private">
    SSLRequireSSL
    AuthName "privatefiles"
    AuthType Basic
    AuthUserFile /usr/local/apache2/conf/passwd_basic
    Require valid-user
</Directory>
```

Se puede ver que es la misma configuración que se usa en el apartado 4.3.1. salvo que se ha añadido la directiva SSLRequireSSL. Esta directiva prohíbe el acceso a no ser que sea HTTP sobre SSL (HTTPS).

Con esto se consigue que para acceder al directorio /usr/local/apache2/htdocs/private sea obligatoriamente usando HTTPS, y previa autenticación. Por último sólo queda arrancar Apache. Hay que tener en cuenta que para arrancar Apache con soporte SSL hay que utilizar:

```
/usr/local/apache2/bin/apachectl startssl
```

Se puede ver que al arrancar Apache pide una "pass phrase". Esta es la que se usa al crear el certificado. Esto lo hace porque la clave esta guardada cifrada.

Si se quiere evitar tener que escribir la “pass phrase” cada vez que se arranca Apache se puede guardar la clave sin cifrar, pero esto no es nada recomendable. Se puede tener un grave problema de seguridad.

Para generar una clave sin cifrar se puede hacer:

```
cd /usr/local/apache2/conf/ssl.key/
openssl rsa -in server.key -out server.unencrypted.key
```

Luego es conveniente restringir los permisos al máximo.

```
chmod 400 server.unencrypted.key
```

Otra manera para no tener que escribir la “pass phrase” es usar la directiva SSLPassPhraseDialog exec: ... Donde se cambiarán los puntos suspensivos por un programa que saque por la salida estándar la “pass phrase”.

Hay que tener en cuenta que este método no tiene por que ser más seguro que el anterior. En general no es recomendable ninguno de los dos para un sistema seguro.

5.2.3.6 Limpieza de archivos.

Los ficheros que necesita Apache sólo son:

Clave: `/usr/local/apache2/conf/ssl.key/server.key`.

Certificado firmado por la CA: `/usr/local/apache2/conf/ssl.crt/server.crt`.

Así que si se quiere se pueden borrar el resto de los archivos que se han generado para firmar el certificado.

```
cd /usr/local/apache2/
rm server.csr newreq.pem
rm -Rf demoCA/
```

LENGUAJE DE PROGRAMACIÓN Web Y BASES DE DATOS.

6.1 Lenguaje de programación Web php

Navegar por la Web en busca de información puede ser una experiencia aburrida o agradable, dependiendo de algunas características de las páginas Web, como su facilidad de uso, apariencia visual e interactividad con el usuario [34]. Si ha navegado por la Web recientemente, habrá notado que el aspecto y funcionalidad de las páginas han sufrido un gran cambio. Ya pasó la época en la que una página se componía sólo de algunas imágenes estáticas e información de texto, actualmente, las páginas Web tienen mucho más que ofrecer que sólo contenidos llamativos o vistosos. Lo que ahora distingue a una página Web de otra, es el nivel de interacción con el usuario y su dinamismo, afectando directamente a su popularidad.

Algunos de los lenguajes de script más famosos que hoy se usan para la Web son Perl, Tcl, Python y JavaScript. Sin embargo, un nombre que destaca sobre todos ellos es PHP. Se trata de un lenguaje relativamente simple y uno de los más jóvenes comparado con otros del mismo tipo.

Hoy en día, la interacción y el dinamismo se consideran las características más importantes de una página Web, aparte obviamente de su viscosidad. Para incluir estas características en las páginas, los lenguajes de script deben centrar su atención en las aplicaciones Web.

PHP es un lenguaje para programar scripts del lado del servidor, que se incrustan dentro del código HTML. Este lenguaje es gratuito y multiplataforma. Es el acrónimo de Hipertext Preprocesor, gratuito e independiente de plataforma, rápido, con una gran librería de funciones y mucha documentación.

Un lenguaje del lado del servidor es aquel que se ejecuta en el servidor Web, justo antes de que se envíe la página a través de Internet al cliente. Las páginas que se ejecutan en el servidor pueden realizar accesos a bases de datos, conexiones en red, y otras tareas para crear la página final que verá el cliente. El cliente solamente recibe una página con el código HTML resultante de la ejecución de la PHP. Como la página resultante contiene únicamente código HTML, es compatible con todos los navegadores.

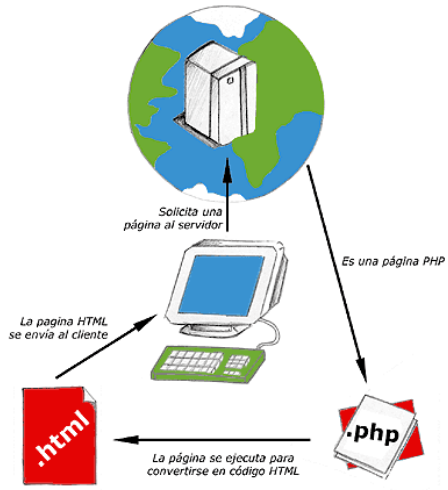

Figura 20: Esquema del funcionamiento de las páginas PHP.

Una vez que ya se conoce el concepto de lenguaje de programación de scripts del lado del servidor se puede hablar de PHP: se escribe dentro del código HTML, lo que lo hace realmente fácil de utilizar, pero con algunas ventajas como su gratuidad, independencia de plataforma, rapidez y seguridad. Cualquiera puede descargar a través de la página principal de PHP [35] y de manera gratuita, un módulo que hace que el servidor Web comprenda los scripts realizados en este lenguaje. Es independiente de plataforma, puesto que existe un módulo de PHP para casi cualquier servidor Web. Esto hace que cualquier sistema pueda ser compatible con el lenguaje y significa una ventaja importante, ya que permite portar el sitio desarrollado en PHP de un sistema a otro sin prácticamente ningún trabajo.

PHP incorpora la potencia de los lenguajes relativamente antiguos, como Perl y Tcl, pero elimina sus debilidades:

- Es un lenguaje de script de código abierto para servidores.
- Es independiente del sistema operativo y puede ser utilizado en cualquiera de ellos, incluyendo Microsoft Windows, Mac OS, Linux, HP-UX y Solaris.
- Utiliza una amplia gama de servidores Web, tales como Apache, Microsoft Internet Information Sever, Netscape e iPlanet.
- Se conecta a gran cantidad de bases de datos, como MySQL, Ingres, Sybase, Oracle, Base, Informix, FrontBase y Unix dbm. Una de las características que distingue a PHP es que proporciona soporte a los sitios Web de comercio electrónico que manejan bases de datos.
- El código de PHP es más simple que en otros lenguajes de script.
- Se puede utilizar para crear imágenes y ficheros de lectura/escritura, así como para enviar mensajes de correo electrónico. Para proporcionar estor servicios, PHP se sirve de bastantes protocolos, como http, POP3, SNMP, LDAP e IMAP.

PHP, en el caso de estar montado sobre un servidor Linux u Unix, es más rápido que ASP, dado que se ejecuta en un único espacio de memoria y esto evita las comunicaciones entre componentes COM que se realizan entre todas las tecnologías implicadas en una página ASP.

Por último se señala la seguridad, en este punto también es importante el hecho de que en muchas ocasiones PHP se encuentra instalado sobre servidores Unix o Linux, que son de sobra conocidos como más veloces y seguros que el sistema operativo donde se ejecuta las ASP, Windows NT o 2000. Además, PHP permite configurar el servidor de modo que se permita o rechacen diferentes usos, lo que puede hacer al lenguaje más o menos seguro dependiendo de las necesidades de cada cual.

Fue creado originalmente en 1994 por Rasmus Lerdorf, pero como PHP está desarrollado en política de código abierto, a lo largo de su historia ha tenido muchas contribuciones de otros desarrolladores. Actualmente PHP se encuentra en su versión 4, que utiliza el motor Zend, desarrollado con mayor meditación para cubrir las necesidades de las aplicaciones Web actuales.

Este lenguaje de programación está preparado para realizar muchos tipos de aplicaciones web gracias a la extensa librería de funciones con la que está dotado. La librería de funciones cubre desde cálculos matemáticos complejos hasta tratamiento de conexiones de red.

Algunas de las más importantes capacidades de PHP son: compatibilidad con las bases de datos más comunes, como MySQL, mSQL, Oracle, Informix, y ODBC, por ejemplo. Incluye funciones para el envío de correo electrónico, upload de archivos, crear dinámicamente en el servidor imágenes en formato GIF, incluso animadas y una lista interminable de utilidades adicionales.

6.1.1 Instalación de php.

Para la instalación de php se debe tener el servidor Apache parado, se debe ser root y se seguirán los siguientes pasos [36]:

- Se baja el php de http://www.php.net/downloads.php y se utiliza la versión PHP 4.4.0 (tar.gz).
- Se descomprime el paquete en un directorio con el comando:

  ```
  tar -zxvf php-4.4.0.tar.gz
  ```

  Al descomprimirlo se crea la carpeta "php-4.4.0"
- Se entra en esta carpeta y se configura el php con:

  ```
  ./configure --with-mysql --with-apxs2=/usr/local/apache2/bin/apxs
  ```

El parámetro --with-mysql indica que el módulo PHP debe ser compilado con soporte para la Base de Datos MySQL. Y el parámetro --with-apxs2=/usr/local/apache2/bin/apxs indica el directorio del ejecutable apxs utilizado para compilar módulos, dicho ejecutable se encuentra en la instalación de Apache 2.

- Se compilan los archivos:

  ```
  make
  ```
- Se instalan los mismos:

  ```
  make install
  ```
- Terminada la instalación se genera el archivo php.ini, con el comando:

  ```
  cp php.ini-dist /usr/local/lib/php.ini
  ```
- Ahora hay q modificar el httpd.conf (`/usr/local/apache2/conf/`), indicarle al apache que cargue el módulo de PHP, pero se debe tener en cuenta que el comando *make install* puede haber puesto ya esa línea:

  ```
  LoadModule php5_module modules/libphp5.so
  ```
- Se indica al apache que todo documento con la extensión .php sea procesado por PHP:

  ```
  AddType application/x-httpd-php .php .phtml
  ```
- Se comprueba si funciona el módulo de php, para ello se arranca el apache:

  ```
  # /usr/local/apache2/bin/apachectl start
  ```
- Se crea un archivo al cual se le llama index.php que contenga:

  ```
  <?php phpinfo(); ?>
  ```

Se copia a /usr/local/apache2/htdocs o donde le hayas dicho al apache que busque las Web. Se abre el navegador y se pone *http://localhost/index.php*, debe de mostrar características de PHP.

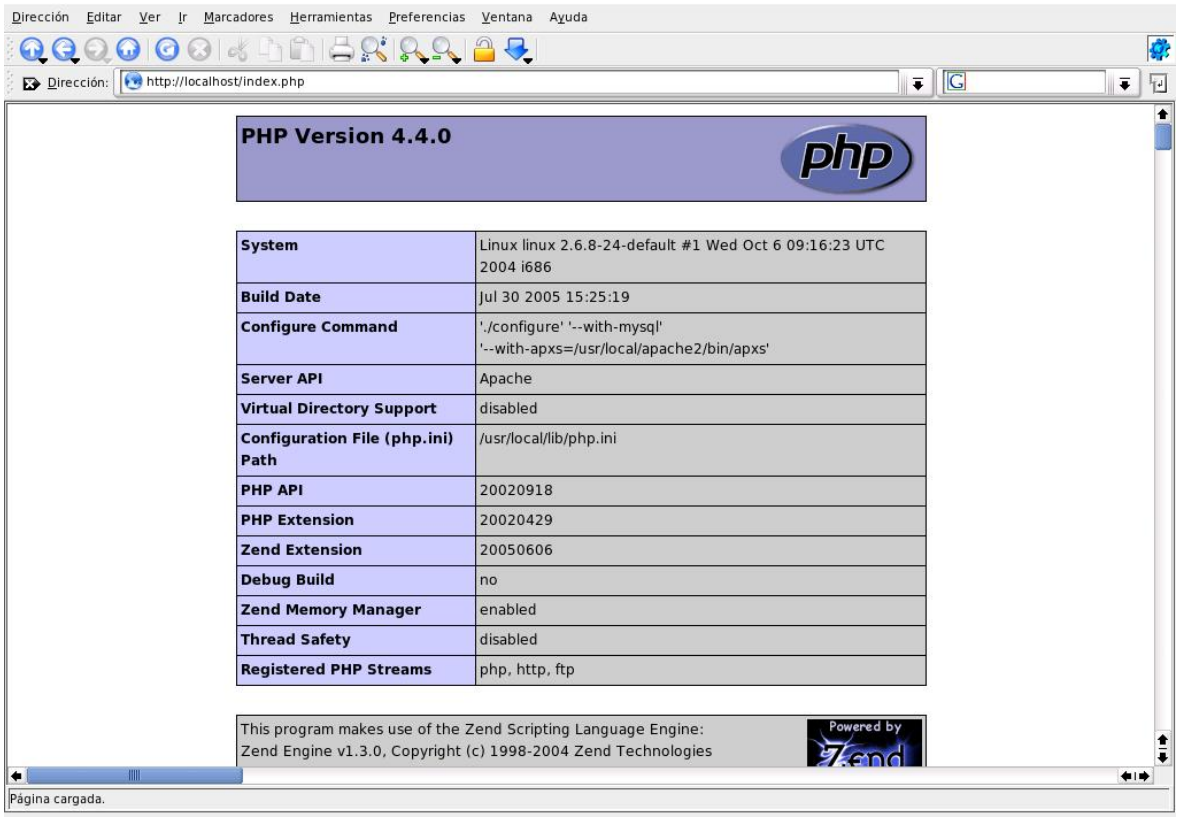

Figura 21: Pàgina principal de PHP.

6.2 Sesiones vs Cookies

6.2.1 Sesiones

Las sesiones son una facilidad que permite vincular información a un visitante a lo largo de sus diversos accesos a nuestro sitio web. Un visitante puede acceder a varias páginas del sitio, las sesiones ayudan a identificarlo y a vincularle información. Su uso cabe destacar que cuando se quiera almacenar información en relación a un visitante de forma persistente a lo largo de su visita las sesiones facilitarán la vida.

6.2.1.1 Funcionamiento sesiones

Se puede imaginar a la sesión como un número estampado en la frente del visitante que ayuda a identificarlo en cada página que visite del sitio. Además para cada número asignado a un visitante habrá un locker donde guardar sus cosas. Así que quien está a la entrada (el código de manejo de sesiones) al llegar un visitante escoge un número para él y se lo estampa en la frente para que lo utilice durante su visita.

Para poner a funcionar una sesión se necesitan cumplir dos requisitos fundamentales:

- Asignar al visitante un identificador único (escoger un número).
- Propagar este identificador único a través de los diferentes accesos del visitante para que podamos reconocerlo (estampárselo en la frente).

6.2.2 Cookies

Las cookie son nada más que ficheros de texto que algunos servidores piden al navegador que escriba en el disco duro, con información acerca de lo que se ha estado haciendo por sus páginas.

Estos son una potente herramienta para almacenar o recuperar información empleada por los servidores web debido al HTTP (protocolo de transferencia de ficheros). Este protocolo es sin estados (no almacena el estado de la sesión entre peticiones sucesivas) y ahí es dónde entra la "cookie" proporcionando una manera de conservar la información. El servidor con el envío de estos ficheros puede recopilar datos concernientes al perfil del usuario, tales como sus preferencias, nombre y contraseña, productos que más le interesan, etc.

6.2.3 Sesiones vs Cookies

Las cookies se almacenan en el disco duro del usuario, liberando así al servidor de una importante sobrecarga. Esto produce un aumento de ficheros en el disco duro del usuario y la posible manipulación de algún experto en el funcionamiento de este tipo de ficheros.

Sin embargo las sesiones se ejecutan en el lado del servidor, dándole mayor seguridad, aunque suponga una mayor carga para el servidor.

6.3 Bases de datos

Una base de datos es un conjunto de datos que pertenecen al mismo contexto almacenados sistemáticamente para su uso posterior [37]. En este sentido, una biblioteca puede considerarse una base de datos compuesta en su mayoría por documentos y textos impresos en papel e indexados para su consulta.

En la actualidad, y gracias al desarrollo tecnológico de campos como la informática y la electrónica, la mayoría de las bases de datos tienen formato electrónico, que ofrece un amplio rango de soluciones al problema de almacenar datos.

En informática existen los sistemas gestores de bases de datos (SGBD), que permiten almacenar y posteriormente acceder a los datos de forma rápida y estructurada.

Las aplicaciones más usuales son para la gestión de empresas e instituciones públicas. También son ampliamente utilizadas en entornos científicos con el objeto de almacenar la información experimental.

Aunque las bases de datos pueden contener muchos tipos de datos, algunos de ellos se encuentran protegidos por las leyes de varios países. Por ejemplo en España, los datos personales se encuentran protegidos por la Ley Orgánica de Protección de Datos de Carácter Personal (LOPD).

Sus características pueden ser ventajosas o desventajosas, pueden ser de ayuda para almacenar, organizar, recuperar, comunicar y manejar información en formas que serían imposibles sin los computadores, pero también afecta de alguna manera ya que existen enormes cantidades de información en bases de datos de las que no se tiene control del acceso.

Las bases de Datos tienen muchos usos: facilitan el almacenamiento de grandes cantidades de información; permiten la recuperación rápida y flexible de información, con ellas se puede organizar y reorganizar la información, así como imprimirla o distribuirla en formas diversas.

Los tipos de datos que se pueden introducir a una base de datos son:

- numéricos: se pueden introducir números para identificar partes del archivo, esto identifica la parte que numera al archivo o lo distingue de alguna manera.
- texto: el texto es un nombre que identifica al campo.
- etiquetas: son los títulos con los que cada campo es designado.
- fórmulas: son datos que aparecen como numéricos pero fueron hechos por medio de fórmulas.

6.3.1 Instalación de MySQL.

MySQL es actualmente el sistema de administración de bases de datos SQL más popular en el Open Source, ya que es ampliamente utilizado en los desarrollos Web gracias a su rapidez, estabilidad y a su integración con otros lenguajes de script como por ejemplo PHP [38]. Es recomendable utilizar la versión mas actualizada de los paquetes para no estar vulnerables a bugs u otro tipo de fallas.

Los pasos que se deben seguir para instalar una base de datos en Linux son los siguientes:

- Lo primero es verificar si existe el usuario y el grupo mysql, para esto se tienen que mirar los archivos /etc/passwd y /etc/group. El archivo /etc/passwd contiene los usuarios del sistema y el /etc/group tiene los grupos. Si se encuentra user/group mysql entonces se deben crear:

  `# groupadd mysql`: Crea el grupo mysql

  `# useradd -g mysql mysql`: Crea el usuario mysql y lo añade al grupo mysql

- Se baja el programa de: http://dev.mysql.com/get/Downloads/MySQL-4.0/mysql-4.0.21.tar.gz/from/http://mysql.rediris.es/ la versión bajada para este proyecto es la mysql-4.0.21.tar.gz
- Se descomprime el paquete utilizando:

  ```
  tar -zxvf mysql-4.0.21.tar.gz
  ```

- Una vez descomprimido el paquete se entra en la carpeta que se ha creado y se configura el código fuente del servidor MySQL para que se instale en /usr/local/mysql:

  ```
  ./configure --prefix=/usr/local/mysql
  ```

- Terminada la configuración se procede a generar los binarios para ello:

  ```
  make
  ```

- Con los binarios creados se procede a su instalación:

  ```
  make install
  ```

- Seguidamente cuando se ha terminado la instalación se copia el archivo my-medium.cnf al directorio /etc/ y se pone el nombre my.cnf:

  ```
  cp support-files/my-medium.cnf /etc/my.cnf
  ```

- Se va al directorio /usr/local/mysql y se crean unas tablas que le hacen falta a la base de datos con:

  ```
  cd /usr/local/mysql
  ```

```
bin/mysql_install_db --user=mysql
```

- Ahora se cambia el usuario y grupo de los ficheros:

```
chown -R root .
chown -R mysql var
chgrp -R mysql .
```

- Para arrancar MYSQL se debe poner:

```
# /usr/local/mysql/bin/mysqld_safe --user=mysql &
```

- Aparecen unas letras se pulsa intro y se cambia la contraseña donde pone nuevo_password entre comillas simples:

```
bin/mysqladmin -u root password 'nuevo_password'  Cambia la contraseña de root de mysql
```

- Ya se tiene el servidor MySQL instalado y en funcionamiento. Para comprobar que esta funcionando se pone:

```
/usr/local/mysql/bin/mysqlshow -p
```

Se teclea el password y debe salir una lista de las bases de datos que hay hechas, por defecto son dos, una que se llama mysql que sirve para controlar los usuarios y los permisos de las bases de datos y una de prueba que se llama test:

```
mysql> SHOW DATABASES; para mostrar las bases de datos que hay
+----------+
| Database |
+----------+
| mysql    |
| test     |
+----------+
2 rows in set (0.00 sec)
```

6.3.2 phpMyAdmin

PhpMyAdmin es una utilidad que sirve para interactuar con una base de datos de forma muy sencilla y desde una interfaz Web [39]. Sirve por ejemplo para crear bases de datos, tablas, borrar o modificar dados, añadir registros, hacer copias de seguridad, etc. Es una aplicación tan útil que casi todos los hosting con MySQL disponen de ella, por ello se analizará su instalación. Además, se usa para crear los usuarios MySQL para así poder utilizar las bases de datos de forma segura. Al ser una aplicación escrita en PHP, necesita de Apache y MySQL para poder funcionar.

Este proyecto se encuentra vigente desde el año 1998, como esta herramienta corre en máquinas con Servidores Web y Soporte de php y MySQL, la tecnología utilizada ha ido variando durante su desarrollo.

6.3.2.1 Instalación de phpMyAdmin

Se deben seguir los siguientes pasos:

- Se bajan los archivos de: http://www.phpmyadmin.net/home_page/downloads.php, para este proyecto se ha bajado la versión: phpMyAdmin-2.6.0-pl1.tar.gz
- Para descomprimirlo se utiliza:

  `tar -xzvf phpMyAdmin-2.6.0-pl1.tar.gz`
- Se cambia el nombre, se nombra phpMyAdmin y se copia al DocumetRoot:

  `mv phpMyAdmin-2.6.0-pl1 phpMyAdmin` para cambiarle el nombre

  `cp -r phpMyAdmin /usr/local/apache2/htdocs/` para copiarlo al DocumetRoot
- Se edita el archivo /usr/local/apache2/htdocs/phpMyAdmin/config.inc.php para cambiar los valores de host de la base de datos y el usuario y password con el que conectamos. Se pueden configurar muchos aspectos de la herramienta, se modifica:

  ```
  $cfg['PmaAbsoluteUri']= 'http://localhost/phpMyAdmin' Debemos asignarlo a la ruta completa necesaria para acceder a phpMyAdmin.
  $cfg['Servers'][$i]['user'] = 'root';
  $cfg['Servers'][$i]['password'] = 'Ponemos la contraseña de root de mysql';
  ```

Configurando las líneas señaladas anteriormente ya se está habilitado para utilizar la herramienta. En el navegador se ingresa a la dirección correspondiente http://localhost/phpMyAdmin y muestra:

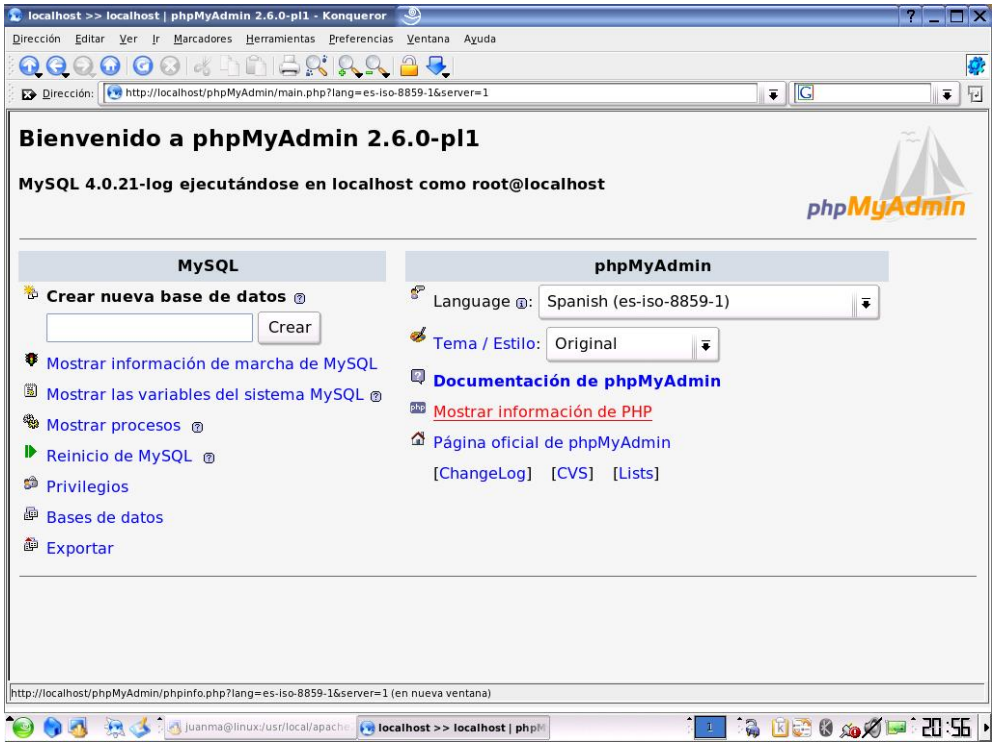

Figura 22: página principal de phpMyAdmin.

Si entramos en “Bases de datos” muestra las diferentes bases de datos que hay y la que nosotros hemos creado llamada “datos_usuarios”

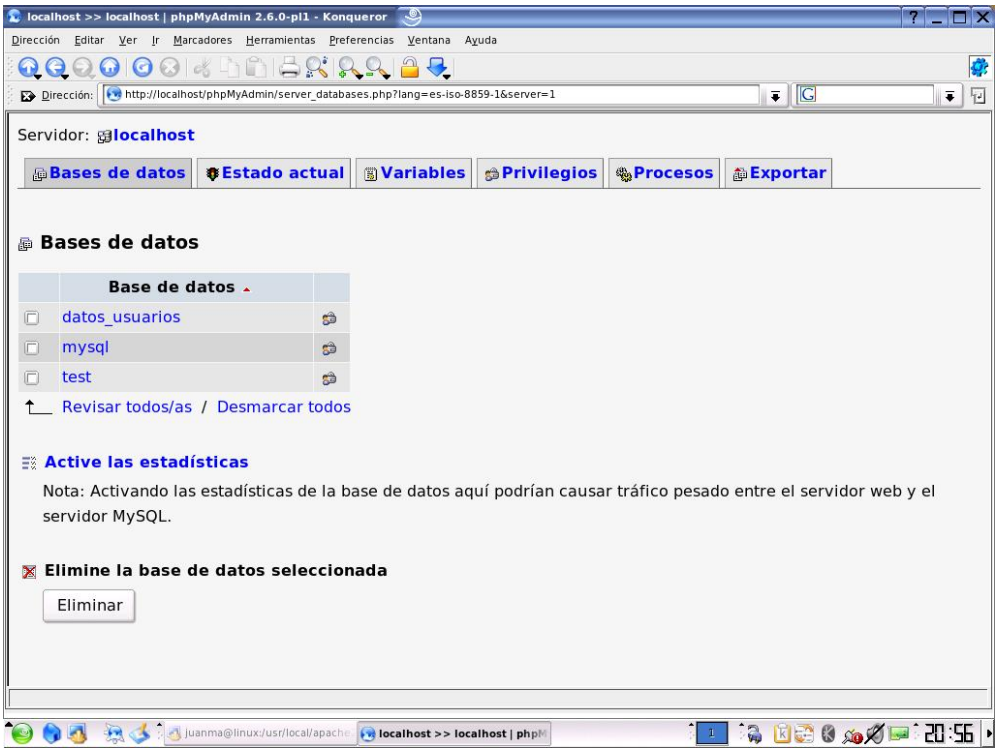

Figura 23: Bases de datos.

Como se puede comprobar en la figura 21 muestra la base de datos que se ha creado llamada “datos_usuarios” y otras que el programa MySQL crea automáticamente y que ya se ha explicado para que sirve cada una. Esta base de datos que se ha creado esta formada por tablas, si se entra en ella se puede ver su estructura:

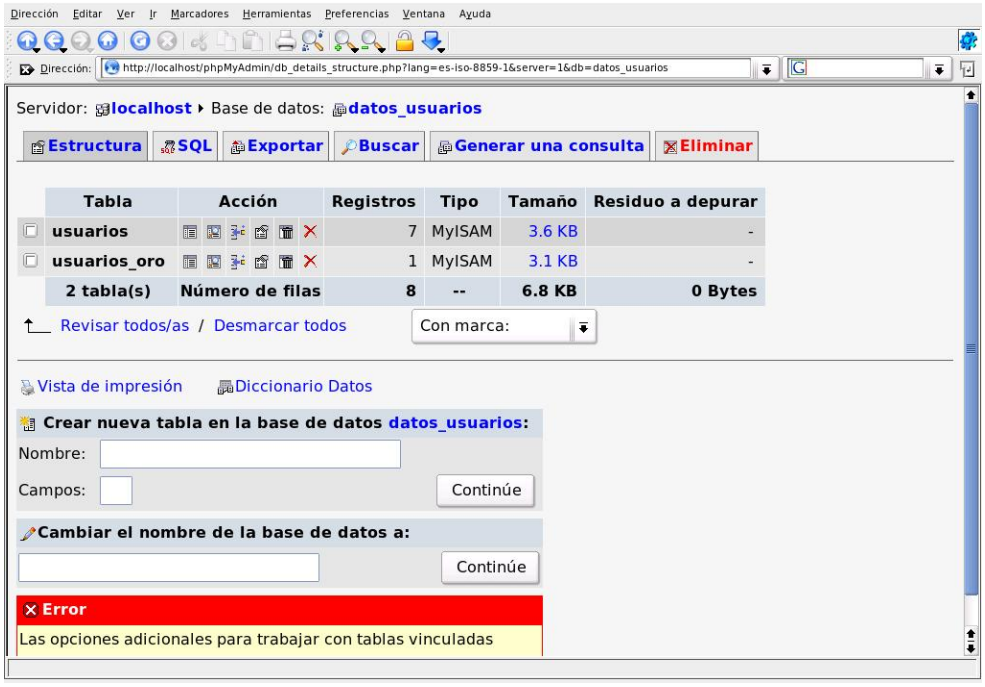

Figura 24: Tablas de la base de datos.

Si se observa la figura 22 se puede ver que aparecen dos tablas, una llamada “usuarios” y otra llamada “usuarios_oro”, en la tabla usuarios se guardan los datos de las personas que entran a la página y se registran, la otra tabla (“usuarios_oro”) es una tabla administrada por el creador de la página y es el administrador quien crea a los usuarios y

les da permisos para acceder, esto se ha hecho así para dar distintos privilegios a los distintos usuarios, el diseño de las tablas es el mismo en ambos casos.

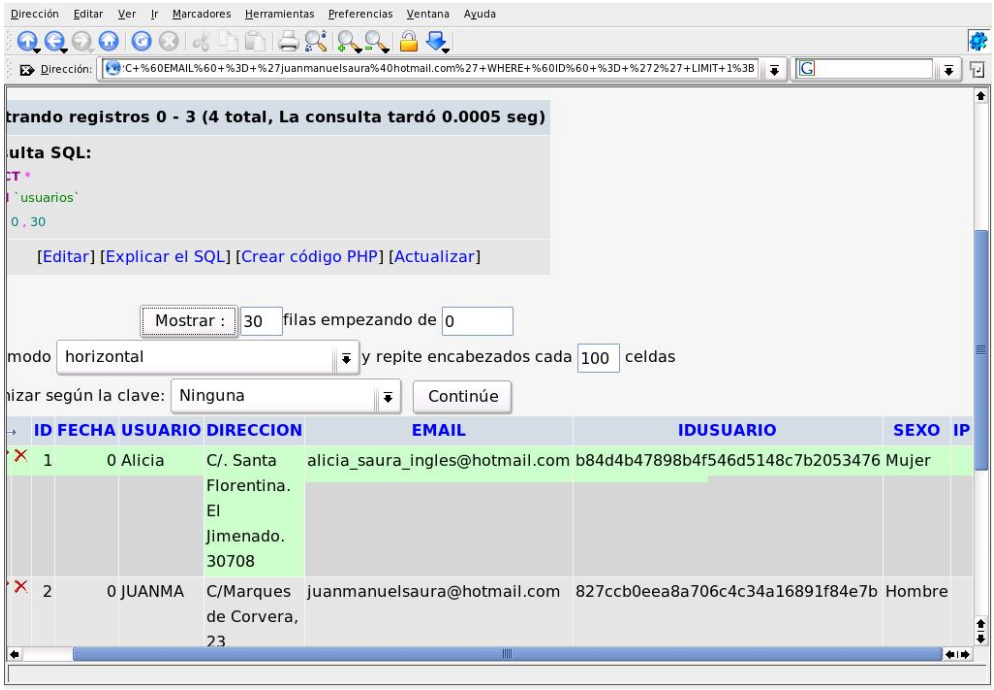

Figura 25: Registros de la base de datos usuarios.

Se puede ver que cuando un usuario se registra tiene que rellenar un formulario que contiene los diferentes campos que se muestran en la figura, fijándose en el campo “IDUSUARIO” se comprueba que la contraseña esta encriptada para dar mayor seguridad a la base de datos, ya que alguna persona puede averiguar la contraseña de la misma y entrar, pero si además se encriptan las contraseñas de los distintos usuarios que se registran va a ser más difícil la obtención de la contraseña.

APLICACIÓN DE SEGURIDAD: CASO PRÁCTICO

7.1 Introducción.

En este capítulo se explica el funcionamiento de la demostración que se ha realizado para este proyecto. Se ha programado una página Web y se ha hecho diversas pruebas de seguridad, tanto a nivel de software, como a nivel de programación. Como se ha comentado anteriormente el servidor Web es apache y el lenguaje de programación php, los datos se guardan en la base de datos MySQL, la instalación de los distintos programas y configuración de los mismos para que se interrelacionen entre sí está explicado en capítulos anteriores, aquí se va a indagar en la elaboración de la seguridad y cuales han sido los pasos para implantar esta.

7.2 Funcionamiento de la página Web.

Se arranca el servidor apache con soporte SSL, para que la información vaya cifrada, para ello se ejecuta lo siguiente:

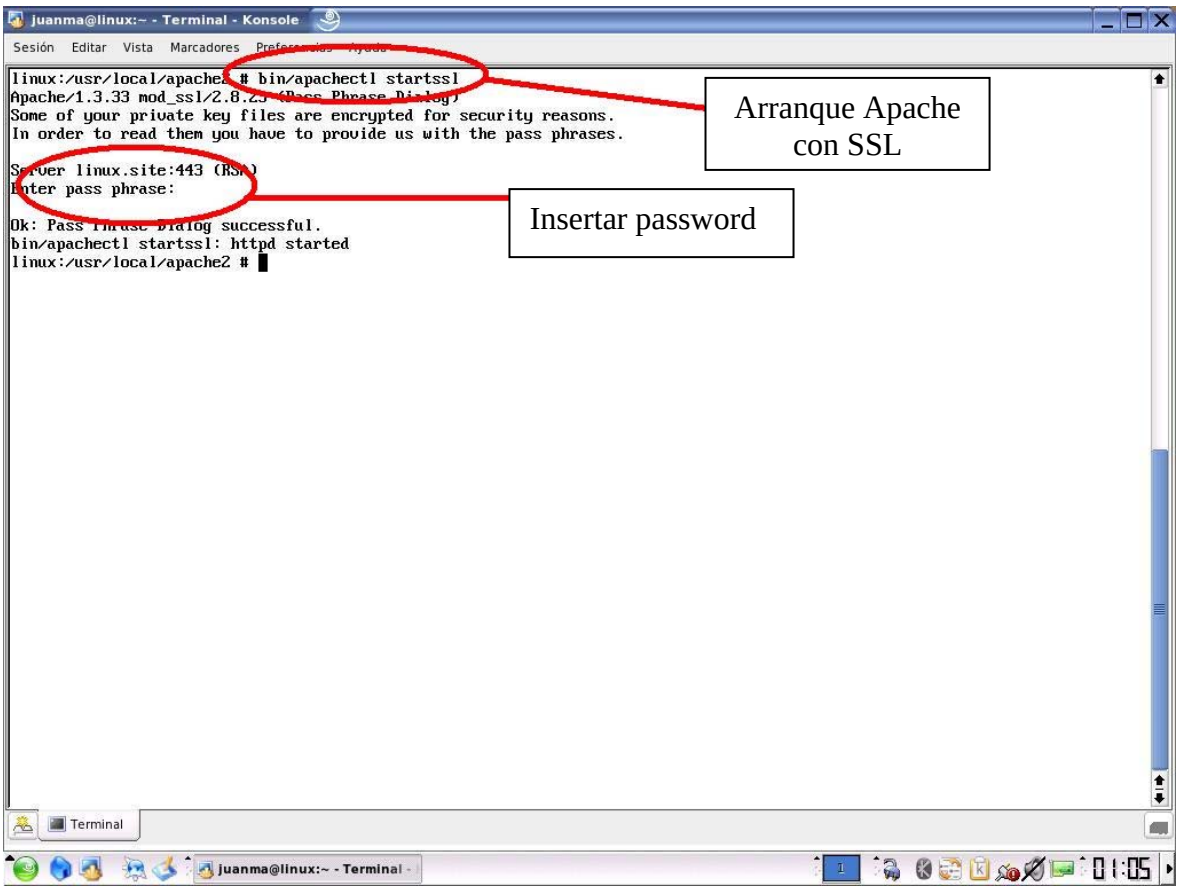

Imagen 26: Arranque del servidor Apache con soporte SSL.

Una vez en marcha el servidor Web, se debe arrancar la base de datos MySQL para ello se ejecuta lo siguiente:

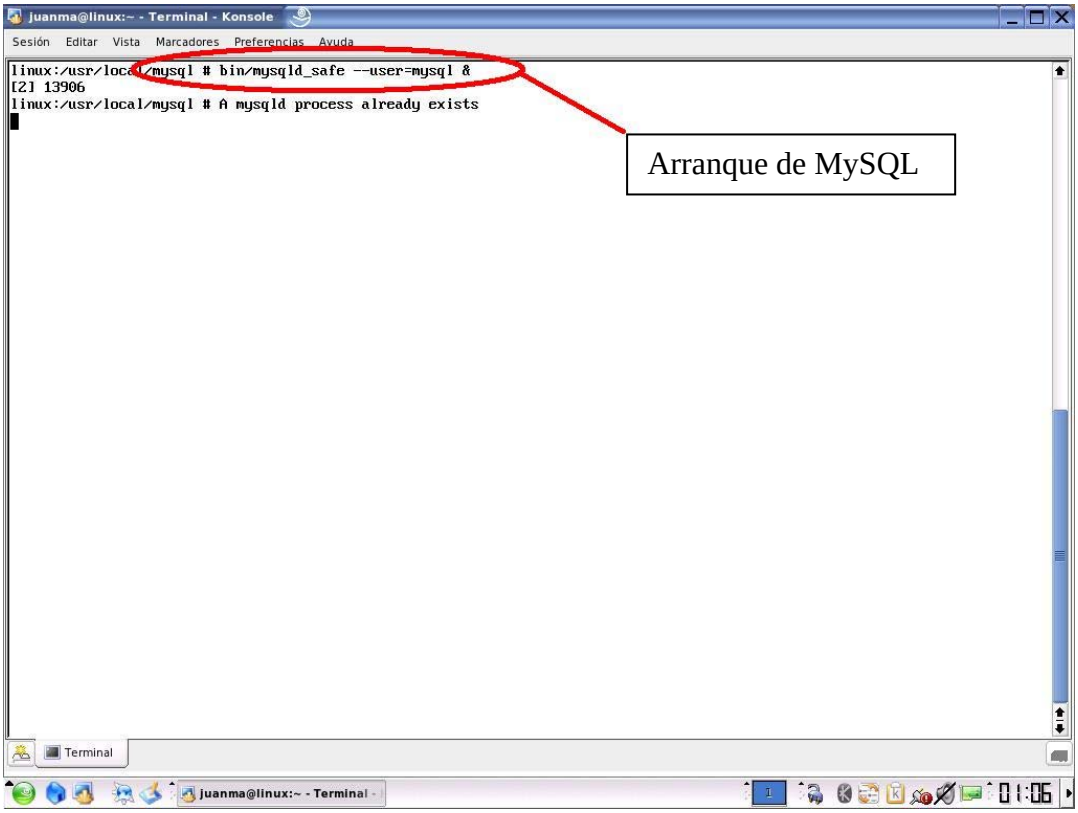

Figura 27: Arranque de la base de datos MySQL.

Arrancada la base de datos y el servidor, si se escribe en el navegador *localhost/MiSitio* aparecerá la página realizada:

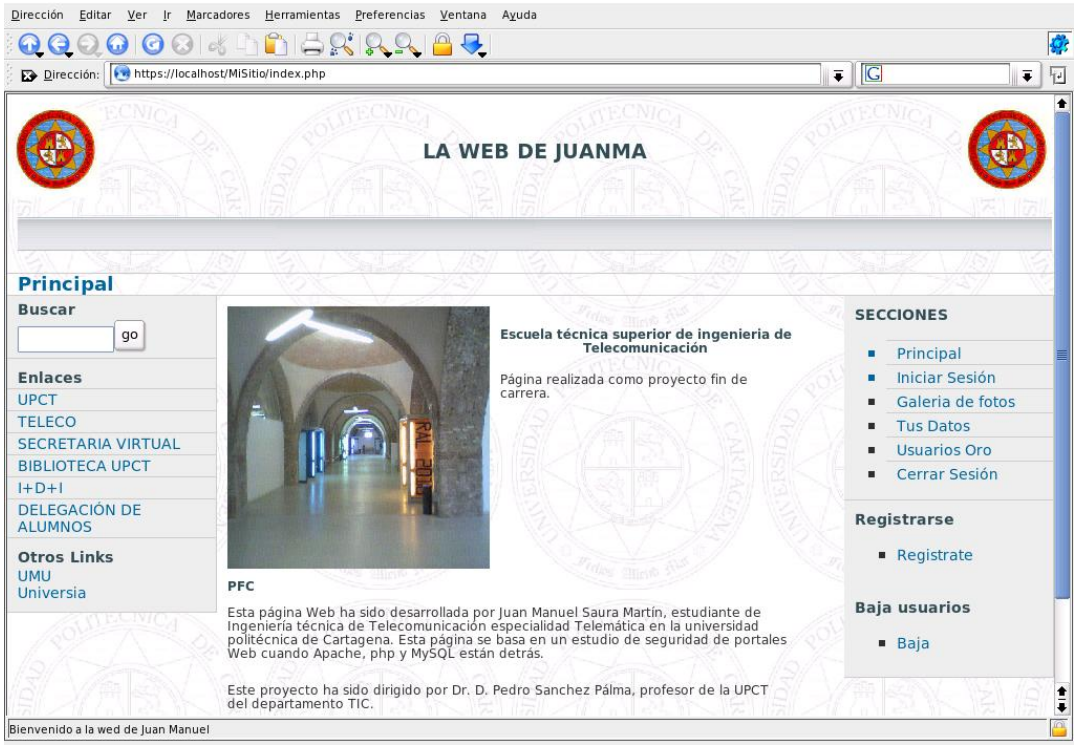

Figura 28: Página inicial

Como se observa en la figura 27, en la parte de la derecha hay distintos menús para iniciar la sesión, poder registrarse, o darse de baja, lo que se desee realizar y todo ello bajo la seguridad de que nuestro tráfico esta cifrado. Si se observa la parte inferior de la figura 27 vemos que muestra un candado, esto es debido a que se ha creado una Certification Authority (CA) que es la entidad de confianza encargada de firmar certificados (CSR).

Al entrar en un sitio seguro, se muestra la siguiente información:

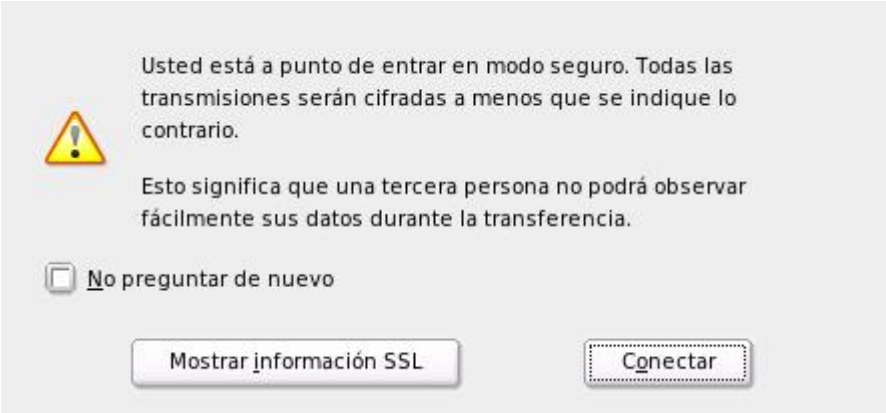

Figura 29: Advertencia de seguridad SSL

Al conectar con el servidor seguro, se ve como en el navegador añade a “http” una “s”, es decir, queda “https” lo que indica que se conecta por el puerto 443 que es el puerto de seguridad de SSL.

El certificado digital que se muestra, es el que se ha creado, si se pincha sobre el candado que sale en la esquina inferior derecha vemos dicho certificado:

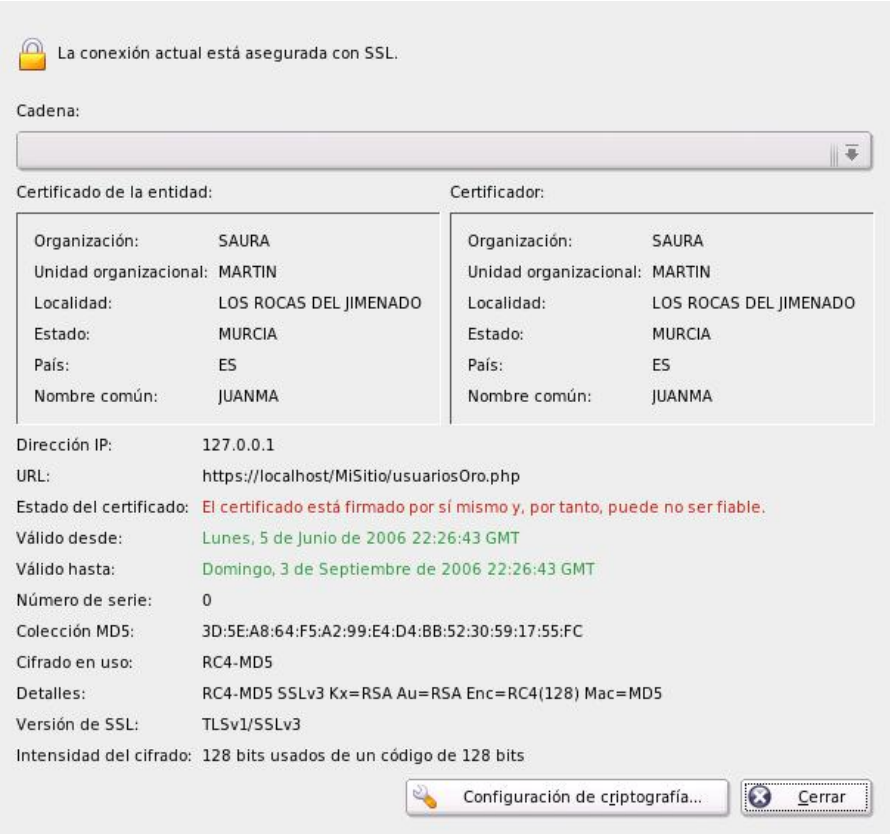

Figura 30: Certificado SSL.

Como se ve en la figura 29 dice que el certificado puede no ser fiable, esto es debido a que este certificado ha sido creado y firmado por la misma persona, podemos generar un certificado real con un tiempo de vida limitado como Thawte o Verisign, pero debido a esta limitación de tiempo se ha decidido hacerlo y ponerle el tiempo que se crea conveniente.

Entre las opciones que se muestran si se pincha en *iniciar sesión* se muestra la siguiente imagen:

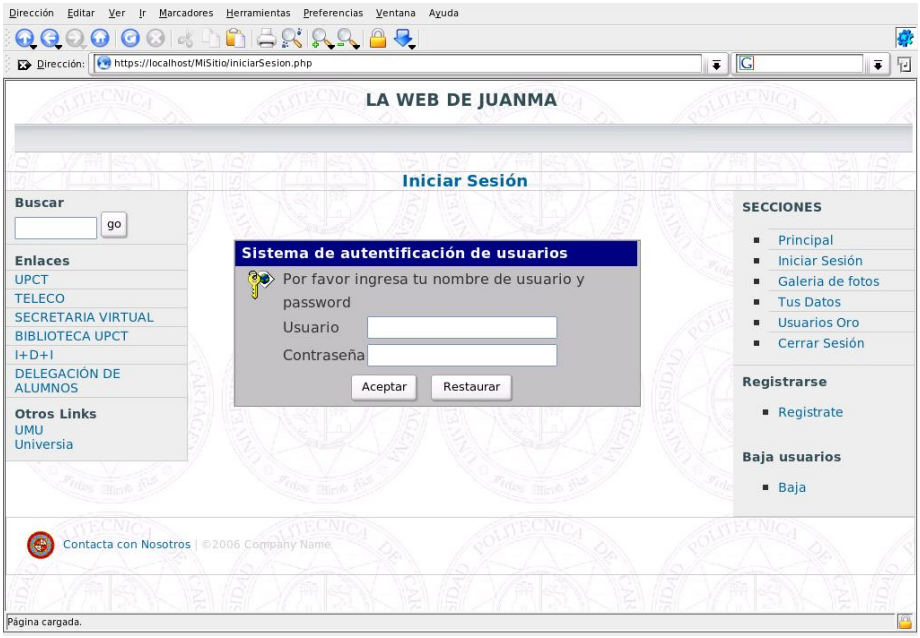

Figura 31: Sección: Iniciar Sesión.

Se observa como pide un usuario y una contraseña, para ello se ha creado un formulario como el que vemos en la figura de arriba, en el cual se introduce el usuario y el password y al aceptar se conecta a la base de datos y comprueba si el usuario esta registrado y si es así se inicia sesión para poder navegar por la página bajo criptografía SSL, la página que se muestra es la siguiente:

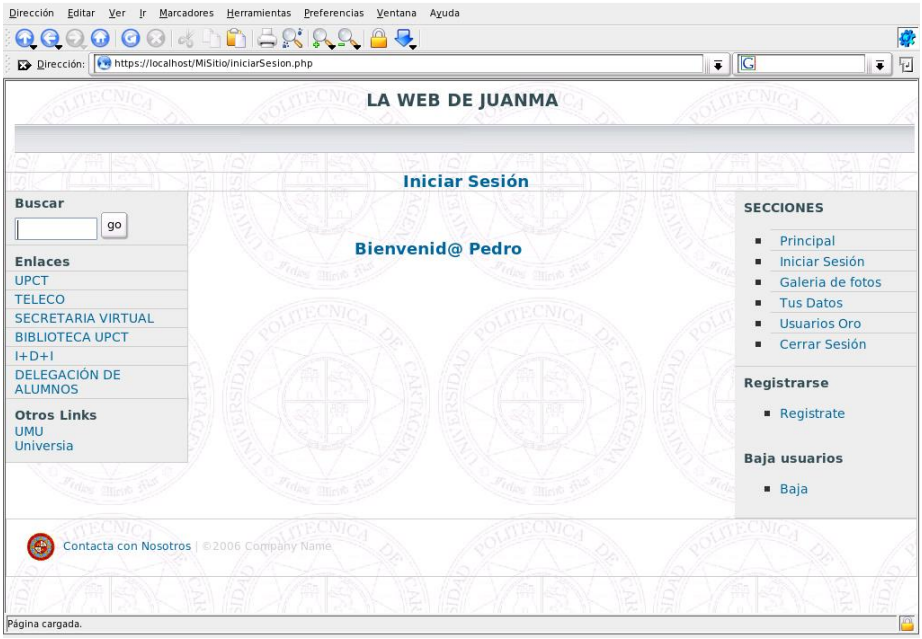

Figura 32: Usuario registrado.

En la derecha de la página vemos que hay una sección que es *Usuarios Oro*, esta sección tiene de particular que los usuarios que pueden acceder a este sitio son administrados por el administrador de la página, es decir, es el administrador quien los da de alta y les asigna una contraseña a cada usuario, como podemos ver el usuario *"Pedro"* no tiene los privilegios para acceder a esta sección ya que él se registró pero el administrador no le ha otorgado los permisos para esta sección:

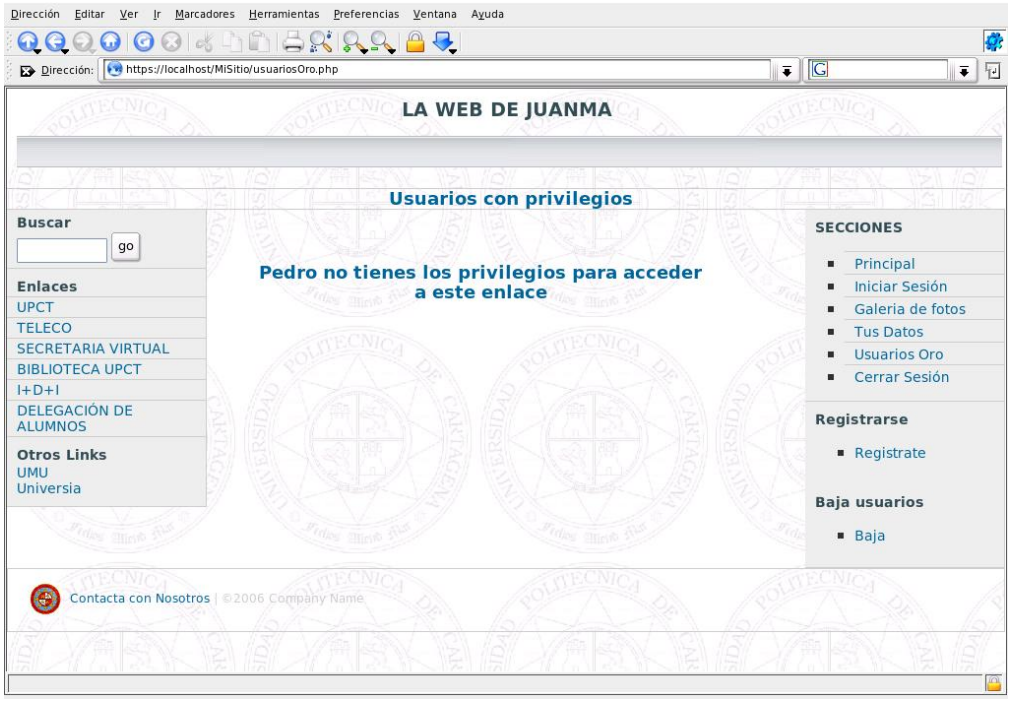

Figura 33: Usuario sin privilegios.

Sin embargo si accede un usuario el cual le ha sido asignada una contraseña por el administrador, si que tiene acceso:

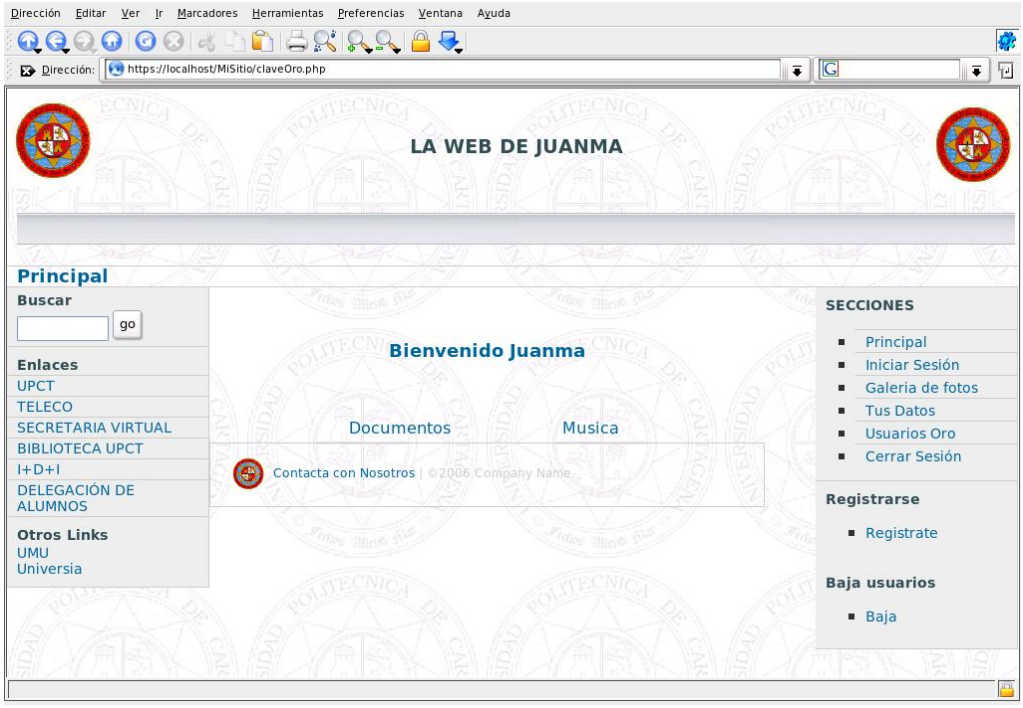

Figura 34: Usuario con privilegios

Se comprueba como el usuario *Juanma* si que tiene todos los permisos para acceder a este sitio y poder ver los documentos, archivos… que allí se encuentren. Además para acceder a dichos documentos he autentificado la carpeta donde estos se encuentran y aún así aunque seas usuario privilegiado si el administrador no te concede dicha contraseña no podemos acceder a ellos:

Figura 35: Carpeta del servidor apache autentificada.

Si no se conoce el usuario y contraseña, no podemos acceder a este sitio y el mensaje de error que nos muestra es el siguiente:

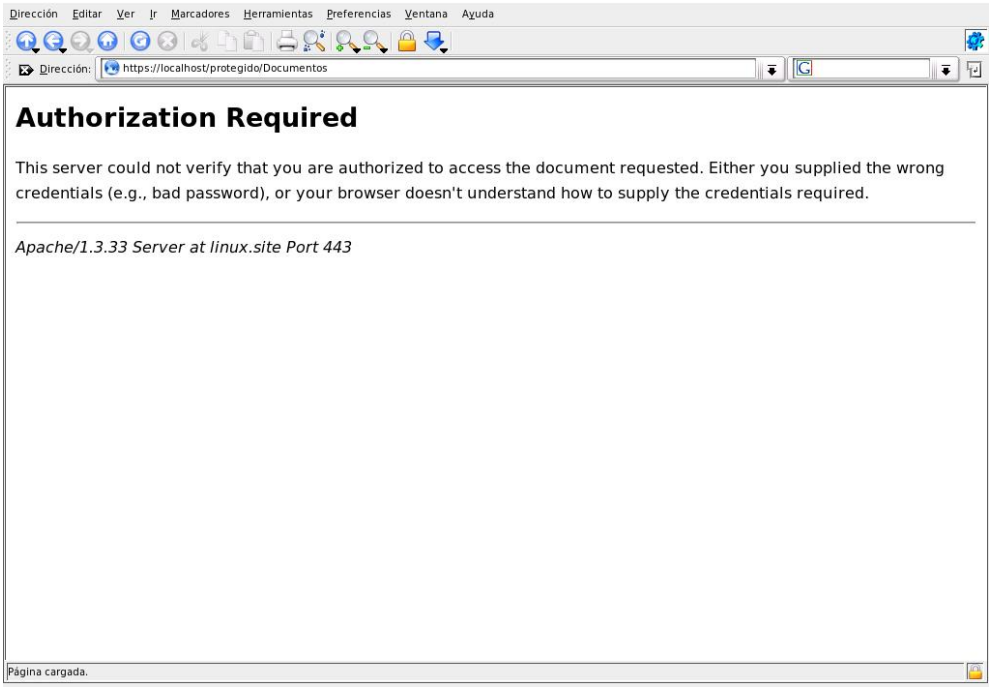

Figura 36: Autentificación requerida.

CONCLUSIONES Y LÍNEAS FUTURAS DE TRABAJO

Este proyecto final de carrera nació con el objetivo de hacer un estudio de seguridad a páginas Web. A nivel teórico, a lo largo del documento, se han revisado la mayoría de las consideraciones que han de tenerse en cuenta en el proceso de diseño de una página Web segura, desde problemas que afectan al servidor apache hasta los pasos que hay que seguir para una instalación satisfactoria. Junto con éstas, se ha continuado exponiendo las características del protocolo SSL y diferentes maneras de guardar las contraseñas, como por ejemplo mediante la base de datos MySQL.

En este proyecto se han revisado las bases más importantes de la seguridad en entornos Web, evidentemente, muchas cosas se han quedado en el tintero, y otras muchas no han sido comentadas con la profundidad que sin duda merecen. Se han intentado ofrecer ejemplos aplicados a entornos donde la seguridad es clave, haciendo uso de un pequeño ejemplo que se ha programado para aplicar dicha seguridad.

Para solucionar el problema, como se ha comentado a lo largo del proyecto, existen dos soluciones que todos deberían intentar aplicar: en primer lugar la concienciación de los problemas que nos pueden acarrear los fallos de seguridad. Tras la concienciación, es necesaria una formación adecuada a cada tipo de persona (evidentemente no podemos exigir los mismos conocimientos a un administrador responsable de varias máquinas que a un usuario que sólo conecta al sistema) no es necesario convertirse en un experto, simplemente hay que leer un poco y conocer unas normas básicas. Con estos dos pasos seguramente no se pararán a todos los piratas que nos intenten atacar, pero sí a la gran mayoría de ellos, que es lo que realmente interesa en el mundo de la seguridad. Por esto este proyecto lo puede aplicar cualquier usuario a su propio servidor, solo basta con seguir los pasos que se han comentado en los capítulos anteriores y tendremos un servidor seguro.

Sin embargo en la segunda parte del documento, desaparece este componente teórico para dar protagonismo a lo que ha supuesto el mayor esfuerzo realizado, la implementación de una página Web con estos sistemas de seguridad. Como se aprecia en los tres últimos capítulos, estos se resumen en diseñar, buscar, instalar y configurar los programas básicos encargados de mostrar la página: servidor apache, base de datos MySQL y lenguaje programación PHP.

Poner en funcionamiento un sistema de seguridad no es un proceso sencillo. Parte de las consideraciones van surgiendo a medida que se va desarrollando el proyecto, de forma que aparecen modificaciones que varían el diseño inicial. De hecho una de las modificaciones, entre otras, fue cambiar de sistema operativo, cuya necesidad queda justificada en el capítulo 5, otra modificación a destacar fue la necesidad de utilizar una base de datos para guardar la información del usuario… todo ello por cuestiones de seguridad. La principal función de este proyecto es que ofrece a cualquier técnico o usuario con nivel medio de programación Web, poder instalar un servidor Web seguro y crear su propia página Web bajo el protocolo SSL.

En futuras versiones se podrían introducir nuevos lenguajes de programación como Java Web Start que está construido sobre la plataforma Java 2, que proporciona una amplia arquitectura de seguridad. Java Web Start le permite ejecutar aplicaciones basadas en Java

directamente desde la Web. La ventaja que nos proporciona este tipo de programación es que permite lanzar aplicaciones de forma transparente al usuario.

Sería interesante utilizar nuevos protocolos de seguridad a parte del mencionado SSL, probar su integridad con el lenguaje PHP y Apache y sino se adaptan bien buscar otro servidor con las mismas características que Apache y que proporcione como mínimo la misma seguridad.

También de podría crear un sistema de copias de seguridad, es decir que automáticamente cada día guarde una copia de la información de la base de datos, para tener la seguridad de que si un atacante consigue borrar los datos, no se pierdan todos y podamos restaurar nuestro sistema.

Por último cabe destacar el uso de tunneling IPSec en futuras versiones, aunque este protocolo este más orientado a redes, es decir, asegura que el medio por donde viaja la información este cifrado con una clave y ponga más difícil a un atacante la obtención de los datos que viajen a través de él, aunque en este proyecto los datos se han cifrado mediante SSL, pero así también se asegura el medio de transmisión mejor.

Apéndice A

FILTRADO DE INFORMACIÓN

El HTTP se ejecuta en el puerto 80, con tcp, y si es sólo para uso interno (una Intranet, o un mecanismo de control basado en www, para, digamos, un servidor cortafuegos) definitivamente habría que filtrarlo con el cortafuegos.

```
ipfwadm –I –a accept –P tcp –S 10.0.0.0/8 –D 0.0.0.0/0 80
ipfwadm –I –a accept –P tcp –S un.host.fiable –D 0.0.0.0/0 80
ipfwadm –I –a deny –P tcp –S 0.0.0.0/0 –D 0.0.0.0/0 80
o en ipchains:
ipchains –A input –p all –j ACCEPT –s 10.0.0.0/8 –d 0.0.0.0/0 80
ipchains –A input –p all –j ACCEPT –s un.host.fiable –d 0.0.0.0/0 80
ipchains –A input –p all –j DENY –s 0.0.0.0/0 –d 0.0.0.0/0 80
```

El HTTPS se ejecuta en el puerto 443, con tcp, y si sólo es para uso interno también debería filtrarse con el cortafuegos:

```
ipfwadm –I –a accept –P tcp –S 10.0.0.0/8 –D 0.0.0.0/0 443
ipfwadm –I –a accept –P tcp –S un.host.fiable –D 0.0.0.0/0 443
ipfwadm –I –a deny –P tcp –S 0.0.0.0/0 –D 0.0.0.0/0 443
o en ipchains:
ipchains –A input –p all –j ACCEPT –s 10.0.0.0/8 –d 0.0.0.0/0 443
ipchains –A input –p all –j ACCEPT –s un.host.fiable –d 0.0.0.0/0
443
ipchains –A input –p all –j DENY –s 0.0.0.0/0 –d 0.0.0.0/0 443
```

Apéndice B

CÓDIGO PHP: PÁGINA PRINCIPAL

```
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-
1" />
    <title>PFC: Implementación de seguridad Web</title>
    <link rel="stylesheet" href="estilos.css" type="text/css" />
</head>
<body background="Imagenes/fondo.jpg">
    <div id="masthead">
        <table width="100%">
            <td width="98" align="left">
                <img src="Imagenes/upct.jpg" width="85" height="85"/>
            </td>
            <td width="840" align="center">
                <h1 align="center">LA WEB DE JUANMA</h1>
            </td>
            <td width="86" align="right">
                <img src="Imagenes/upct.jpg" width="85" height="85" align
="right"/>
            </td>
        </table>
        <script language="JAVASCRIPT">
            var refresH=1200;
            var numMensaje=0;
            msgS=new Array();
            msgS[1]="Bienvenido a la wed de Juan Manuel";
            msgS[2]="Universidad Politécnica de Cartagena";
            msgS[3]="PFC titulado: 'Portales webs seguros'";
            msgS[4]="PFC dirigido por: Pedro Sánchez Palma";
            msgS[5]=" ";

            function refrescar()
            {
                numMensaje += 1;
                if (numMensaje == msgS.length)
                    numMensaje=1;
                window.status=msgS[numMensaje];
                refreshC=setTimeout("refrescar();",refresH);
            }
            refrescar();
        </script>

        <div id="globalNav">
            <p><img alt="" src="Imagenes/glbnav_background.gif" height="3
2" width="100%" id="gnl" /></p>
        </div>
        <h2 id="pageName">Principal</h2>
    </div>
    <div id="navBar">
        <div id="search">
            <form action="http://www.google.es">
                <label>Buscar</label>
                <input type="text" size="10" />
                <input type="submit" value="go" />
            </form>
        </div>
```

```
        <div id="sectionLinks">
            <h3>Enlaces</h3>
            <ul>
                <li><a href="http://www.upct.es">UPCT</a></li>
                <li><a href="http://www.teleco.upct.es">TELECO</a></li>
                <li><a href="http://redcampus.upct.es/redcampus/acceso.ht
m">SECRETARIA VIRTUAL</a></li>
                <li><a href="http://www.bib.upct.es/">BIBLIOTECA UPCT</a>
</li>
                <li><a href="http://www.upct.es/contenido/idi/index_idi.p
hp/">I+D+I</a></li>
                <li><a href="http://www.alumnosteleco.upct.es/">DELEGACIÓ
N DE ALUMNOS</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Otros Links</h3>
            <ul>
                <li><a href="http://www.umu.es">UMU</a></li>
                <li><a href="http://www.universia.es">Universia</a></li>
            </ul>
        </div>
    </div>

    <!--end navBar1 div -->
    <div id="navBar1">
        <div id="sectionLinks">
            <h3>SECCIONES</h3>
            <ul>
                <li><a href="index.php">Principal</a></li>
                <li><a href="https://localhost/MiSitio/iniciarSesion.php"
>Iniciar Sesión</a></li>
                <li><a href="https://localhost/MiSitio/galeriaFotos.php">
Galeria de fotos</a></li>
                <li><a href="https://localhost/MiSitio/tusDatos.php">Tus
Datos</a></li>
                <li><a href="https://localhost/MiSitio/usuariosOro.php">U
suarios Oro</a></li>
                <li><a href="https://localhost/MiSitio/cerrarSesion.php">
Cerrar Sesión</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Registrarse</h3>
            <ul>
                <li><a href="registrarse.php">Registrate</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Baja usuarios</h3>
            <ul>
                <li><a href="bajaUsuarios.php">Baja</a></li>
            </ul>
        </div>
    </div>

    <!--end headlines -->
    <div id="content">
        <div class="feature">
            <img src="Imagenes/Cuartel_Antiguones.jpg" alt="" width="250"
 height="250" />
```

```
            <h3>Escuela técnica superior de ingenieria de Telecomunicació
n</h3>
            <p>
                Página realizada como proyecto fin de carrera.
            </p>
        </div>
        <div class="story">
            <h3>PFC</h3>
            <p>
                Esta página Web ha sido desarrollada por Juan Manuel Saur
a Martín, estudiante
                de Ingeniería técnica de Telecomunicación especialidad Te
lemática en la
                universidad politécnica de Cartagena. Esta página se basa
 en un estudio de
                seguridad de portales Web cuando Apache, php y MySQL está
n detrás.
            </p>
            <p>
                Este proyecto ha sido dirigido por Dr. D. Pedro Sanchez P
álma, profesor
                de la UPCT del departamento TIC.
            </p>
        </div>
    </div>
    <!--end content -->
    <div id="siteInfo">
        <img src="Imagenes/upct.jpg" width="35" height="35" />
        <a href="/protegido/email.doc">Contacta con Nosotros</a> | &copy;
2006 Company Name
    </div>
    <br />
</body>
</html>
```

Apéndice C

CÓDIGO PHP: INICIAR SESIÓN

```
<?
    session_start();
?>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-
1" />
    <title>PFC: Implementación de seguridad Web</title>
    <link rel="stylesheet" href="estilos.css" type="text/css" />
</head>
<body background="Imagenes/fondo.jpg">
    <div id="masthead">
        <h1 align="center" id="siteName">LA WEB DE JUANMA </h1>
        <div id="globalNav">
            <p><img alt="" src="Imagenes/glbnav_background.gif" height="3
2" width="100%" id="gnl" /></p>
        </div>
        <h2 align="center" id="pageName">Iniciar Sesión</h2>
    </div>
    <div id="navBar">
        <div id="search">
            <form action="http://www.google.es">
                <label>Buscar</label>
                <input type="text" size="10" />
                <input type="submit" value="go" />
            </form>
        </div>
        <div id="sectionLinks">
            <h3>Enlaces</h3>
            <ul>
                <li><a href="http://www.upct.es">UPCT</a></li>
                <li><a href="http://www.teleco.upct.es">TELECO</a></li>
                <li><a href="http://redcampus.upct.es/redcampus/acceso.ht
m">SECRETARIA VIRTUAL</a></li>
                <li><a href="http://www.bib.upct.es/">BIBLIOTECA UPCT</a>
</li>
                <li><a href="http://www.upct.es/contenido/idi/index_idi.p
hp/">I+D+I</a></li>
                <li><a href="http://www.alumnosteleco.upct.es/">DELEGACIÓ
N DE ALUMNOS</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Otros Links</h3>
            <ul>
                <li><a href="http://www.umu.es">UMU</a></li>
                <li><a href="http://www.universia.es">Universia</a></li>
            </ul>
        </div>
    </div>
    <!--end navBar1 div -->
    <div id="navBar1">
        <div id="sectionLinks">
            <h3>SECCIONES</h3>
            <ul>
                <li><a href="index.php">Principal</a></li>
```

```
                <li><a href="iniciarSesion.php">Iniciar Sesión</a></li>
                <li><a href="galeriaFotos.php">Galeria de fotos</a></li>
                <li><a href="tusDatos.php">Tus Datos</a></li>
                <li><a href="usuariosOro.php">Usuarios Oro</a></li>
                <li><a href="cerrarSesion.php">Cerrar Sesión</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Registrarse</h3>
            <ul>
                <li><a href="registrarse.php">Registrate</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Baja usuarios</h3>
            <ul>
                <li><a href="bajaUsuarios.php">Baja</a></li>
            </ul>
        </div>
    </div>
    <!--end headlines -->
    <div id="content">
        <?
            $NombreBD = "datos_usuarios";
            $Servidor = "localhost";
            $Usuario = "root";
            $password = "45612696";

            $IdConexion = mysql_connect($Servidor, $Usuario, $password);

            mysql_select_db($NombreBD, $IdConexion) or die ("No se encont
ro la base de datos");

            //Esta es la consulta que queremos hacer.
            $consulta = 'SELECT USUARIO,DIRECCION,EMAIL,IDUSUARIO,SEXO FR
OM usuarios;';

            //mysql_query nos devuelve el identificador de la consulta.
            $IdConsulta = mysql_query($consulta, $IdConexion);

            //Nos devuelve el número de filas;
            $NFilas = mysql_num_rows($IdConsulta);

            for( $i=0; $i<$NFilas; $i++ )
            {
                //Para recuperar los datos.
                $resultadoConsulta = mysql_fetch_array($IdConsulta);

                if( ( $resultadoConsulta[ "USUARIO" ] == $_POST[ 'usuario
' ] ) && ( $resultadoConsulta[ "IDUSUARIO" ] == md5( $_POST[ 'IDUsuario'
] ) ) )
                {
                    $_SESSION[ 'nombre' ] = $_POST[ 'usuario' ];
                    $_SESSION[ 'id' ] = $_POST[ 'IDUsuario' ];

                    $correcto = 1;
                    ?>
                        <center>
                            <br><br><br>
                            <h2>Bienvenid@  <? echo $_POST[ 'usuario' ] ?
></h2>
```

```
                    </center>
                <?
                break;
            }
            elseif( ( $resultadoConsulta[ "USUARIO" ] == $_SESSION[ '
nombre' ] ) && ( $resultadoConsulta[ "IDUSUARIO" ] == md5( $_SESSION[ 'id
' ] ) ) )
            {
                $correcto = 1;
                ?>
                    <center>
                        <br><br><br>
                        <h2>Sesion establecida</h2>
                    </center>
                <?
                break;
            }
        }

        if( $correcto != 1 )
        {
            echo
                "<form action=\"iniciarSesion.php\" method=\"POST\">
                    <br><br><br>
                    <table align=\"center\" border=\"1\" cellspacing=
\"0\" cellpadding=\"0\" width=\"450\">
                        <tr bgcolor=\"#C6C3C6\">
                            <td>
                                <table align=\"center\" width=\"450\"
 border=\"0\" cellspacing=\"0\" cellpadding=\"1\">
                                    <tr bgcolor=\"#400080\">
                                        <td height=\"20\" class=\"td1
\" bgcolor=\"#000084\">
                                            <b>
                                                <font color=\"#FFFFFF
\">
                                                     Sistema de
autentificación de usuarios
                                                </font>
                                            </b>
                                        </td>
                                    </tr>
                                    <tr align=\"center\">
                                        <td colspan=\"2\">
                                            <table border=\"0\" cells
pacing=\"0\" cellpadding=\"1\" width=\"100%\">
                                                <tr>
                                                    <td rowspan=\"3\"
 valign=\"top\" align=\"right\" width=\"12%\">
                                                        <img src=\"Im
agenes/key.gif\" width=\"35\" height=\"40\">
                                                    </td>
                                                    <td colspan=\"2\"
 class=\"td1\" height=\"42\">
                                                        Por favor ing
resa tu nombre de usuario y  password
                                                    </td>
                                                </tr>
                                                <tr>
                                                    <td width=\"18%\"
 class=\"td1\">
```

```
                                                                    Usuario
                                                                </td>
                                                                <td width=\"70%\"
>
                                                                    <input type=\
"text\" class=\"campo\" name=\"usuario\" size=\"25\">
                                                                </td>
                                                            </tr>
                                                            <tr>
                                                                <td height=\"35\"
 width=\"18%\" class=\"td1\">
                                                                    Contraseña
                                                                </td>
                                                                <td height=\"35\"
 width=\"70%\">
                                                                    <input type=\
"password\" class=\"campo\" name=\"IDUsuario\" size=\"25\">
                                                                </td>
                                                            </tr>
                                                        </table>
                                                        <table width=\"100%\" bor
der=\"0\" cellspacing=\"0\" cellpadding=\"1\">
                                                            <tr>
                                                                <td align=\"right
\" width=\"47%\">
                                                                    <input class=
\"boton\" type=\"submit\" value=\"Aceptar\">
                                                                </td>
                                                                <td align=\"right
\" width=\"1%\">
                                                                     
                                                                </td>
                                                                <td width=\"52%\"
>
                                                                    <input class=
\"boton\" type=\"reset\" value=\"Restaurar\">
                                                                </td>
                                                            </tr>
                                                        </table>
                                                    </td>
                                                </tr>
                                            </table>
                                        </td>
                                    </tr>
                                </table>
                            </form>";
                }
            ?>
            </center>
        </div>
        <!--end content -->
        <div id="siteInfo">
            <img src="Imagenes/upct.jpg" width="35" height="35" />
            <a href="/protegido/email.doc">Contacta con Nosotros</a> | &copy;
2006 Company Name
        </div>
        <br />
</body>
</html>
```

Apéndice D

CÓDIGO PHP: REGISTRARSE

```
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-
1" />
    <title>PFC: Implementación de seguridad Web</title>
    <link rel="stylesheet" href="estilos.css" type="text/css" />
</head>
<body background="Imagenes/fondo.jpg">
    <div id="masthead">
        <h1 align="center" id="siteName">LA WEB DE JUANMA </h1>
        <div id="globalNav">
            <p><img alt="" src="Imagenes/glbnav_background.gif" height="3
2" width="100%" id="gnl" /></p>
        </div>
        <h2 align="center" id="pageName">Registrarse</h2>
    </div>
    <div id="navBar">
        <div id="search">
            <form action="http://www.google.es">
                <label>Buscar</label>
                <input type="text" size="10" />
                <input type="submit" value="go" />
            </form>
        </div>
        <div id="sectionLinks">
            <h3>Enlaces</h3>
            <ul>
                <li><a href="http://www.upct.es">UPCT</a></li>
                <li><a href="http://www.teleco.upct.es">TELECO</a></li>
                <li><a href="http://redcampus.upct.es/redcampus/acceso.ht
m">SECRETARIA VIRTUAL</a></li>
                <li><a href="http://www.bib.upct.es/">BIBLIOTECA UPCT</a>
</li>
                <li><a href="http://www.upct.es/contenido/idi/index_idi.p
hp/">I+D+I</a></li>
                <li><a href="http://www.alumnosteleco.upct.es/">DELEGACIÓ
N DE ALUMNOS</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Otros Links</h3>
            <ul>
                <li><a href="http://www.umu.es">UMU</a></li>
                <li><a href="http://www.universia.es">Universia</a></li>
            </ul>
        </div>
    </div>
    <!--end navBar1 div -->
    <div id="navBar1">
        <div id="sectionLinks">
            <h3>SECCIONES</h3>
            <ul>
                <li><a href="index.php">Principal</a></li>
                <li><a href="iniciarSesion.php">Iniciar Sesión</a></li>
                <li><a href="galeriaFotos.php">Galeria de fotos</a></li>
                <li><a href="tusDatos.php">Tus Datos</a></li>
```

```
                <li><a href="usuariosOro.php">Usuarios Oro</a></li>
                <li><a href="cerrarSesion.php">Cerrar Sesión</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Registrarse</h3>
            <ul>
                <li><a href="registrarse.php">Registrate</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Baja usuarios</h3>
            <ul>
                <li><a href="bajaUsuarios.php">Baja</a></li>
            </ul>
        </div>
    </div>
    <!--end headlines -->
    <div id="content">
        <br><br><br>
        <h2 align="center">Inscripción de usuarios</h2>
        <form name="Inscripcion" method="post" action="confirmacionDatos.
php" enctype="multipart/form-data">
            <table width="400" border="1" align="center" cellpadding="5"
cellspacing="0" bgcolor="#CCCCCC">
                <tr>
                    <td width="47%">
                        Nombre
                    </td>
                    <td colspan="2">
                        <input type="text" name="usuario" size="25" maxle
ngth="25"><br><br>
                    </td>
                </tr>
                <tr>
                    <td width="47%">
                        Dirección
                    </td>
                    <td height="57" colspan="2">
                        <textarea name="direccion" cols="26" rows="4"></t
extarea><br><br>
                    </td>
                </tr>
                <tr>
                    <td width="47%">
                        Correo electrónico
                    </td>
                    <td height="2" colspan="2">
                        <input type="text" name="email" size="25" maxleng
ht="50"><br><br>
                    </td>
                </tr>
                <tr>
                    <td width="47%">
                        Contraseña
                    </td>
                    <td height="2">
                        <input type="password" name="IDUsuario" size=="25
" maxlenght="15"><br><br>
                    </td>
                </tr>
```

```
                <tr>
                    <td width="47%">
                        Confirmación de contraseña
                    </td>
                    <td height="2">
                        <input type="password" name="CIDUsuario" size=="2
5" maxlenght="15"><br><br>
                    </td>
                </tr>
                <tr>
                    <td width="47%">
                        Sexo
                    </td>
                    <td height="2">
                        <input type="radio" name="genero" value="Hombre">
                            Hombre
                        <input type="radio" name="genero" value="Mujer">
                            Mujer
                    </td>
                </tr>
                <tr>
                    <td colspan="3" align="center">
                        <input type="submit" value="Enviar">
                        <input type="reset" value="Borrar Datos">
                    </td>
                </tr>
            </table>
        </form>
    </div>
    <!--end content -->
    <div id="siteInfo">
        <img src="Imagenes/upct.jpg" width="35" height="35" />
        <a href="/protegido/email.doc">Contacta con Nosotros</a> | &copy;
2006 Company Name
    </div>
    <br />
</body>
</html>
```

Apéndice E

CÓDIGO PHP: TUS DATOS

```
<?
    session_start();
?>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-
1" />
    <title>PFC: Implementación de seguridad Web</title>
    <link rel="stylesheet" href="estilos.css" type="text/css" />
</head>
<body background="Imagenes/fondo.jpg">
    <div id="masthead">
        <h1 align="center" id="siteName">LA WEB DE JUANMA </h1>
        <div id="globalNav">
            <p><img alt="" src="Imagenes/glbnav_background.gif" height="3
2" width="100%" id="gnl" /></p>
        </div>
        <h2 align="center" id="pageName">Mostrar Datos</h2>
    </div>
    <div id="navBar">
        <div id="search">
            <form action="http://www.google.es">
                <label>Buscar</label>
                <input type="text" size="10" />
                <input type="submit" value="go" />
            </form>
        </div>
        <div id="sectionLinks">
            <h3>Enlaces</h3>
            <ul>
                <li><a href="http://www.upct.es">UPCT</a></li>
                <li><a href="http://www.teleco.upct.es">TELECO</a></li>
                <li><a href="http://redcampus.upct.es/redcampus/acceso.ht
m">SECRETARIA VIRTUAL</a></li>
                <li><a href="http://www.bib.upct.es/">BIBLIOTECA UPCT</a>
</li>
                <li><a href="http://www.upct.es/contenido/idi/index_idi.p
hp/">I+D+I</a></li>
                <li><a href="http://www.alumnosteleco.upct.es/">DELEGACIÓ
N DE ALUMNOS</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Otros Links</h3>
            <ul>
                <li><a href="http://www.umu.es">UMU</a></li>
                <li><a href="http://www.universia.es">Universia</a></li>
            </ul>
        </div>
    </div>
    <!--end navBar1 div -->
    <div id="navBar1">
        <div id="sectionLinks">
            <h3>SECCIONES</h3>
            <ul>
                <li><a href="index.php">Principal</a></li>
```

```
                <li><a href="iniciarSesion.php">Iniciar Sesión</a></li>
                <li><a href="galeriaFotos.php">Galeria de fotos</a></li>
                <li><a href="tusDatos.php">Tus Datos</a></li>
                <li><a href="usuariosOro.php">Usuarios Oro</a></li>
                <li><a href="cerrarSesion.php">Cerrar Sesión</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Registrarse</h3>
            <ul>
                <li><a href="registrarse.php">Registrate</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Baja usuarios</h3>
            <ul>
                <li><a href="bajaUsuarios.php">Baja</a></li>
            </ul>
        </div>
    </div>
    <!--end headlines -->
    <div id="content">
        <?
            $NombreBD = "datos_usuarios";
            $Servidor = "localhost";
            $Usuario = "root";
            $password = "45612696";

            $IdConexion = mysql_connect($Servidor, $Usuario, $password);

            mysql_select_db($NombreBD, $IdConexion) or die ("No se encont
ro la base de datos");

            //Esta es la consulta que queremos hacer.
            $consulta = 'SELECT USUARIO,DIRECCION,EMAIL,IDUSUARIO,SEXO FR
OM usuarios;';

            //mysql_query nos devuelve el identificador de la consulta.
            $IdConsulta = mysql_query($consulta, $IdConexion);

            //Nos devuelve el número de filas;
            $NFilas = mysql_num_rows($IdConsulta);

            for( $i=0; $i<$NFilas; $i++ )
            {
                //Para recuperar los datos.
                $resultadoConsulta = mysql_fetch_array($IdConsulta);

                if( ( $resultadoConsulta[ "USUARIO" ] == $_POST[ 'usuario
' ] ) && ( $resultadoConsulta[ "IDUSUARIO" ] == md5( $_POST[ 'IDUsuario'
] ) ) )
                {
                    $_SESSION[ 'nombre' ] = $_POST[ 'usuario' ];
                    $_SESSION[ 'id' ] = $_POST[ 'IDUsuario' ];

                    $correcto = 1;
                    print("<br><br><br>");
                    ?>
                        <center>
                            <h2>Los datos guardados en la base de datos s
on:</h2>
```

```
                            <TABLE BORDER=1 BGCOLOR=#7BA9E8 CELLPADDING=4
 CELLSPACING=3>

                            <?
                            print("<tr><td>Nombre: </td>");
                            print("<td>".$resultadoConsulta[ "USUARIO" ].
"</TD></tr>");
                            print("<tr><td>Dirección: ");
                            print("<td>".$resultadoConsulta[ "DIRECCION"
]."</td></tr>");
                            print("<tr><td>Contraseña: </td>");
                            print("<td>".$_SESSION[ 'id' ]."</td></tr>");
                            print("<tr><td>E-mail: ");
                            print("<td>".$resultadoConsulta[ "EMAIL" ]."<
/td></tr>");
                            print("<tr><td>Sexo: ");
                            print("<td>".$resultadoConsulta[ "SEXO" ]."</
td></tr>");
                            break;
                            ?>
                        </center>
                    <?
                }
                elseif( ( $resultadoConsulta[ "USUARIO" ] == $_SESSION[ '
nombre' ] ) && ( $resultadoConsulta[ "IDUSUARIO" ] == md5( $_SESSION[ 'id
' ] ) ) )
                {
                    $correcto = 1;
                    print("<br><br><br>");
                    ?>
                        <center>
                            <h2>Los datos guardados en la base de datos s
on:</h2>
                            <TABLE BORDER=1 BGCOLOR=#7BA9E8 CELLPADDING=4
 CELLSPACING=3>

                            <?
                            print("<tr><td>Nombre: </td>");
                            print("<td>".$resultadoConsulta[ "USUARIO" ].
"</TD></tr>");
                            print("<tr><td>Dirección: ");
                            print("<td>".$resultadoConsulta[ "DIRECCION"
]."</td></tr>");
                            print("<tr><td>Contraseña: </td>");
                            print("<td>".$_SESSION[ 'id' ]."</td></tr>");
                            print("<tr><td>E-mail: ");
                            print("<td>".$resultadoConsulta[ "EMAIL" ]."<
/td></tr>");
                            print("<tr><td>Sexo: ");
                            print("<td>".$resultadoConsulta[ "SEXO" ]."</
td></tr>");
                            ?>
                            </table>
                            <?
                            break;
                            ?>
                        </center>
                    <?
                }
            }
```

```
            if( $correcto != 1 )
            {
                echo
                    "<form action=\"tusDatos.php\" method=\"POST\">
                        <br><br><br>
                        <table align=\"center\" border=\"1\" cellspacing=
\"0\" cellpadding=\"0\" width=\"450\">
                            <tr bgcolor=\"#C6C3C6\">
                                <td>
                                    <table align=\"center\" width=\"450\"
 border=\"0\" cellspacing=\"0\" cellpadding=\"1\">
                                        <tr bgcolor=\"#400080\">
                                            <td height=\"20\" class=\"td1
\" bgcolor=\"#000084\">
                                                <b>
                                                    <font color=\"#FFFFFF
\">
                                                         Sistema de
autentificación de usuarios
                                                    </font>
                                                </b>
                                            </td>
                                        </tr>
                                        <tr align=\"center\">
                                            <td colspan=\"2\">
                                                <table border=\"0\" cells
pacing=\"0\" cellpadding=\"1\" width=\"100%\">
                                                    <tr>
                                                        <td rowspan=\"3\"
 valign=\"top\" align=\"right\" width=\"12%\">
                                                            <img src=\"Im
agenes/key.gif\" width=\"35\" height=\"40\">
                                                        </td>
                                                        <td colspan=\"2\"
 class=\"td1\" height=\"42\">
                                                            Por favor ing
resa tu nombre de usuario y  password
                                                        </td>
                                                    </tr>
                                                    <tr>
                                                        <td width=\"18%\"
 class=\"td1\">
                                                            Usuario
                                                        </td>
                                                        <td width=\"70%\"
>
                                                            <input type=\
"text\" class=\"campo\" name=\"usuario\" size=\"25\">
                                                        </td>
                                                    </tr>
                                                    <tr>
                                                        <td height=\"35\"
 width=\"18%\" class=\"td1\">
                                                            Contraseña
                                                        </td>
                                                        <td height=\"35\"
 width=\"70%\">
                                                            <input type=\
"password\" class=\"campo\" name=\"IDUsuario\" size=\"25\">
                                                        </td>
                                                    </tr>
```

```
                                                    </table>
                                                    <table width=\"100%\" bor
der=\"0\" cellspacing=\"0\" cellpadding=\"1\">
                                                        <tr>
                                                            <td align=\"right
\" width=\"47%\">
                                                                <input class=
\"boton\" type=\"submit\" value=\"Aceptar\">
                                                            </td>
                                                            <td align=\"right
\" width=\"1%\">
                                                                 
                                                            </td>
                                                            <td width=\"52%\"
>
                                                                <input class=
\"boton\" type=\"reset\" value=\"Restaurar\">
                                                            </td>
                                                        </tr>
                                                    </table>
                                                </td>
                                            </tr>
                                        </table>
                                    </td>
                                </tr>
                            </table>
                        </form>";
                }
            ?>
        </div>
        <!--end content -->
        <div align="left" id="siteInfo">
            <img src="Imagenes/upct.jpg" width="35" height="35" />
            <a href="/protegido/email.doc">Contacta con Nosotros</a> | &copy;
2006 Company Name
        </div>
        <br />
</body>
</html>
```

Apéndice F

CÓDIGO PHP: USUARIOS ORO

```
<?
    session_start();
?>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-
1" />
    <title>PFC: Implementación de seguridad Web</title>
    <link rel="stylesheet" href="estilos.css" type="text/css" />
</head>
<body background="Imagenes/fondo.jpg">
    <div id="masthead">
        <h1 align="center" id="siteName">LA WEB DE JUANMA </h1>
        <div id="globalNav">
            <p><img alt="" src="Imagenes/glbnav_background.gif" height="3
2" width="100%" id="gnl" /></p>
        </div>
        <h2 align="center" id="pageName">Usuarios con privilegios</h2>
    </div>
    <div id="navBar">
        <div id="search">
            <form action="http://www.google.es">
                <label>Buscar</label>
                <input type="text" size="10" />
                <input type="submit" value="go" />
            </form>
        </div>
        <div id="sectionLinks">
            <h3>Enlaces</h3>
            <ul>
                <li><a href="http://www.upct.es">UPCT</a></li>
                <li><a href="http://www.teleco.upct.es">TELECO</a></li>
                <li><a href="http://redcampus.upct.es/redcampus/acceso.ht
m">SECRETARIA VIRTUAL</a></li>
                <li><a href="http://www.bib.upct.es/">BIBLIOTECA UPCT</a>
</li>
                <li><a href="http://www.upct.es/contenido/idi/index_idi.p
hp/">I+D+I</a></li>
                <li><a href="http://www.alumnosteleco.upct.es/">DELEGACIÓ
N DE ALUMNOS</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Otros Links</h3>
            <ul>
                <li><a href="http://www.umu.es">UMU</a></li>
                <li><a href="http://www.universia.es">Universia</a></li>
            </ul>
        </div>
    </div>
    <!--end navBar1 div -->
    <div id="navBar1">
        <div id="sectionLinks">
            <h3>SECCIONES</h3>
            <ul>
                <li><a href="index.php">Principal</a></li>
```

```
                <li><a href="iniciarSesion.php">Iniciar Sesión</a></li>
                <li><a href="galeriaFotos.php">Galeria de fotos</a></li>
                <li><a href="tusDatos.php">Tus Datos</a></li>
                <li><a href="usuariosOro.php">Usuarios Oro</a></li>
                <li><a href="cerrarSesion.php">Cerrar Sesión</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Registrarse</h3>
            <ul>
                <li><a href="registrarse.php">Registrate</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Baja usuarios</h3>
            <ul>
                <li><a href="bajaUsuarios.php">Baja</a></li>
            </ul>
        </div>
    </div>
    <!--end headlines -->
    <div id="content">
        <?
        if( $_SESSION['nombre'] == '' )
        {
            echo
                "<form action=\"claveOro.php\" method=\"POST\">
                    <br><br><br>
                    <table align=\"center\" border=\"1\" cellspacing=\"0\
" cellpadding=\"0\" width=\"450\">
                        <tr bgcolor=\"#C6C3C6\">
                            <td>
                                <table align=\"center\" width=\"450\" bor
der=\"0\" cellspacing=\"0\" cellpadding=\"1\">
                                    <tr bgcolor=\"#400080\">
                                        <td height=\"20\" class=\"td1\" b
gcolor=\"#000084\">
                                            <b>
                                                <font color=\"#FFFFFF\">
                                                     Sistema de aute
ntificación de usuarios
                                                </font>
                                            </b>
                                        </td>
                                    </tr>
                                    <tr align=\"center\">
                                        <td colspan=\"2\">
                                            <table border=\"0\" cellspaci
ng=\"0\" cellpadding=\"1\" width=\"100%\">
                                                <tr>
                                                    <td rowspan=\"3\" val
ign=\"top\" align=\"right\" width=\"12%\">
                                                        <img src=\"Imagen
es/key.gif\" width=\"35\" height=\"40\">
                                                    </td>
                                                    <td colspan=\"2\" cla
ss=\"td1\" height=\"42\">
                                                        Por favor ingresa
 tu nombre de usuario y  password
                                                    </td>
                                                </tr>
```

```
                                                    <tr>
                                                        <td width=\"18%\" cla
ss=\"td1\">
                                                            Usuario
                                                        </td>
                                                        <td width=\"70%\">
                                                            <input type=\"tex
t\" class=\"campo\" name=\"usuario\" size=\"25\">
                                                        </td>
                                                    </tr>
                                                    <tr>
                                                        <td height=\"35\" wid
th=\"18%\" class=\"td1\">
                                                            Contraseña
                                                        </td>
                                                        <td height=\"35\" wid
th=\"70%\">
                                                            <input type=\"pas
sword\" class=\"campo\" name=\"IDUsuario\" size=\"25\">
                                                        </td>
                                                    </tr>
                                                </table>
                                                <table width=\"100%\" border=
\"0\" cellspacing=\"0\" cellpadding=\"1\">
                                                    <tr>
                                                        <td align=\"right\" w
idth=\"47%\">
                                                            <input class=\"bo
ton\" type=\"submit\" value=\"Aceptar\">
                                                        </td>
                                                        <td align=\"right\" w
idth=\"1%\">
                                                             
                                                        </td>
                                                        <td width=\"52%\">
                                                            <input class=\"bo
ton\" type=\"reset\" value=\"Restaurar\">
                                                        </td>
                                                    </tr>
                                                </table>
                                            </td>
                                        </tr>
                                    </table>
                                </td>
                            </tr>
                        </table>
                    </form>";
        }
        else
        {
            $NombreBD = "datos_usuarios";
            $Servidor = "localhost";
            $Usuario = "root";
            $password = "45612696";

            $IdConexion = mysql_connect($Servidor, $Usuario, $password);
            mysql_select_db($NombreBD, $IdConexion);

            //Esta es la consulta que queremos hacer.
            $consulta = 'SELECT USUARIO,DIRECCION,EMAIL,IDUSUARIO,SEXO FR
OM usuarios_oro;';
```

```
            //mysql_query nos devuelve el identificador de la consulta.
            $IdConsulta = mysql_query($consulta, $IdConexion);

            //Nos devuelve el número de filas;
            $NFilas = mysql_num_rows($IdConsulta);

            for( $i=0; $i<$NFilas; $i++ )
            {
                //Para recuperar los datos.
                $resultadoConsulta = mysql_fetch_array($IdConsulta);

                if( ( $resultadoConsulta[ "USUARIO" ] == $_SESSION[ 'nomb
re' ] ) && ( $resultadoConsulta[ "IDUSUARIO" ] == md5( $_SESSION[ 'id' ]
) ) )
                {
                    ?>
                        <center>
                            <br><br><br>
                            <h2>Bienvenido  <? echo $_SESSION[ 'nombre' ]
 ?></h2>

                            <br><br><br>
                            <table align="center" border="0" cellspacing=
"0" cellpadding="1" width="50%">
                            <td align="left">
                                <a href="/protegido/Documentos" onmouseov
er="window.status='Enlace al directorio documentos';return true">Document
os</a>
                            </td>
                            <td align="right">
                                <a href="/protegido/Musica" onmouseover="
window.status='Enlace al directorio Musica';return true">Musica</a>
                            </td>
                            </table>
                            <?
                                break;
                }
            }
            if( ( $resultadoConsulta[ "USUARIO" ] != $_SESSION[ 'nombre'
] ) || ( $resultadoConsulta[ "IDUSUARIO" ] != md5( $_SESSION[ 'id' ] ) )
)
            {
                ?>
                    <br><br><br>
                    <center>
                        <h2><? echo $_SESSION[ 'nombre' ]; ?> no tienes l
os privilegios para acceder<br>a este enlace</h2>
                    </center>
                <?
            }
        }
    ?>
    </div>
    <!--end content -->
    <div align="left" id="siteInfo">
        <img src="Imagenes/upct.jpg" width="35" height="35" />
        <a href="/protegido/email.doc">Contacta con Nosotros</a> | &copy;
2006 Company Name
    </div>
    <br />
```

```
</body>
</html>
```

Apéndice G

CÓDIGO PHP: ALMACENADO DE DATOS

```
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-
1" />
    <title>PFC: Implementación de seguridad Web</title>
    <link rel="stylesheet" href="estilos.css" type="text/css" />
</head>
<body background="Imagenes/fondo.jpg">
    <div id="masthead">
        <h1 align="center" id="siteName">LA WEB DE JUANMA </h1>
        <div id="globalNav">
            <p><img alt="" src="Imagenes/glbnav_background.gif" height="3
2" width="100%" id="gnl" /></p>
        </div>
        <h2 align="center" id="pageName">Almacenar datos en base de datos
</h2>
    </div>
    <div id="navBar">
        <div id="search">
            <form action="http://www.google.es">
                <label>Buscar</label>
                <input type="text" size="10" />
                <input type="submit" value="go" />
            </form>
        </div>
        <div id="sectionLinks">
            <h3>Enlaces</h3>
            <ul>
                <li><a href="http://www.upct.es">UPCT</a></li>
                <li><a href="http://www.teleco.upct.es">TELECO</a></li>
                <li><a href="http://redcampus.upct.es/redcampus/acceso.ht
m">SECRETARIA VIRTUAL</a></li>
                <li><a href="http://www.bib.upct.es/">BIBLIOTECA UPCT</a>
</li>
                <li><a href="http://www.upct.es/contenido/idi/index_idi.p
hp/">I+D+I</a></li>
                <li><a href="http://www.alumnosteleco.upct.es/">DELEGACIÓ
N DE ALUMNOS</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Otros Links</h3>
            <ul>
                <li><a href="http://www.umu.es">UMU</a></li>
                <li><a href="http://www.universia.es">Universia</a></li>
            </ul>
        </div>
    </div>
    <!--end navBar1 div -->
    <div id="navBar1">
        <div id="sectionLinks">
            <h3>SECCIONES</h3>
            <ul>
                <li><a href="index.php">Principal</a></li>
                <li><a href="iniciarSesion.php">Iniciar Sesión</a></li>
```

```
                <li><a href="galeriaFotos.php">Galeria de fotos</a></li>
                <li><a href="tusDatos.php">Tus Datos</a></li>
                <li><a href="usuariosOro.php">Usuarios Oro</a></li>
                <li><a href="cerrarSesion.php">Cerrar Sesión</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Registrarse</h3>
            <ul>
                <li><a href="registrarse.php">Registrate</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Baja usuarios</h3>
            <ul>
                <li><a href="bajaUsuarios.php">Baja</a></li>
            </ul>
        </div>
    </div>
    <!--end headlines -->
    <div id="content">
        <?php
            $NombreBD = "datos_usuarios";
            $Servidor = "localhost";
            $Usuario = "root";
            $password = "45612696";

            $IdConexion = mysql_connect($Servidor, $Usuario, $password);

            mysql_select_db($NombreBD, $IdConexion);

            //Esta es la consulta que queremos hacer.
            $consulta = 'SELECT USUARIO,DIRECCION,EMAIL,IDUSUARIO,SEXO FR
OM usuarios;';

            //mysql_query nos devuelve el identificador de la consulta.
            $IdConsulta = mysql_query($consulta, $IdConexion);

            //Nos devuelve el número de filas;
            $NFilas = mysql_num_rows($IdConsulta);
        ?>

        <CENTER>
            <?
                for( $i=0; $i<$NFilas; $i++ )
                {
                    //Para recuperar los datos.
                    $resultadoConsulta = mysql_fetch_array($IdConsulta);
                    if( $resultadoConsulta[ "USUARIO" ] == $_POST['usuari
o' ] )
                    {
                        $aux = 1;
                        ?><br><br><br>
                        <h2>Usuario registrado</h2>
                        <?
                        break;
                    }
                }
                if( $aux != 1 )
                {
                    $insercion = 'INSERT INTO usuarios'.
```

```
                                    ' (USUARIO,DIRECCION,EMAIL,IDUSUARIO,SEX
O)'.
                                    ' VALUES ('.'"'.$_POST['usuario' ].'","'
.$_POST['direccion' ].'","'.$_POST['email' ].'","'.md5( $_POST['IDUsuario
'] ).'","'.$_POST['genero'].'");';
                        $IdInsercion = mysql_query($insercion, $IdConexion);
                        ?><br><br><br>
                        <h2>Tus datos han sido guardados</h2>
                        <?
                    }
                ?>
            </CENTER>
        </div>
        <!--end content -->
        <div id="siteInfo">
            <img src="Imagenes/upct.jpg" width="35" height="35" />
            <a href="/protegido/email.doc">Contacta con Nosotros</a> | &copy;
2006 Company Name
        </div>
        <br />
</body>
</html>
```

Apéndice H

CÓDIGO PHP: BAJA DE USUARIOS

```
<?
    session_start();
?>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-
1" />
    <title>PFC: Implementación de seguridad Web</title>
    <link rel="stylesheet" href="estilos.css" type="text/css" />
</head>
<body background="Imagenes/fondo.jpg">
    <div id="masthead">
        <h1 align="center" id="siteName">LA WEB DE JUANMA </h1>
        <div id="globalNav">
            <p><img alt="" src="Imagenes/glbnav_background.gif" height="3
2" width="100%" id="gnl" /></p>
        </div>
        <h2 align="center" align="center" id="pageName">Baja de Usuarios<
/h2>
    </div>
    <div id="navBar">
        <div id="search">
            <form action="http://www.google.es">
                <label>Buscar</label>
                <input type="text" size="10" />
                <input type="submit" value="go" />
            </form>
        </div>
        <div id="sectionLinks">
            <h3>Enlaces</h3>
            <ul>
                <li><a href="http://www.upct.es">UPCT</a></li>
                <li><a href="http://www.teleco.upct.es">TELECO</a></li>
                <li><a href="http://redcampus.upct.es/redcampus/acceso.ht
m">SECRETARIA VIRTUAL</a></li>
                <li><a href="http://www.bib.upct.es/">BIBLIOTECA UPCT</a>
</li>
                <li><a href="http://www.upct.es/contenido/idi/index_idi.p
hp/">I+D+I</a></li>
                <li><a href="http://www.alumnosteleco.upct.es/">DELEGACIÓ
N DE ALUMNOS</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Otros Links</h3>
            <ul>
                <li><a href="http://www.umu.es">UMU</a></li>
                <li><a href="http://www.universia.es">Universia</a></li>
            </ul>
        </div>
    </div>
    <!--end navBar1 div -->
    <div id="navBar1">
        <div id="sectionLinks">
            <h3>SECCIONES</h3>
            <ul>
```

```
                <li><a href="index.php">Principal</a></li>
                <li><a href="iniciarSesion.php">Iniciar Sesión</a></li>
                <li><a href="galeriaFotos.php">Galeria de fotos</a></li>
                <li><a href="tusDatos.php">Tus Datos</a></li>
                <li><a href="usuariosOro.php">Usuarios Oro</a></li>
                <li><a href="cerrarSesion.php">Cerrar Sesión</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Registrarse</h3>
            <ul>
                <li><a href="registrarse.php">Registrate</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Baja usuarios</h3>
            <ul>
                <li><a href="bajaUsuarios.php">Baja</a></li>
            </ul>
        </div>
    </div>
    <!--end headlines -->
    <div id="content">
        <?
            $NombreBD = "datos_usuarios";
            $Servidor = "localhost";
            $Usuario = "root";
            $password = "45612696";

            $IdConexion = mysql_connect($Servidor, $Usuario, $password);

            mysql_select_db($NombreBD, $IdConexion);
        ?>
        <CENTER>
            <?
                if( $_SESSION['nombre'] == '' )
                {
                    echo
                        "<form action=\"comprobarDatosBorrados.php\" meth
od=\"POST\">
                            <br><br><br>
                            <table align=\"center\" border=\"1\" cellspac
ing=\"0\" cellpadding=\"0\" width=\"450\">
                                <tr bgcolor=\"#C6C3C6\">
                                    <td>
                                        <table align=\"center\" width=\"4
50\" border=\"0\" cellspacing=\"0\" cellpadding=\"1\">
                                            <tr bgcolor=\"#400080\">
                                                <td height=\"20\" class=\
"td1\" bgcolor=\"#000084\">
                                                    <b>
                                                        <font color=\"#FF
FFFF\">
                                                             Sistema
 de autentificación de usuarios
                                                        </font>
                                                    </b>
                                                </td>
                                            </tr>
                                            <tr align=\"center\">
                                                <td colspan=\"2\">
```

```
                                                    <table border=\"0\" c
ellspacing=\"0\" cellpadding=\"1\" width=\"100%\">
                                                        <tr>
                                                            <td rowspan=\
"3\" valign=\"top\" align=\"right\" width=\"12%\">
                                                                <img src=
\"Imagenes/key.gif\" width=\"35\" height=\"40\">
                                                            </td>
                                                            <td colspan=\
"2\" class=\"td1\" height=\"42\">
                                                                Por favor
 ingresa tu nombre de usuario y  password
                                                            </td>
                                                        </tr>
                                                        <tr>
                                                            <td width=\"1
8%\" class=\"td1\">
                                                                Usuario
                                                            </td>
                                                            <td width=\"7
0%\">
                                                                <input ty
pe=\"text\" class=\"campo\" name=\"usuario\" size=\"25\">
                                                            </td>
                                                        </tr>
                                                        <tr>
                                                            <td height=\"
35\" width=\"18%\" class=\"td1\">
                                                                Contraseñ
a
                                                            </td>
                                                            <td height=\"
35\" width=\"70%\">
                                                                <input ty
pe=\"password\" class=\"campo\" name=\"IDUsuario\" size=\"25\">
                                                            </td>
                                                        </tr>
                                                    </table>
                                                    <table width=\"100%\"
 border=\"0\" cellspacing=\"0\" cellpadding=\"1\">
                                                        <tr>
                                                            <td align=\"r
ight\" width=\"47%\">
                                                                <input cl
ass=\"boton\" type=\"submit\" value=\"Aceptar\">
                                                            </td>
                                                            <td align=\"r
ight\" width=\"1%\">
                                                                 
                                                            </td>
                                                            <td width=\"5
2%\">
                                                                <input cl
ass=\"boton\" type=\"reset\" value=\"Restaurar\">
                                                            </td>
                                                        </tr>
                                                    </table>
                                                </td>
                                            </tr>
                                        </table>
                                    </td>
```

```
                                </tr>
                            </table>
                        </form>";
                }
                else
                {
                    $nom = $_SESSION[ 'nombre' ];
                    $query = "delete from usuarios where usuario='$nom'";
                    mysql_query($query);
                    unset( $_SESSION[ 'nombre' ] ) ;
                    echo "<h2><b><br>Tus datos han sido eliminados</b></h
2>";
                }
            ?>
        </center>
    </div>
    <!--end content -->
    <div id="siteInfo">
        <img src="Imagenes/upct.jpg" width="35" height="35" />
        <a href="/protegido/email.doc">Contacta con Nosotros</a> | &copy;
2006 Company Name
    </div>
    <br />
</body>
</html>
```

Apéndice I

CÓDIGO PHP: CERRAR SESIÓN

```
<?
    session_start();
    // Destruye todas las variables de la sesi&oacute;n
    unset( $_SESSION[ 'nombre' ]) ;
?>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-
1" />
    <title>PFC: Implementación de seguridad Web</title>
    <link rel="stylesheet" href="estilos.css" type="text/css" />
</head>
<body background="Imagenes/fondo.jpg">
    <div id="masthead">
        <h1 align="center" id="siteName">LA WEB DE JUANMA </h1>
        <div id="globalNav">
            <p><img alt="" src="Imagenes/glbnav_background.gif" height="3
2" width="100%" id="gnl" /></p>
        </div>
        <h2 align="center" id="pageName">Cerrar Sesión</h2>
    </div>
    <div id="navBar">
        <div id="search">
            <form action="http://www.google.es">
                <label>Buscar</label>
                <input type="text" size="10" />
                <input type="submit" value="go" />
            </form>
        </div>
        <div id="sectionLinks">
            <h3>Enlaces</h3>
            <ul>
                <li><a href="http://www.upct.es">UPCT</a></li>
                <li><a href="http://www.teleco.upct.es">TELECO</a></li>
                <li><a href="http://redcampus.upct.es/redcampus/acceso.ht
m">SECRETARIA VIRTUAL</a></li>
                <li><a href="http://www.bib.upct.es/">BIBLIOTECA UPCT</a>
</li>
                <li><a href="http://www.upct.es/contenido/idi/index_idi.p
hp/">I+D+I</a></li>
                <li><a href="http://www.alumnosteleco.upct.es/">DELEGACIÓ
N DE ALUMNOS</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Otros Links</h3>
            <ul>
                <li><a href="http://www.umu.es">UMU</a></li>
                <li><a href="http://www.universia.es">Universia</a></li>
            </ul>
        </div>
    </div>
    <!--end navBar1 div -->
    <div id="navBar1">
        <div id="sectionLinks">
            <h3>SECCIONES</h3>
```

```
            <ul>
                <li><a href="index.php">Principal</a></li>
                <li><a href="iniciarSesion.php">Iniciar Sesión</a></li>
                <li><a href="galeriaFotos.php">Galeria de fotos</a></li>
                <li><a href="tusDatos.php">Tus Datos</a></li>
                <li><a href="usuariosOro.php">Usuarios Oro</a></li>
                <li><a href="cerrarSesion.php">Cerrar Sesión</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Registrarse</h3>
            <ul>
                <li><a href="registrarse.php">Registrate</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Baja usuarios</h3>
            <ul>
                <li><a href="bajaUsuarios.php">Baja</a></li>
            </ul>
        </div>
    </div>
    <!--end headlines -->
    <div id="content">
        <br><br><br>
        <center>
            <h1>SESION CERRADA</h1>
        </center>
    </div>
    <!--end content -->
    <div id="siteInfo">
        <img src="Imagenes/upct.jpg" width="35" height="35" />
        <a href="/protegido/email.doc">Contacta con Nosotros</a> | &copy;
2006 Company Name
    </div>
    <br />
</body>
</html>
```

Apéndice J

CÓDIGO PHP: CLAVE ORO

```
<?
    session_start();
    $_SESSION[ 'nombre' ] = $_POST[ 'usuario' ];
    $_SESSION[ 'id' ] = $_POST[ 'IDUsuario' ];
?>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-
1" />
    <title>PFC: Implementación de seguridad Web</title>
    <link rel="stylesheet" href="estilos.css" type="text/css" />
</head>
<body background="Imagenes/fondo.jpg">
    <div id="masthead">
        <table width="100%">
            <td width="98" align="left">
                <img src="Imagenes/upct.jpg" width="85" height="85"/>
            </td>
            <td width="840" align="center">
                <h1 align="center">LA WEB DE JUANMA</h1>
            </td>
            <td width="86" align="right">
                <img src="Imagenes/upct.jpg" width="85" height="85" align
="right"/>
            </td>
        </table>
        <script language="JAVASCRIPT">
            var refresH=1200;
            var numMensaje=0;
            msgS=new Array();
            msgS[1]="Bienvenido a la wed de Juan Manuel";
            msgS[2]="Universidad Politécnica de Cartagena";
            msgS[3]="PFC titulado: 'Portales webs seguros'";
            msgS[4]="PFC dirigido por: Pedro Sánchez Palma";
            msgS[5]=" ";

            function refrescar()
            {
                numMensaje += 1;
                if (numMensaje == msgS.length)
                    numMensaje=1;
                window.status=msgS[numMensaje];
                refreshC=setTimeout("refrescar();",refresH);
            }
            refrescar();
        </script>

        <div id="globalNav">
            <p><img alt="" src="Imagenes/glbnav_background.gif" height="3
2" width="100%" id="gnl" /></p>
        </div>
        <h2 id="pageName">Principal</h2>
    </div>
    <div id="navBar">
        <div id="search">
            <form action="http://www.google.es">
```

```
                <label>Buscar</label>
                <input type="text" size="10" />
                <input type="submit" value="go" />
            </form>
        </div>
        <div id="sectionLinks">
            <h3>Enlaces</h3>
            <ul>
                <li><a href="http://www.upct.es">UPCT</a></li>
                <li><a href="http://www.teleco.upct.es">TELECO</a></li>
                <li><a href="http://redcampus.upct.es/redcampus/acceso.ht
m">SECRETARIA VIRTUAL</a></li>
                <li><a href="http://www.bib.upct.es/">BIBLIOTECA UPCT</a>
</li>
                <li><a href="http://www.upct.es/contenido/idi/index_idi.p
hp/">I+D+I</a></li>
                <li><a href="http://www.alumnosteleco.upct.es/">DELEGACIÓ
N DE ALUMNOS</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Otros Links</h3>
            <ul>
                <li><a href="http://www.umu.es">UMU</a></li>
                <li><a href="http://www.universia.es">Universia</a></li>
            </ul>
        </div>
    </div>
    <!--end navBar1 div -->
    <div id="navBar1">
        <div id="sectionLinks">
            <h3>SECCIONES</h3>
            <ul>
                <li><a href="index.php">Principal</a></li>
                <li><a href="iniciarSesion.php">Iniciar Sesión</a></li>
                <li><a href="galeriaFotos.php">Galeria de fotos</a></li>
                <li><a href="tusDatos.php">Tus Datos</a></li>
                <li><a href="usuariosOro.php">Usuarios Oro</a></li>
                <li><a href="cerrarSesion.php">Cerrar Sesión</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Registrarse</h3>
            <ul>
                <li><a href="registrarse.php">Registrate</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Baja usuarios</h3>
            <ul>
                <li><a href="bajaUsuarios.php">Baja</a></li>
            </ul>
        </div>
    </div>
    <!--end headlines -->
    <div id="content">
        <?
            $NombreBD = "datos_usuarios";
            $Servidor = "localhost";
            $Usuario = "root";
            $password = "45612696";
```

```
            $IdConexion = mysql_connect($Servidor, $Usuario, $password);

            mysql_select_db($NombreBD, $IdConexion);

            //Esta es la consulta que queremos hacer.
            $consulta = 'SELECT USUARIO,DIRECCION,EMAIL,IDUSUARIO,SEXO FR
OM usuarios_oro;';

            //mysql_query nos devuelve el identificador de la consulta.
            $IdConsulta = mysql_query($consulta, $IdConexion);

            //Nos devuelve el número de filas;
            $NFilas = mysql_num_rows($IdConsulta);

            for( $i=0; $i<$NFilas; $i++ )
            {
                //Para recuperar los datos.
                $resultadoConsulta = mysql_fetch_array($IdConsulta);

                if( ( $resultadoConsulta[ "USUARIO" ] == $_POST[ 'usuario
' ] ) && ( $resultadoConsulta[ "IDUSUARIO" ] == md5( $_POST[ 'IDUsuario'
] ) ) )
                {
                    ?>
                        <center>
                            <br><br><br>
                            <h2>Bienvenido  <? echo $_SESSION[ 'nombre' ]
 ?></h2>

                            <br><br><br>
                                <table align="center" border="0" cellspac
ing="0" cellpadding="1" width="50%">
                                <td align="left">
                                    <a href="/protegido/Documentos" onmou
seover="window.status='Enlace al directorio documentos';return true">Docu
mentos</a>
                                </td>
                                <td align="right">
                                    <a href="/protegido/Musica" onmouseov
er="window.status='Enlace al directorio Musica';return true">Musica</a>
                                </td>
                                </table>
                    <?
                    break;
                }
            }
            if( ( $resultadoConsulta[ "USUARIO" ] != $_POST[ 'usuario' ]
) || ( $resultadoConsulta[ "IDUSUARIO" ] != md5( $_POST[ 'IDUsuario' ] )
) )
            {
                echo
                    "<form action=\"claveOro.php\" method=\"POST\">
                        <br><br><br>
                        <table align=\"center\" border=\"1\" cellspacing=
\"0\" cellpadding=\"0\" width=\"450\">
                            <tr bgcolor=\"#C6C3C6\">
                            <td>
                                <table align=\"center\" width=\"450\"
 border=\"0\" cellspacing=\"0\" cellpadding=\"1\">
                                    <tr bgcolor=\"#400080\">
```

```
                                        <td height=\"20\" class=\"td1
\" bgcolor=\"#000084\">
                                            <b>
                                                <font color=\"#FFFFFF
\">
                                                     Sistema de
autentificación de usuarios
                                                </font>
                                            </b>
                                        </td>
                                    </tr>
                                    <tr align=\"center\">
                                        <td colspan=\"2\">
                                            <table border=\"0\" cells
pacing=\"0\" cellpadding=\"1\" width=\"100%\">
                                                <tr>
                                                    <td rowspan=\"3\"
 valign=\"top\" align=\"right\" width=\"12%\">
                                                        <img src=\"Im
agenes/key.gif\" width=\"35\" height=\"40\">
                                                    </td>
                                                    <td colspan=\"2\"
 class=\"td1\" height=\"42\">
                                                        <font color=\
"#0000ff\">
                                                            Nombre de
 usuario o password incorrectos
                                                        </font>
                                                    </td>
                                                </tr>
                                                <tr>
                                                    <td width=\"18%\"
 class=\"td1\">
                                                        Usuario
                                                    </td>
                                                    <td width=\"70%\"
>
                                                        <input type=\
"text\" class=\"campo\" name=\"usuario\" size=\"25\">
                                                    </td>
                                                </tr>
                                                <tr>
                                                    <td height=\"35\"
 width=\"18%\" class=\"td1\">
                                                        Contraseña
                                                    </td>
                                                    <td height=\"35\"
 width=\"70%\">
                                                        <input type=\
"password\" class=\"campo\" name=\"IDUsuario\" size=\"25\">
                                                    </td>
                                                </tr>
                                            </table>
                                            <table width=\"100%\" bor
der=\"0\" cellspacing=\"0\" cellpadding=\"1\">
                                                <tr>
                                                    <td align=\"right
\" width=\"47%\">
                                                        <input class=
\"boton\" type=\"submit\" value=\"Aceptar\">
                                                    </td>
```

```
                                                        <td align=\"right
\" width=\"1%\">
                                                             
                                                        </td>
                                                        <td width=\"52%\"
>
                                                            <input class=
\"boton\" type=\"reset\" value=\"Restaurar\">
                                                        </td>
                                                    </tr>
                                                </table>
                                            </td>
                                        </tr>
                                    </table>
                                </td>
                            </tr>
                        </table>
                    </form>";
            }
        ?>
    </div>
    <!--end content -->
    <div align="left" id="siteInfo">
        <img src="Imagenes/upct.jpg" width="35" height="35" />
        <a href="/protegido/email.doc">Contacta con Nosotros</a> | &copy;
2006 Company Name
    </div>
    <br />
</body>
</html>
```

Apéndice K

CÓDIGO PHP: CONFIRMACIÓN DE DATOS

```
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-
1" />
    <title>PFC: Implementación de seguridad Web</title>
    <link rel="stylesheet" href="estilos.css" type="text/css" />
</head>
<body background="Imagenes/fondo.jpg">
    <div id="masthead">
        <h1 align="center" id="siteName">LA WEB DE JUANMA </h1>
        <div id="globalNav">
            <p><img alt="" src="Imagenes/glbnav_background.gif" height="3
2" width="100%" id="gnl" /></p>
        </div>
        <h2 align="center" id="pageName">Confirmar Datos</h2>
    </div>
    <div id="navBar">
        <div id="search">
            <form action="http://www.google.es">
                <label>Buscar</label>
                <input type="text" size="10" />
                <input type="submit" value="go" />
            </form>
        </div>
        <div id="sectionLinks">
            <h3>Enlaces</h3>
            <ul>
                <li><a href="http://www.upct.es">UPCT</a></li>
                <li><a href="http://www.teleco.upct.es">TELECO</a></li>
                <li><a href="http://redcampus.upct.es/redcampus/acceso.ht
m">SECRETARIA VIRTUAL</a></li>
                <li><a href="http://www.bib.upct.es/">BIBLIOTECA UPCT</a>
</li>
                <li><a href="http://www.upct.es/contenido/idi/index_idi.p
hp/">I+D+I</a></li>
                <li><a href="http://www.alumnosteleco.upct.es/">DELEGACIÓ
N DE ALUMNOS</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Otros Links</h3>
            <ul>
                <li><a href="http://www.umu.es">UMU</a></li>
                <li><a href="http://www.universia.es">Universia</a></li>
            </ul>
        </div>
    </div>
    <!--end navBar1 div -->
    <div id="navBar1">
        <div id="sectionLinks">
            <h3>SECCIONES</h3>
            <ul>
                <li><a href="index.php">Principal</a></li>
                <li><a href="iniciarSesion.php">Iniciar Sesión</a></li>
                <li><a href="galeriaFotos.php">Galeria de fotos</a></li>
                <li><a href="tusDatos.php">Tus Datos</a></li>
```

```
                <li><a href="usuariosOro.php">Usuarios Oro</a></li>
                <li><a href="cerrarSesion.php">Cerrar Sesión</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Registrarse</h3>
            <ul>
                <li><a href="registrarse.php">Registrate</a></li>
            </ul>
        </div>
        <div class="relatedLinks">
            <h3>Baja usuarios</h3>
            <ul>
                <li><a href="bajaUsuarios.php">Baja</a></li>
            </ul>
        </div>
    </div>
    <!--end headlines -->
    <div id="content">
        <br><br><br>
        <h2 align="center">Confirmación de los datos</h2>
        <form name="Confirmacion" method="post" action="almacenadoDatos.p
hp" enctype="multipart/form-data">
            <table width="400" border="1" align="center" cellpadding="5"
cellspacing="0" bgcolor="#CCCCCC">
                <tr>
                    <!--Validando el valor de entrada para el nombre-->
                    <td width="47%">
                        Nombre
                    </td>
                    <td>
                        <?php
                            if( empty( $_POST[ 'usuario' ] ) )
                            {
                                echo( "Nombre inválido" );
                                $contador = 1;
                            }
                            elseif( ( strlen( $_POST[ 'usuario' ] ) < 5 )
 || ( strlen( $_POST[ 'usuario' ] ) > 20 ) )
                            {
                                echo( "Longitud del nombre invalida, entr
e 5 y 20 caracteres" );
                                $contador = 1;
                            }
                            else
                            {
                                echo $_POST[ 'usuario' ];
                            }
                        ?>
                    </td>
                </tr>
                <tr>
                    <!--Validando el valor del campo address-->
                    <td width="47%">
                        Dirección
                    </td>
                    <td>
                        <?php
                            if( empty( $_POST[ 'direccion' ] ) )
                            {
                                echo( "Dirección inválida" );
```

```
                                $contador = 1;
                            }
                            elseif( ( strlen( $_POST[ 'direccion' ] ) < 5
 ) || ( strlen( $_POST[ 'direccion' ] ) > 200  ) )
                            {
                                echo( "Longitud de la dirección invalida,
 entre 5 y 200 caracteres" );
                                $contador = 1;
                            }
                            else
                                echo $_POST[ 'direccion' ];
                        ?>
                    </td>
                </tr>
                <tr>
                    <td width="47%">
                        Correo electrónico
                    </td>
                    <td>
                        <?php
                            if( empty( $_POST[ 'email' ] ) )
                            {
                                echo( "Dirección de correo inválida" );
                                $contador = 1;
                            }
                            elseif( ( strlen( $_POST[ 'email' ] ) < 5 ) |
| ( strlen( $_POST[ 'email' ] ) > 50  ) )
                            {
                                echo( "Longitud del email invalida, entre
 5 y 50 caracteres" );
                                $contador = 1;
                            }
                            else
                                echo $_POST[ 'email' ];
                        ?>
                    </td>
                <tr>
                    <td width="47%">
                        Contraseña
                    </td>
                    <td>
                        <?php
                            if( empty( $_POST[ 'IDUsuario' ] ) || empty(
$_POST[ 'CIDUsuario' ] ) )
                            {
                                echo( "Contraseña inválida" );
                                $contador = 1;
                            }
                            elseif( ( strlen( $_POST[ 'IDUsuario' ] ) < 5
 ) || ( strlen( $_POST[ 'CIDUsuario' ] ) > 15  ) )
                            {
                                echo( "Longitud de la contraseña invalida
, entre 5 y 15 caracteres" );
                                $contador = 1;
                            }
                            elseif( !( strlen( $_POST[ 'IDUsuario' ] ) ==
 strlen( $_POST[ 'CIDUsuario' ] ) ) )
                            {
                                echo( "Las contraseñas no coinciden" );
                                $contador = 1;
                            }
```

```
                            elseif( !( $_POST[ 'IDUsuario' ] === $_POST[
'CIDUsuario' ] ) )
                            {
                                echo( "Las contraseñas no coinciden" );
                                $contador = 1;
                            }
                            else
                                echo $_POST[ 'IDUsuario' ];
                        ?>
                    </td>
                </tr>
                <tr>
                    <td width="47%">
                        Sexo
                    </td>
                    <td>
                        <?php
                            if( empty( $_POST[ 'genero' ] ) )
                            {
                                echo( "No se ha especificado el sexo" );
                                $contador = 1;
                            }
                            elseif( !( ( $_POST[ 'genero' ] == "Hombre" )
 || ( $_POST[ 'genero' ] == "Mujer" ) ) )
                            {
                                echo( "Valor inválido para el sexo" );
                                $contador = 1;
                            }
                            else
                                echo $_POST[ 'genero' ];
                        ?>
                    </td>
                </tr>
                <tr>
                    <td colspan="3">
                        <form name=confir action="almacenadoDatos.php">
                            <?php
                                echo "<input type=hidden name=\"usuario\"
 value=\"".$_POST[ 'usuario' ]."\">\n";
                                echo "<input type=hidden name=\"direccion
\" value=\"".$_POST[ 'direccion' ]."\">\n";
                                echo "<input type=hidden name=\"email\" v
alue=\"".$_POST[ 'email' ]."\">\n";
                                echo "<input type=hidden name=\"IDUsuario
\" value=\"".$_POST[ 'IDUsuario' ]."\">\n";
                                echo "<input type=hidden name=\"CIDUsuari
o\" value=\"".$_POST[ 'CIDUsuario' ]."\">\n";
                                echo "<input type=hidden name=\"genero\"
value=\"".$_POST[ 'genero' ]."\">\n";
                                if( $contador != 1 )
                                {
                                    ?>
                                        <center>
                                            <input type="submit" name="Su
bmit" value="Confirmar">
                                        </center>
                                    <?
                                }
                            ?>
                        </form>
                    </td>
```

```
                </tr>
            </table>
        </form>
    </div>
    </div>
    <!--end content -->
    <div id="siteInfo">
        <img src="Imagenes/upct.jpg" width="35" height="35" />
        <a href="/protegido/email.doc">Contacta con Nosotros</a> | &copy;
2006 Company Name
    </div>
    <br />
</body>
</html>
```

Apéndice L

CÓDIGO PHP: FOTOS

```
<?
    session_start();
?>
<HTML>
    <HEAD>
        <title>LA WEB DE JUANMA MACHOTE. FOTOS AMIGOS</title>
        <link href="estilos1.css" rel="stylesheet" type="text/css" media=
"print">
    <HEAD>

    <BODY bgcolor="#CCCCCC" oncontextmenu="return false" onkeydown="retur
n false" background="Imagenes/fondo.jpg">
        <div class=\"nover\">
            <?
                if ( $_SESSION['nombre'] == '' )
                {
                    echo "caca";
                    exit;
                }
                else
                    echo "<table width=\"100%\" border=\"0\" cellspacing=
\"1\" cellpadding=\"0\">
                            <tr>
                                <td align=\"left\"><img src=\"./Imagenes/
eli&po.jpg\" width=\"150\" hight=\"150\"></td>
                                <td align=\"center\"><img src=\"./Imagene
s/ju&ali.jpg\" width=\"150\" hight=\"150\"></td>
                                <td align=\"right\"><img src=\"./Imagenes
/luc&noe.jpg\" width=\"150\" hight=\"150\"></td>
                            </tr>
                            <tr>
                                <td align=\"left\"><img src=\"./Imagenes/
\" width=\"150\" hight=\"150\"></td>
                                <td align=\"center\"><img src=\"./Imagene
s/pe&pa.jpg\" width=\"150\" hight=\"150\"></td>
                                <td align=\"right\"><img src=\"./Imagenes
/ali&jua.jpg\" width=\"150\" hight=\"150\"></td>
                            </tr>
                            <tr>
                                <td align=\"left\"><img src=\"./Imagenes/
noe&lola.jpg\" width=\"150\" hight=\"150\"></td>
                                <td align=\"center\"><img src=\"./Imagene
s/\" width=\"150\" hight=\"150\"></td>
                                <td align=\"right\"><img src=\"./Imagenes
/ma&pa.jpg\" width=\"150\" hight=\"150\"></td>
                            </tr>
                            <tr>
                                <td align=\"left\"><img src=\"./Imagenes/
fi&ma.jpg\" width=\"150\" hight=\"150\"></td>
                                <td align=\"center\"><img src=\"./Imagene
s/parejas.jpg\" width=\"150\" hight=\"150\"></td>
                                <td align=\"right\"><img src=\"./Imagenes
/chicas.jpg\" width=\"150\" hight=\"150\"></td>
                            </tr>
                        </table>"
```

```
            ?>
        </div>
    </BODY>
</HTML>
```

Apéndice M

CONFIGURACIÓN httpd.conf

```
LoadModule php4_module        libexec/libphp4.so
.

.

.
<IfDefine SSL>
Listen 80
Listen 443
</IfDefine>
.

.

.
DocumentRoot "/usr/local/apache2/htdocs"
.

.

.
<Directory "/usr/local/apache2/htdocs/protegido">
    AuthName "privatefiles"
    AuthType Basic
    AuthUserFile /usr/local/apache2/conf/passwd_basic
    Require valid-user
</Directory>

<Directory "/usr/local/apache2/htdocs/protegido">
    AuthName "privatefiles"
    AuthType Basic
    AuthUserFile /usr/local/apache2/conf/passwd_basic
    AuthGroupFile /usr/local/apache2/conf/ficherogrupos
    Require group grupo1
</Directory>

<Directory />
      Options None
      AllowOverride None
</Directory>

<Directory /MiSitio>
      Options Indexes FollowSymLinks Includes
      AllowOverride None
</Directory>
.

.
```

```
.
<IfModule mod_dir.c>
    DirectoryIndex index.html index.php index.php3
</IfModule>
.
.
.
```

BIBLIOGRAFÍA

[1] http://www.sindominio.net/suburbia/article.php3?id_article=33

[2]http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/MonogSO/SEGUNIX01.htm

[3] http://www.cert.org

[4] http://www.sans.org

[5] http://www.17799central.com/spain.htm

[6] Seguridad en las comunicaciones y en la información. Gabriel Díaz, Francisco Mur, Elio Sancristóbal, Manuel Castro, Juan Peire. Páginas 144-155. Enero 2004

[7] http://www.aeat.es/normlegi/otros/lortad2000.htm

[8] http://es.tldp.org/almacen/Manuales-LuCAS/GARL2/garl2/x-082-2-firewall.attacks.html

[9] http://www.todomba.com/displayarticle483.html

[10] http://es.wikibooks.org/wiki/Hacks_para_Unix-like:Tipos_de_ataques_remotos#Escaneo_de_puertos

[11] http://es.tldp.org/Manuales-LuCAS/

[12] http://es.wikipedia.org/wiki/Sistema_de_detecci%C3%B3n_de_intrusos

[13] http://es.wikipedia.org/

[14] http://www.cert.org

[15] http://www.technotronic.com/

[16] http://www.dgonzalez.net

[17] http://www-i2.informatik.rwth-aachen.de/Staff/Current/noll/Publications/

[18] http://www.rediris.es/ y Teleco-Forum UPCT 2006 conferencia "Seguridad en Redes: BotNets".

[19] http://csrc.nist.gov/publications/nistpubs/

[20] http://penta.ufrgs.br

[21] http://www.cisco.com/

[22] Seguridad en las comunicaciones y en la información. Gabriel Díaz, Francisco Mur, Elio Sancristóbal, Manuel Castro, Juan Peire. Páginas 340-346. Enero 2004

[23] www.fnmt.es/

[24] https://www.cert.fnmt.es/certifi.htm

[25] http://www.ace.es/

[26] http://www.feste.com/

[27] http://www.ipsca.es/

[28] http://revista.robotiker.com/

[29] http://httpd.apache.org/docs/2.0/es/env.html

[30] http://www.theregister.co.uk/2003/10/06/linux_vs_windows_viruses/

[31] http://es.tldp.org/Manuales-LuCAS/GSAL/gsal-19991128-htm/

[32] Seguridad y Comercio en el Web. Simson Garfinkel y Gene Spafford. Páginas 233-243

[33] http://www.verisign.es/support/ssl-certificates-support/page_es_es_dev022169.html

[34] http://www.desarrolloweb.com/articulos/392.php?manual=27

[35] www.php.net

[36] Proyectos profesionales PHP. Asís Wilfred, Meeta Gupta y Kartik Bhatnagar con NIIT.

[37] http://www.programacion.com/bbdd/

[38] Aprenda programación en SQL Server 2000 Ya. Rebecca M. Riordan. Páginas 311-363

[39] http://www.abcdatos.com/webmasters/tutorial/o899.html