UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

"PRUEBAS DE HACKING ÉTICO EN UN LABORATORIO DE LA FACULTAD DE INGENIERÍA DE LA UNAM"

FI - UNAM

TESIS

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMPUTACIÓN

PRESENTAN:

LUIS HUGO FLORES ROMÁN
GUSTAVO GABRIEL HERNÁNDEZ HERNÁNDEZ

DIRECTORA DE TESIS: ING. MARÍA EUGENIA MACÍAS RÍOS

MAYO 2011.

Agradecimientos

A la Universidad Nacional Autónoma de México, por otorgarme la oportunidad de formar parte de su comunidad.

A mis profesores, por transmitirme parte de sus conocimientos con tanto interés y dedicación.

A mi familia, por su apoyo, aliento e insistencia.

A la Ing. María Eugenia Macías Ríos, por su paciencia y conducción.

A mis compañeros y amigos.

Muchas gracias.

Luis Hugo Flores Román

Quiero agradecer primeramente a mi familia, sin cuyo apoyo jamás habría llegado hasta estas instancias. Su soporte me da fuerza, su recuerdo determinación y su cariño valor. Gracias a todos ellos.

Asimismo, quiero agradecer a la Facultad de Ingeniería de la Universidad Nacional Autónoma de México y a sus profesores por brindarme la oportunidad de crecer profesionalmente, gracias a todos ellos por haberme brindado la oportunidad de recibir sus conocimientos y con ello, haberme motivado a seguir adelante.

Por último, quisiera agradecer a nuestra tutora de tesis, Ing. María Eugenia Macías Ríos por el tiempo dedicado y las facilidades brindadas sin las cuáles, esta tesis no habría sido posible.

Gustavo Gabriel Hernández Hernández

Pruebas de Hacking Ético en un laboratorio de la Facultad de Ingeniería de la UNAM

Contenido

Introducción

Justificación y objetivo

Actualmente, la información tiene una relevancia crítica en el mundo empresarial. El avance en la tecnología ha permitido que un gran número de personas tengan acceso a ésta y a los servicios de las organizaciones, tanto gubernamentales como educativas. Dicha información se genera, procesa, transforma y almacena a través de diversas infraestructuras de red y, debido a que dichas tecnologías no se encuentran exentas a los errores o a los descuidos, los activos más valiosos de las empresas se exponen de manera constante a la utilización no autorizada por terceras personas. Considerando lo anterior, es evidente la importancia que adquiere el hecho de identificar las vulnerabilidades y mitigar las fallas de seguridad existentes en los sistemas.

Una de las maneras de prevenir y disminuir los riesgos antes mencionados es la realización de pruebas de penetración o Hacking Ético, con la finalidad de intentar comprometer la red de la organización y así evaluar su seguridad previamente.

El desarrollo de este proyecto se llevó a cabo dentro de la red de un laboratorio que forma parte de la Facultad de Ingeniería de la Universidad Nacional Autónoma de México. Éste cuenta con 11 equipos, un *switch* (siguiendo la topología estrella), un *router* inalámbrico y un *servidor*, entre otros dispositivos. Sus servicios de red, que en el pasado han sufrido ataques y bloqueos, proporcionan información a los estudiantes de la carrera de Ingeniería en Computación para su formación educativa.

Es por ello que el objetivo de este proyecto de tesis es llevar a cabo las pruebas pertinentes, a través del uso de herramientas para encontrar las vulnerabilidades y evaluar la seguridad de la red del laboratorio antes mencionado pretendiendo con ello hacer las recomendaciones necesarias para mitigar o solucionar tales vulnerabilidades.

Aplicación

El campo de aplicación de este proyecto es muy amplio, ya que puede ser implementado en sistemas computacionales pertenecientes tanto a organizaciones públicas como privadas, en donde la manipulación de información crítica no sólo es frecuente, sino indispensable para el buen desarrollo de las actividades diarias.

Método

Las actividades realizadas se basan en el concepto de *pentesting*, que consiste en simular un ataque real a los sistemas que se fijan como objetivo, con la finalidad de evaluar las vulnerabilidades existentes en ellos y, finalmente, mejorar la seguridad de la información de una entidad.

Para tales fines, desarrollamos las pruebas siguiendo las siguientes fases:

- Investigación previa.
- Pruebas de penetración a la red interna.
- Análisis de resultados.
- Recomendaciones y conclusiones.

Herramientas

Para realizar una prueba de penetración es necesario contar con una serie de aplicaciones que permitan realizar ataques a dispositivos específicos además de la evaluación de los resultados. Las herramientas seleccionadas para tal fin serán enumeradas y descritas más adelante cuando se defina el plan de pruebas habiéndose seleccionado de un amplio catálogo las que pensamos, cumplirán de un mejor modo nuestras expectativas.

Perspectivas y contribuciones

Con la correcta preparación y puesta en práctica del proyecto se pretende, en primera instancia, identificar posibles fallas de seguridad que comprometan en mayor o menor medida la integridad de la servicios de red e información del laboratorio de computación de la Facultad de Ingeniería de la UNAM.

Justo es decir que además se obtendrá una valiosa experiencia en el manejo práctico de algunas de las herramientas utilizadas en el mundo de la seguridad informática aunque, si bien es cierto que no se habrán de manejar todas las herramientas disponibles, se realizó un gran esfuerzo por seleccionar de una larga lista las más usadas y populares con el fin de realizar una adecuada valoración de la seguridad.

Por otro lado, se pretende también que la información obtenida en las fases de análisis de resultados y recomendaciones, sea lo bastante trascendente como para ser tomada en cuenta por el administrador del laboratorio logrando así, contribuir a solucionar las vulnerabilidades críticas que posiblemente sean encontradas y así mitigar el impacto de un ataque.

Tema 1. Fundamentos teóricos

1.1 Introducción

Desde el comienzo mismo del desarrollo de la humanidad, la comunicación, en cualquier forma que esta se presentase, ha resultado indispensable para el crecimiento y avance de cada aspecto de la civilización humana, pasando en su evolución por diferentes niveles de desarrollo de acuerdo con cada etapa en particular.

El desarrollo de las comunicaciones, auspiciado por los avances en electrónica de la mitad del siglo XX, ha propiciado desde entonces la evolución de la comunicación a distancia, así como diversas técnicas para garantizar su confidencialidad e integridad, entre otras.

A raíz del impetuoso avance de tecnologías tales como las telecomunicaciones, la información ha sufrido una radical transformación que permite observarla no sólo en cuestión de una comunidad o nación en particular, sino como un fenómeno de alcance mundial que precisa de normas reguladoras que protejan información sensible e importante.

Dentro de este indetenible desarrollo de la tecnología de la información, el uso de "Internet" juega un papel preponderante. Concebida originalmente como un proyecto de aplicaciones militares, su desarrollo fue rápidamente conglomerando computadoras enlazadas entre sí mediante protocolos de comunicación; perdiendo así su carácter militar para convertirse en una red global de comunicación, información, educación y ocio, entre muchas otras, a distancia.

La utilización masiva de computadoras y redes como dispositivos de almacenamiento, transferencia y procesamiento de información se ha incrementado exponencialmente en los últimos años, llegando a convertirse en un elemento indispensable en la sociedad actual; consecuentemente, la información en cualquiera de sus formas se ha convertido en un activo de gran valor e importancia que debe ser protegido y resguardado de influencias dañinas provenientes tanto del exterior como del interior y en cualquier forma que ésta se presente. Es por ello, que a continuación presentamos una serie de definiciones, normas y ejemplos que nos ayudarán a comprender mejor el mundo de la seguridad informática.

1.2 Concepto de seguridad informática

La seguridad informática es aquella disciplina que se relaciona a técnicas, aplicaciones y dispositivos encargados de asegurar la integridad y privacidad de la información de un sistema informático y sus usuarios. [1]

Técnicamente es imposible lograr un sistema informático cien por ciento seguro, pero buenas medidas de seguridad evitan daños y problemas que puedan ocasionar intrusos.

[1] "Introducción a la seguridad informática" en http://es.kioskea.net/contents/secu/secuintro.php3, 12/09/2009.

1.2.1 Seguridad con respecto a la naturaleza de la amenaza

Existen dos tipos de seguridad con respecto a la naturaleza de la amenaza:

- Seguridad lógica. Es la aplicación de procedimientos adecuados para evitar el acceso a los recursos del sistema por parte de personas no autorizadas, ya sea a nivel local o vía red. Ejemplos de lo anterior son las aplicaciones para seguridad, herramientas informáticas, por citar algunos. [2] (Véase figura 1.)

Figura 1. Logos de algunos de los más importantes antivirus del mercado, los antivirus son esenciales en cuanto a la seguridad lógica de un equipo se refiere. [3]

- Seguridad física. Se refiere a los controles y mecanismos de seguridad implementados para proteger el hardware y medios de almacenamiento de datos. Por ejemplo, el mantenimiento de las instalaciones eléctricas y anti-incendio, prevención de la humedad, entre muchos otros. [4] (Véase figura 2.)

Figura 2. Componentes básicos de un sistema anti-incendio, detectores de humo, extintor de polvo, manta ignífuga. [5]

[2] "Definición de Seguridad informática" en http://www.alegsa.com.ar/Dic/seguridad%20informatica.php, 12/09/2009.
[3] Tomada de http://www.blog.agenciabanana.com/programas/122-antivirus-.html
[4] "Definición de Seguridad informática" en http://www.alegsa.com.ar/Dic/seguridad%20informatica.php, 12/09/2009.
[5] Tomada de http://www.inforsecuritel.com/default.php?cPath=1_101

1.2.2 Amenazas a la seguridad de un sistema informático

- Programas malignos. Virus, gusanos, *phising*, *spamming*, sólo por mencionar algunos.

- Siniestros. Robos, incendios, humedad, entre otros, pueden provocar pérdida de información.

- Intrusos o piratas informáticos pueden acceder remotamente (en el caso de que se esté conectado a una red) o físicamente a un sistema para provocar daños.

- Usuarios. Los mismos usuarios de un sistema pueden debilitar y ser una amenaza a la seguridad de un sistema, no sólo por boicot, sino también por falta de capacitación o desidia. [6] (Véase figura 3.)

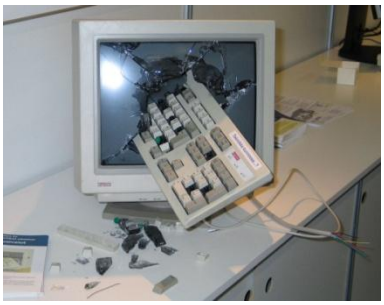


Figura 3. Muchas veces son los mismos usuarios autorizados la mayor amenaza de un sistema de información. [7]

1.3 Objetivos de la seguridad informática

Generalmente, la seguridad informática consiste en garantizar que el material y los recursos de software de una organización se usen únicamente para los propósitos para los que fueron creados y dentro del marco previsto.

Los objetivos de la seguridad informática se pueden resumir en la garantía de los seis servicios de seguridad:

- Integridad. Garantizar que los datos sean los que se supone que son.

- Confidencialidad. Asegurar que sólo los individuos autorizados tengan acceso a los recursos que se intercambian.

- Disponibilidad. Garantizar el correcto funcionamiento de los sistemas de información.

- No repudio. Certificar que una operación realizada no pueda ser negada.

[6] "Introducción a la seguridad informática" en http://es.kioskea.net/contents/secu/secuintro.php3, 12/09/2009.
[7] Tomada de http://www.tomandang.com/blog/images/smashedComputer.jpg

- Control de acceso. Avalar que sólo los individuos autorizados tengan acceso a los recursos. (Véase Figura 4.)

- Autenticación. Verificar que los individuos sean quienes dicen ser. [8]

Figura 4. La lectura de la huella digital es un medio para autenticar la identidad de una persona. [9]

Como ya se mencionó, los puntos anteriores tienen el objetivo de proteger los activos de las organizaciones. Son tres los elementos que los conforman:

- Información. Es el objeto de mayor valor para una organización, el objetivo es el reguardo de la misma, independientemente del lugar en donde se encuentre registrada, en algún medio electrónico o físico.

- Equipos. Software, hardware y organización.

- Usuario. Individuos que utilizan la estructura tecnológica y de comunicaciones que manejan la información. [10]

1.4 Amenazas y vulnerabilidades

1.4.1 Conceptos

Por vulnerabilidad entendemos la exposición latente a un riesgo u amenaza. En el área de la informática existen varias amenazas tales como: ataque de virus, códigos maliciosos, gusanos, caballos de Troya y *hackers*. Con la adopción de Internet como instrumento de comunicación y colaboración, los riesgos han evolucionado y ahora las empresas deben enfrentar ataques de denegación de servicio y amenazas combinadas; es decir, la integración de herramientas automáticas de *hackeo*, accesos no autorizados a los sistemas y capacidad de identificar y explotar las vulnerabilidades de los sistemas operativos o aplicaciones para dañar los recursos informáticos. De lo anterior, podemos definir amenaza como un factor externo de riesgo, representado por la posibilidad de que ocurra

[8] "Introducción a la seguridad informática" en http://es.kioskea.net/contents/secu/secuintro.php3, 12/09/2009.
[9] Tomada de http://www.masternewmedia.org/es/2005/05/15/la_red_de_autoidentificacion_para.htm
[10] "Introducción a la seguridad informática" en http://es.kioskea.net/contents/secu/secuintro.php3, 12/09/2009.

un fenómeno o evento adverso que podría generar daño parcial o total en bienes o en los servicios. [11]

1.4.2 Clasificación de amenazas y vulnerabilidades

Las amenazas y vulnerabilidad se clasifican como sigue:

A. Fuentes de amenazas

Las amenazas provienen de cinco fuentes principales:

- Desastres naturales. Popularmente conocidos como actos de Dios debido a que el hombre no puede controlar su ocurrencia ni predecirlos; son desastres naturales los huracanes, terremotos, maremotos, por ejemplo.

- Errores de hardware. Se refieren a las fallas físicas en cualquiera de los dispositivos involucrados; la falla puede ser parcial o total.

- Errores de software. Se refieren a las posibles fallas debido a incorrectas implementaciones en el sistema o a vulnerabilidades en el código fuente del software, aquí intervienen los códigos maliciosos (virus, gusanos, caballos de Troya, entre otros) que explotan dichas vulnerabilidades.

- Errores de red. Se presentan debido a un mal diseño, uso o implementación de la vía de comunicación informática.

- Humana. Se presenta por la ignorancia, diversión, descuido, malicia o indiferencia del usuario. [12]

B. Tipos de vulnerabilidades

Los tipos de vulnerabilidades, de manera muy similar a las amenazas, se clasifican como sigue:

- Desastres naturales. Se refiere al grado en que cierto sistema pudiera verse afectado por algún incidente natural. Se podría contar como una vulnerabilidad de este tipo cuando no se contara con un sistema adecuado de ventilación y nuestros equipos trabajen en condiciones climáticas adversas como podría ser un clima caluroso y húmedo.

- De Hardware. Se trata de la utilización de equipo en mal estado, inadecuado para el trabajo o bajo condiciones que no son las propicias. Como ejemplo, podemos considerar el uso de equipo de cómputo sin el uso de regulador en una zona con altas variaciones de voltaje.

[11] María Jaquelina López Barrientos, Cintia Quezada Reyes, *Fundamentos de seguridad informática*, México, Universidad Nacional Autónoma de México, Facultad de Ingeniería, 2006. p. 90.
[12] *Idem*, p. 91.

- De Software. Entendemos que existe vulnerabilidad de software cuando se emplean aplicaciones con carencias a nivel de programación que las hacen más susceptibles a ser atacadas exitosamente, este tipo de vulnerabilidades también engloban a los sistemas operativos. Un ejemplo son las páginas Web que requieren autenticación por parte de sus usuarios y que no cuentan con protección contra ataques de tipo *sql injection:* un usuario maligno podría ver, modificar o aun borrar su base de datos.

- De Red. Al conectar cualquier equipo a una red, se incrementa de forma exponencial el riesgo al que estará expuesto, y es que aumenta el número de personas que podrían tener acceso a él ya sea de forma legítima o no. Además, una pobre implementación en la estructura y diseño del cableado estructurado de la misma podría llevar a su colapso con todas sus implicaciones.

- Humana. Se refiere al papel que juegan los usuarios o personal, cuando por desidia, malestar, diversión, ignorancia o simple ocio, comprometen la seguridad del sistema. No contar con un departamento de recursos humanos eficiente, reglamento claro y a la vista de todos, no capacitar a nuestros usuarios o empleados para que eviten ciertas prácticas riesgosas, así como el no contar con un sistema de control de acceso a las instalaciones, son fuentes de vulnerabilidad humana.

- Física. Esta se refiere al emplazamiento en el cual se encuentra el equipo informático, un ejemplo de vulnerabilidad física podría ser, por ejemplo, el no considerar que el edificio en el que se instalará nuestro sistema tenga cimientos firmes, si cuenta con un sistema contra incendios adecuado, entre otros. [13]

1.5 Políticas de seguridad

Actualmente la importancia que la información tiene dentro de una organización es sumamente grande. Las necesidades y retos que las empresas deben satisfacer en este rubro cambian constantemente a la par de la evolución de los medios de transmisión existentes, de las peticiones de sus clientes o de sus socios.

Una de las tareas clave dentro de las organizaciones debe ser proteger su información de riesgos que comprometan su integridad, confidencialidad o disponibilidad, esto a través de normas o políticas dirigidas tanto a los sistemas que manipulan los datos como a cada uno de los miembros de la empresa.

Una política de seguridad es un conjunto de leyes, reglas y prácticas que regulan la manera de dirigir, proteger y distribuir recursos en una organización para llevar a cabo los objetivos de seguridad informática dentro de la misma. [14]

[13] *Idem*, pp. 100-103.
[14] *Idem*, pp. 127-131.

Al redactarse debe considerarse la visión de la empresa involucrada, las potenciales amenazas a las que la información de ésta estará expuesta y, sobre todo, debe ser escrita en un lenguaje que todos y cada uno de los miembros de la organización entiendan. Además, debe atender a cada uno de los principios siguientes:

- Responsabilidad individual. Los miembros de la organización deben estar conscientes de todas sus actividades, ya que éstas serán registradas y examinadas.

- Autorización. Son las reglas que especifican quién, cuándo y cómo puede acceder a la información.

- Mínimo privilegio. Los miembros de la organización deben tener los permisos mínimos necesarios para realizar sus funciones.

- Separación de obligaciones. Las funciones deben estar divididas entre todo el personal relacionado con la misma actividad o función.

- Auditoría. Las actividades del personal deben ser monitoreadas desde el inicio y hasta su término.

- Redundancia. Deben guardarse varias copias de información importante en varios lugares.

- Reducción de riesgo. Se debe reducir, dentro de lo posible, todos los riesgos a un nivel aceptable. [15]

1.6 Normas de seguridad a través de la historia

1.6.1 ITIL

Desarrollado inicialmente en 1980 por el gobierno británico, ITIL (Information Technology Infrastructure Library) es un set de 8 libros (en su versión 2, liberada entre los años 2000 y 2001) que describe conceptos y buenas prácticas concernientes a la administración, desarrollo y operación de las tecnologías de la información. [16]

Cada volumen cubre un área específica de la gestión de servicios:

- Soporte al servicio.

- Provisión del servicio.

- Administración de la infraestructura de las tecnologías de la información y comunicaciones.

[15] *Idem*.
[16] "What is ITIL?" en http://www.itil-officialsite.com/AboutITIL/WhatisITIL.aspx, 01/05/2011.

- Administración de la seguridad.
- La perspectiva de negocio.
- Administración de las aplicaciones. Señala buenas prácticas
- Administración de los activos de software.
- Planificación para implementar la administración de los servicios.

A. Administración de la seguridad en ITIL

La conforman los siguientes 6 procesos.

- Control. Define los procesos subsiguientes, la asignación de responsabilidades, las políticas y el marco de la administración en general.
- Planeación. Detallas las medidas a tomar para elaborar los planes de seguridad correspondientes a cada unidad de las organizaciones.
- Implementación. Tiene el fin de asegurar que todas las medidas especificadas en el subproceso anterior sean implementadas correctamente.
- Evaluación. Mide el éxito en la implementación de los planes de seguridad.
- Mantenimiento. Evalúa la posible actualización de los riesgos de seguridad debida a cambios en la infraestructura de las tecnologías de la información o en la misma organización.
- Modelo completo procesos-datos. Documenta la integración de todos los subprocesos y, por lo tanto, de la infraestructura de seguridad implementada. [17]

1.6.2 TCSEC

TCSEC (Trusted Computer System Evaluation Criteria), también conocido como "Libro Naranja", es un estándar desarrollado en 1983 por el Departamento de Defensa de los Estados Unidos que tiene como objetivo establecer los requerimientos mínimos necesarios para evaluar la seguridad de un sistema computacional dedicado a almacenar y procesar información clasificada. [18] (Véase figura 5.)

Define siete criterios de evaluación y en cada uno de ellos se considera la política de seguridad, la rendición de cuentas, el aseguramiento y la documentación:

[17] *Ídem*.
[18] "TCSEC" en https://www.ccn-cert.cni.es/publico/2008/401/es/t/tcsec.htm, 17/09/2011.

- Clase D: protección mínima.
- Clase C: protección discrecional.
  i. C1: protección de seguridad discrecional.
  ii. C2: protección de acceso controlado.
- Clase B: protección obligatoria.
  i. B1: protección de seguridad etiquetada.
  ii. B2: protección estructurada.
  iii. B3: dominios de seguridad.
- Clase A: protección verificada.
  i. A1: diseño verificado. [19]

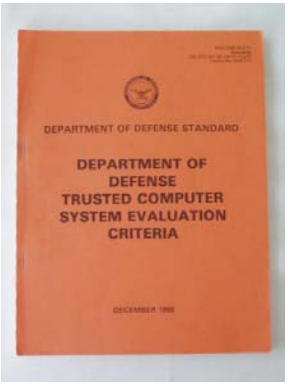


Figura 5. Portada de la norma TCSEC o "libro naranja". [20]

1.6.3 ITSEC

ITSEC (Information Technology Security Evaluation Criteria) es otro de los estándares existentes para la evaluación de la seguridad dentro de productos y sistemas. Fue publicado en conjunto por Francia, Alemania, Holanda y el Reino Unido en 1990 y la validez de criterios de su versión 1.2 fue reconocida por la Unión Europea en 1991. [21]

Las características de seguridad del producto a ser evaluado (llamado objetivo de evaluación) son sometidas a un gran número de pruebas funcionales y de penetración, cuyo nivel de exigencia irá creciendo de acuerdo con el nivel de

[19] "Trusted Computer System Evaluation Criteria, Orange Book" en http://nsi.org/Library/Compsec/orangebo.txt, 17/09/2011.
[20] Tomada de http://upload.wikimedia.org/wikipedia/en/4/4f/Orange-book-small.PNG
[21] "Information Technology Security Evaluation Criteria (ITSEC)" en http://www.ssi.gouv.fr/site_documents/ITSEC/ITSEC-uk.pdf, 18/09/2009.

confianza certificada que el objetivo desee. Estos niveles de exigencia están definidos en 7 criterios de evaluación denotados desde E0 hasta E6 y en cada uno de ellos se consideran los puntos siguientes:

- Construcción. El proceso de desarrollo.
  i. Fase 1. Requerimientos.
  ii. Fase 2. Diseño de arquitectura.
  iii. Fase 3. Diseño Detallado.
  iv. Fase 4. Implementación.
- Construcción. El entorno de desarrollo.
  i. Aspecto 1. Control de configuración.
  ii. Aspecto 2. Lenguajes de programación y compiladores.
  iii. Aspecto 3. Seguridad de los desarrolladores.
- Operación. La documentación operacional.
  i. Aspecto 1. Documentación de usuario.
  ii. Aspecto 2. Documentación de administrador.
- Operación. El entorno operacional.
  i. Aspecto 1. Envío y configuración.
  ii. Aspecto 2. Puesta en marcha y operación. [22]

1.6.4 COBIT

COBIT (Control Objectives for Information and related Technology), liberada originalmente en 1996 por la Asociación de Auditoría y Control de los Sistemas de Información, es un conjunto de buenas prácticas para la administración de las tecnologías de la información. [23]

Clasifica sus procesos en 4 dominios:

[22] *Ídem*.
[23] "COBIT Framework for IT Governance and Control" en http://www.isaca.org/Knowledge-Center/COBIT/Pages/Overview.aspx, 01/05/2011.

- Planificación y organización. Define la estrategia a seguir en el área de los sistemas de información, con la finalidad de proveer eficientemente los servicios que las diferentes áreas de negocio de las organizaciones requieran.

- Adquisición e implementación. Busca garantizar que la compra de aplicaciones comerciales, desarrollo de herramientas, la implementación y el mantenimiento de ambas se encuentre alineado con las necesidades del negocio.

- Entrega y soporte. Busca asegurar la eficiencia y la eficacia en la entrega de los servicios que las organizaciones requieren.

- Supervisión y evaluación. Se centra en validar la alineación de los sistemas de acuerdo a la estrategia del negocio y paralelamente, incluye la verificación de los controles por parte de auditores internos o externos. [24]

1.6.5 CTCPEC

CTCPEC (Canadian Trusted Computer Product Evaluation Criteria) es un estándar de seguridad canadiense que combina las aproximaciones y conceptos de TCSEC e ITSEC. Aborda tanto la funcionalidad como la confiabilidad de los productos a desarrollar o evaluar. [25]

- Funcionalidad. Proporciona las bases para desarrollar un método de especificación para productos de cómputo confiables y funcionales. Evalúa la efectividad de los servicios siguientes:

  i. Confidencialidad.

  ii. Integridad.

  iii. Disponibilidad.

  iv. Auditoría.

- Confiabilidad. Verifica la correcta implementación del producto basándose en las políticas de seguridad. Evalúa los siguientes requerimientos:

  i. Arquitectura.

  ii. Desarrollo ambiental.

  iii. Desarrollo de evidencias.

[24] *Ídem*.
[25] "Estándares de evaluación para sistemas de cómputo seguros" en http://mixtli.utm.mx/Estandares_de_Evaluacion_para_Sistemas_de_Computo_Seguros.ppt, 19/09/2009.

iv. Ambiente operacional.

v. Documentación.

vi. Seguridad.

vii. Seguridad en las pruebas. [26]

1.6.6 Criterios comunes

Los criterios comunes para la evaluación de seguridad de las tecnologías de la información, son una norma internacional basada en los criterios europeos, norteamericanos y canadienses existentes en el tema. Los resultados obtenidos al realizar una evaluación de este tipo son reconocidos internacionalmente. [27] (Véase figura 6.)

Especifican una evaluación de niveles de confianza para los productos y cada uno de éstos provee a consumidores, desarrolladores y evaluadores la información necesaria para determinar las necesidades de seguridad de los productos a adquirir, cubrir los requerimientos de los consumidores en el proceso de desarrollo de una aplicación o determinar el nivel de seguridad que ha alcanzado un producto.

Los niveles de aseguramiento dentro de los criterios comunes son los siguientes:

- EAL1. Es el nivel más bajo tanto para el desarrollador como para el usuario. Se basa en el análisis de las funciones de seguridad del producto, tal como son presentadas por el mismo.

- EAL2. Es el nivel de aseguramiento más alto que se le puede otorgar al desarrollador sin imponerle tareas adicionales. Un software de excelente calidad recibe esta certificación.

- EAL3. Es el nivel moderado de seguridad independiente y lo realiza una fuente externa. La seguridad se toma en cuenta desde la fase de diseño, no solamente cuando el producto está terminado.

- EAL4. Es el nivel de aseguramiento más alto, en el que es factible reparar una línea de productos ya existentes. En este nivel, un producto es diseñado, probado y revisado metódicamente. Dispone también de una búsqueda de puntos vulnerables.

- EAL5. No es común que los productos ya existentes alcancen esta clasificación, ya que deben ser diseñados con tal fin. El desarrollador debe de dar un enfoque riguroso de seguridad, teniendo en cuenta las especificaciones de diseño y su modo funcional en el producto.

[26] Ídem.
[27] María Jaquelina López Barrientos, Cintia Quezada Reyes, Fundamentos de seguridad informática, México, Universidad Nacional Autónoma de México, Facultad de Ingeniería, 2006. pp. 27-30.

- EAL6. Incluye todos los elementos del nivel EAL5, pero garantizando un alto grado de resistencia a ataques. Exige también:

  i. Proceso de desarrollo estructurado.

  ii. Controles de desarrollo.

  iii. Controles de manejo de comunicación.

- EAL7. Está destinado a aquellas aplicaciones de seguridad en las que el alto riesgo de violaciones y ataques justifiquen el alto costo de desarrollo. Es un proceso exhaustivo y la dependencia evaluadora debe participar desde la concepción de la idea hasta la finalización del proyecto. [28]

Figura 6. Logo de la norma "Criterios comunes". [29]

1.6.7 ISO 27002

Tiene sus orígenes en el estándar británico BS7799 publicado en 1995. En el año 2000 fue adoptado y publicado nuevamente por ISO/IEC como ISO 17799 y, tras ser objeto de una revisión en el año 2005, fue renombrado como ISO 27001. Con la finalidad de alinearlo al resto de los estándares de la serie 27000, en 2007 cambia su nombre a ISO 27002, que se mantiene en la actualidad. [30]

Es una guía de buenas prácticas que describe los objetivos de control y controles recomendables en cuanto a seguridad de la información. No especifica los requisitos necesarios para el establecimiento de un sistema de certificación adecuado para este documento. Contiene 39 objetivos de control y 133 controles, agrupados en los siguientes 11 dominios:

- Política de seguridad.
- Aspectos organizativos para la seguridad.
- Clasificación y control de activos.
- Seguridad ligada al personal.
- Seguridad física y del entorno.

[28] *Ídem*, pp. 54-62.
[29] Tomada de http://www.cse.fau.edu/~maria/CommonCriteria_logo.gif
[30] "ISO 27000" en http://www.iso27000.es/iso27000.html, 21/09/2009.

- Gestión de comunicaciones y operaciones.
- Control de accesos.
- Desarrollo y mantenimiento de sistemas.
- Gestión de incidentes de seguridad de la información.
- Gestión de continuidad de negocio.
- Conformidad. [31]

1.7 Servicios de seguridad

Un servicio de seguridad es "*aquel que mejora la seguridad de un sistema de información y el flujo de la información de una organización*" [32]. Tienen la finalidad de evitar ataques y utilizan uno o más mecanismos de seguridad para proveer el mismo.

Los servicios de seguridad se clasifican de la siguiente manera:

- Confidencialidad. Protegen la información y el almacenamiento de la misma para prevenir que nadie pueda leer, copiar o modificar la información sin los permisos necesarios para hacerlo.

  La forma más común de lograr los objetivos antes mencionados es mediante la utilización del cifrado basado en criptografía. (Véase figura 7.)

Figura 7. Ejemplo de funcionamiento de un método de cifrado. [33]

- Autenticación. Tiene la finalidad de asegurar que una comunicación es auténtica. Provee al sistema de una prueba de que realmente se es quien se pretende ser.

  Generalmente se realiza mediante:

[31] *Ídem*.
[32] María Jaquelina López Barrientos, Cintia Quezada Reyes, *Fundamentos de seguridad informática*, México, Universidad Nacional Autónoma de México, Facultad de Ingeniería, 2006, p. 116.
[33] Tomada de http://www.itchetumal.edu.mx/paginasvar/Maestros/redes1/unidad9/unidad9.htm

i. Algo que se sabe: Por ejemplo, una contraseña o número de identificación personal que el sistema compara con una copia almacenada una vez que se ingresa.

ii. Algo que se tiene: Por ejemplo, una tarjeta que el sistema utilizará para verificar la identidad. (Véase figura 8.)

iii. Algo que se es: Cualquier parte biológica como, por ejemplo, la voz, la retina o una huella digital.

Figura 8. La credencial de elector es un ejemplo típico de una forma de autenticación. [34]

- Integridad. Permite asegurar que el contenido de los datos no haya sido modificado, si esta no se garantizara, entonces se corre el riesgo de que cualquier persona puede manipular los mismos según su conveniencia.

  Este servicio se relaciona más con la detección que con la prevención, ya que una vez que una violación de integridad es detectada, se necesita de la partición humana o de otro software para recuperarse de tal violación.

  Los mecanismos de seguridad de este tipo más utilizados son los siguientes:

  i. Código de detección de modificación. Es una suma de comprobación de los datos generada utilizando un algoritmo criptográfico.

  ii. Código de autenticación del mensaje. Es una suma de comprobación cifrada de los datos generada con base en la criptografía.

  iii. Firma digital. Es un tipo de información asociada con los datos que solamente puede ser creada por el firmante y puede ser verificada por cualquier persona. [35] (Véase figura 9.)

[34] Tomada de http://estafadoresdeinternet.blogspot.com/2009_04_02_archive.html
[35] María Jaquelina López Barrientos, Cintia Quezada Reyes, *Fundamentos de seguridad informática*, México, Universidad Nacional Autónoma de México, Facultad de Ingeniería, 2006, pp. 120-122.

Figura 9. La firma digital puede vincularse a un documento para identificar al autor, para señalar conformidad o disconformidad, que no se ha modificado su contenido, etc. [36]

- No repudio. Previene a los emisores o a los receptores de negar un mensaje transmitido, se aplica al problema de la denegación falsa de la información que se recibe de otros o de la que alguien envía a otro. Se clasifican en los siguientes:

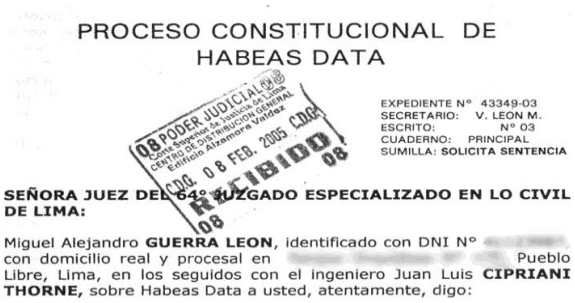
  i. De origen. Provee pruebas del origen de los datos para prevenir denegación falsa en el suministro.

  ii. De envío. Provee pruebas del envío de los datos para prevenir denegación falsa en la recepción.

  iii. De presentación. Provee pruebas de presentación de los datos para prevenir denegación de que la información fue presentada para el envío.

  iv. De transporte. Provee pruebas del transporte de los datos para prevenir que se niegue que los mismos fueron trasladados.

  v. De recepción. Provee pruebas de la recepción de los datos para prevenir que se niegue que la información ha sido recibida. [37] (Véase figura 10.)



PROCESO CONSTITUCIONAL DE HABEAS DATA

PODER JUDICIAL
CENTRO DE DISTRIBUCIÓN GENERAL
Edificio Alzamora Valdez
CDG 08 FEB. 2005 CDG
RECIBIDO
08

EXPEDIENTE N° 43349-03
SECRETARIO: V. LEON M.
ESCRITO: N° 03
CUADERNO: PRINCIPAL
SUMILLA: **SOLICITA SENTENCIA**

SEÑORA JUEZ DEL 64° JUZGADO ESPECIALIZADO EN LO CIVIL DE LIMA:

Miguel Alejandro **GUERRA LEON**, identificado con DNI N° [redacted], con domicilio real y procesal en [redacted] Pueblo Libre, Lima, en los seguidos con el ingeniero Juan Luis **CIPRIANI THORNE,** sobre Habeas Data a usted, atentamente, digo:

Figura 10. El documento sellado impide se niegue haber recibido el documento, esto es una forma de no repudio de recepción. [38]

[36] Tomada de http://www.kimaldi.com/area_de_conocimiento/firma_digital/que_es_la_firma_digital.
[37] María Jaquelina López Barrientos, Cintia Quezada Reyes, *Fundamentos de seguridad informática*, México, Universidad Nacional Autónoma de México, Facultad de Ingeniería, 2006, p. 122.

- Control de acceso. Es la habilidad para limitar y controlar el acceso a los sistemas y las aplicaciones mediante los puentes de comunicación. Para lograrlo, cada entidad que busca ingresar, debe autenticarse y así ganar los derechos de acceso que le correspondan. [39] (Véase figura 11.)

Figura 11. Una forma de control de acceso, mediante llave y cerradura. [40]

- Disponibilidad. Para prevenir que la información no esté disponible cuando se requiere, deben existir soluciones alternas con copias actualizadas de la información crítica y de los programas en un lugar diferente, además de un plan de continuidad que permita restablecer las operaciones a la brevedad. [41] (Véase figura 12.)

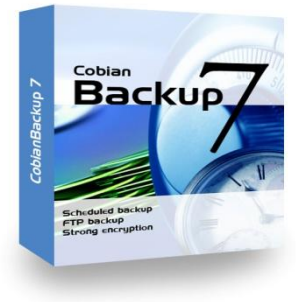


Figura 12. Realizar copias de seguridad de la información crítica garantiza su disponibilidad. [42]

[38] Tomada de http://www.gratisweb.com/miguelguerra/excomunion/habeasdata_3.htm
[39] "Definición de Seguridad informática" en http://www.alegsa.com.ar/Dic/seguridad%20informatica.php, 12/09/2009.
[40] Tomada de http://www.vidanuevalourdes.com/category/el-blog-de-alex/
[41] María Jaquelina López Barrientos, Cintia Quezada Reyes, *Fundamentos de seguridad informática*, México, Universidad Nacional Autónoma de México, Facultad de Ingeniería, 2006, p. 125.
[42] Tomada de http://grupogeek.com/wp-content/uploads/2008/12/26-12-cobian-backup.jpg

Tema 2. Identificación de ataques y técnicas de intrusión

2.1 Introducción

Como se ha visto con anterioridad, en años recientes con el auge de las telecomunicaciones, la información se ha convertido en un bien de extraordinario valor para muchas personas, y su resguardo ha trascendido del ambiente puramente físico, al electrónico. Los métodos, tanto de uno como de otro bando, son cada vez más sofisticados y complejos, una guerra silenciosa, que cada día, a cada hora, se torna más grande y ardua, se libra sin que se alcance a distinguir todavía un final a la misma.

Dadas las condiciones actuales en que la red mundial se encuentra, es imprescindible tener presente la mayor cantidad de información referente a la seguridad informática sin importar si se debe proteger una PC casera o una red corporativa. A la mayoría de los piratas informáticos realmente no les interesa de qué tipo se trate; si alguien está comunicado hacia cualquier tipo de red, es una víctima potencial de sus ataques.

Es por ello, que a continuación ahondamos en los significados de vulnerabilidad y amenaza y veremos someramente algunas de las técnicas más comunes para realizar sus ataques, recurrentes, pero lo realmente desconcertante de todo esto, es que la mayoría de las veces, efectivas, y es que en la mayoría de la sociedad la cultura de seguridad informática no existe, o bien, no se ha desarrollado de manera adecuada.

2.2 Identificación de vulnerabilidades

Como se ha visto anteriormente, una vulnerabilidad "*es la potencialidad o posibilidad de ocurrencia de la materialización de una amenaza sobre un activo*". [43] (Véase figura 13.)

Figura 13. Varias amenazas acechan en Internet. [44]

Para estimar la vulnerabilidad de determinado activo o grupo, es imperativa la participación del responsable y por lo tanto, conocedor de cada activo, además, es necesario que esté suficientemente informado por algún especialista para que comprenda o imagine objetivamente, el daño potencial causado por la acción directa de una amenaza.

Se pueden considerar tres tipos de vulnerabilidades:

[43] Toni Puig, "Gestión de riesgos de los sistemas de información" en http://www.mailxmail.com/curso-gestion-riesgos-sistemas-informacion/identificacion-vulnerabilidades-impactos, 12/02/2010.
[44] Tomada de http://www.dosbit.com/tag/gusano

- Vulnerabilidad intrínseca, es aquella que sólo depende del activo y de la amenaza en sí.

- Vulnerabilidad efectiva, es la resultante luego de que se aplican las medidas correspondientes.

- Vulnerabilidad residual, es la resultante luego de que se aplican las medidas complementarias.

El grado de vulnerabilidad se mide considerando la relación entre la amenaza potencial y su riesgo a materializarse como una acción real sobre el objetivo. También se deberá calcular la frecuencia de ocurrencia a partir de hechos objetivos (estadísticas de incidentes por ejemplo). [45]

Estudios previos, han arrojado la siguiente tabla estadística que clasifican a la vulnerabilidad según la ocurrencia (Véase tabla 1):

Rango de frecuencias	Vulnerabilidad
Superior a 6 años	Muy baja
Menor a 6 años	Baja
Aproximadamente 1 año	Media
Menos de 2 meses	Alta
Menos de 1 semana	Muy alta

Tabla 1. Relación de vulnerabilidad con su tasa de ocurrencia. [46]

Las vulnerabilidades se encuentran en cualquier sistema operativo, llámense Windows, Mac OS, Unix, Linux, OpenBSD, por mencionar algunos, u aplicación. La única manera de reducir la posibilidad de que una vulnerabilidad pueda ser explotada, es permanecer vigilantes y aplicar mantenimiento frecuente al sistema, implementar una arquitectura de seguridad, controles de acceso, así como auditorías de seguridad.

2.2.1 Barrido de puertos

Esta técnica nos ayuda a detectar la situación en la que se encuentra cierto puerto, esto es, si éste se encuentra abierto, cerrado o detrás de un *firewall*. Su fin es revelar los servicios que ofrece el equipo en cuestión y las posibles vulnerabilidades dependiendo de los puertos que se encuentren abiertos. También, mediante el uso de esta técnica, es posible identificar el sistema operativo que está utilizando la máquina. Existe una multitud de programas que realizan el barrido de los puertos, siendo unos de los más conocidos Nmap. (Véase figura 14.)

[45] "Introducción a la seguridad informática" en http://es.kioskea.net/contents/secu/secuintro.php3, 12/09/2009.
[46] "Definición de Seguridad informática" en http://www.alegsa.com.ar/Dic/seguridad%20informatica.php, 12/09/2009.

Figura 14. Logo del sitio insecure.org, lugar de desarrollo de la aplicación Nmap, figura que con frecuencia se asocia al programa. [47]

Esta técnica, se divide a su vez, en las siguientes:

A. Escaneo de puertos TCP

Al establecer una conexión *TCP (Transmission Control Protocol)*, necesariamente se sigue una negociación consistente en tres pasos, regularmente conocida como *Three-way-handshake*. Este protocolo se inicia con la máquina origen que envía un paquete con la bandera SYN activada; la máquina destino responde a su vez con las banderas SYN/ACK prendidas y por último, la computadora origen envía un tercer y último paquete conteniendo la bandera *ACK*; una vez completados dichos pasos, la conexión entre los dos equipos se ha completado. (Véase figura 15.)

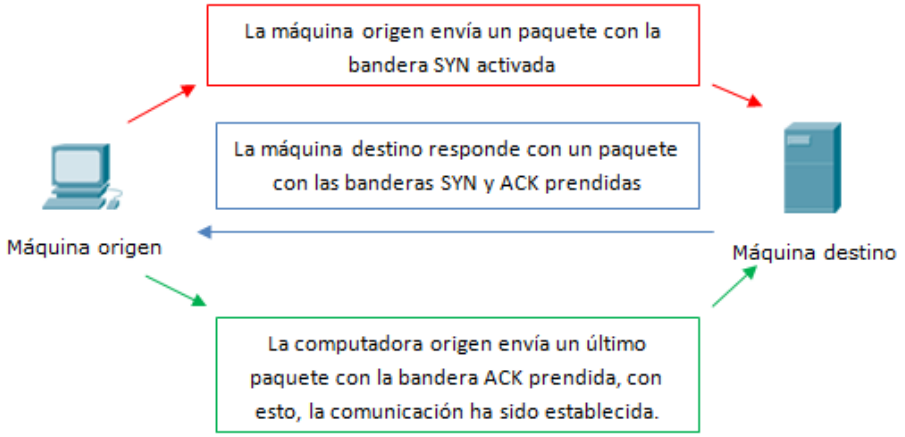


Figura 15. Representación gráfica de los pasos que se siguen durante el protocolo *Three-way-handshake*.

Un escáner de puertos *TCP* envía varios paquetes SYN al equipo que se está atacando y dependiendo de la respuesta que obtenga, interpreta lo siguiente:

- Si la respuesta es un SYN/ACK, entonces el puerto está abierto.
- Si la respuesta es un paquete RST, significará que el puerto está cerrado.

[47] Tomada de http://insecure.org/

- Por último, si lo que se obtiene es un paquete *ICMP (Internet Control Message Protocol)* como puerto inalcanzable, será porque dicho puerto está protegido por un *firewall*. [48]

Realizando dicho procedimiento en los puertos conocidos se tendrá una visión bastante completa del estado del equipo víctima.

B. Escaneo de puertos UDP

Aunque el protocolo *UDP (User Datagram Protocol)* es uno no orientado a conexión, si se manda un paquete a un puerto de estos y se encuentra cerrado, se obtendrá un mensaje de puerto inalcanzable, en cambio, si no obtienen respuesta, es posible inferir entonces que el puerto está abierto, aunque, si este está protegido por un *firewall*, entonces se obtendría información errónea.

Otra opción para descubrir puertos *UDP* es mandar paquetes de una aplicación específica con el fin de generar una respuesta de la capa de aplicación del modelo *OSI (Open System Intercconection)*. Por ejemplo, se podría mandar una consulta *DNS (Domain Name System)*.

Una forma de detectar un barrido de puertos es mediante la implementación de un *IDS (Intrusion Detection System)*, el cual, si detecta la firma de un barrido de puertos, lanzará una alarma y, en conjunción al *firewall*, se pondrá la IP del atacante en una lista negra que deniegue cualquier comunicación con la misma. [49]

2.2.2 Identificación de firewalls

Se podría definir a un *firewall* como un dispositivo informático, ya sea software o hardware que protege a una red privada al definir un perímetro de seguridad, y definírsele reglas de filtrado de paquetes.

La función principal del *firewall* es la de examinar paquetes en busca de coincidencias con las reglas que se le han fijado y dependiendo de ellas, permitirles o negarles el acceso. Además, los *firewalls* pueden también generar alarmas y crear listas negras. (Véase figura 16.)

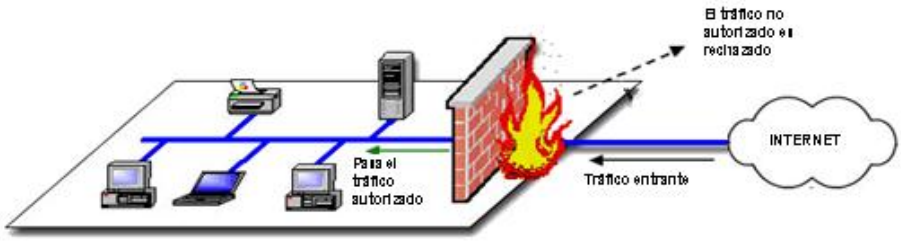


Figura 16. Forma de trabajo de un *firewall*, el paso de los paquetes es permitido o denegado según las reglas que se le hayan fijado. [50]

[48] Jayant Gadge, Anish Patil, "Port Scan Detection", en http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4772622, 30/04/2011.
[49] *Ídem*.
[50] Tomada de http://tindes.com/jj/index.php?option=com_content&view=article&id=41:seguridad-red&catid=1:actualidad&Itemid=27&0de28b3435550b272401162583c2c73f=tqyhgghphlnuox

Existen diferentes tipos de *firewall*:

- *Firewall* de filtrado por paquetes.
- *Firewall* de filtrado por estado.
- *Firewall* de filtrado por contenido o aplicación. [51]

A. Firewall de filtrado por paquetes

El *firewall* de filtrado por paquetes trabaja en la capa tres del modelo *OSI*, examina el encabezado de cada paquete analizando la procedencia y destino de los datos y en vista de esto decide si permite o no su paso.

Las reglas pueden basarse en:

- La dirección IP de origen o un intervalo de dirección IP.
- La dirección o direcciones IP de destino.
- El protocolo de red que usa.
- El número de puerto.
- El puerto de origen o destino. [52]

B. Firewall de filtrado por estado

Este permite guardar un registro de las conexiones existentes. Además, es capaz de trabajar con los protocolos tales como *IP*, *TCP*, *UDP*, *ICMP*, *FTP* e *IRC*. No trabajan filtrando paquetes individuales, sino sesiones enteras, permitiendo una optimización del trabajo de filtrado. [53]

C. Firewall de filtrado por contenido

También conocido como de filtrado por aplicación, analiza la trama a nivel de la capa de aplicación del modelo *OSI*, así, controla además de los puertos y sesiones, los protocolos que se utilizan para la comunicación evitando que se puedan suplantar servicios. Este tipo de *firewalls* implementa reglas más estrictas permitiendo un control mayor sobre las conexiones establecidas. Además, puede ser usado como servidor *proxy* para servicios tales como *HTTP (HyperText Transfer Protocol)*, *FTP*, entre otros.

Para la identificación de *firewalls* se utilizan diversas herramientas tales como el *firewalking*. El *firewalking* es una técnica que fue creada en 1998 con el objeto de descubrir las políticas de filtrado de un *firewall*. Lo que se hace básicamente es

[51] "Introducción a la seguridad informática" en http://es.kioskea.net/contents/secu/secuintro.php3, 12/09/2009.
[52] *Ídem*.
[53] "Definición de Seguridad informática" en http://www.alegsa.com.ar/Dic/seguridad%20informatica.php, 12/09/2009.

enviar un paquete con un *TTL (Time To Live)* fija, que debe expirar en el último *gateway*, antes de llegar al objetivo.

Cuando el paquete expira, y si se recibe un mensaje de error diciendo lo siguiente: exceed in-transit, significará que el paquete superó al *firewall*, pero no llegó al objetivo pues su *TTL* llegó a 0. Si no se recibe respuesta alguna, entonces el *firewall* habrá descartado al paquete puesto que no cumplía sus políticas de filtrado.

En la actualidad, esta técnica podría no funcionar, puesto que la mayoría de los *firewalls* no decrementan el *TTL* de los paquetes.

Una técnica popular para la obtención de información es el *banner grabbing*. Se le llama *banner* a la información que transmite un servicio cuando nos comunicamos con él, dicha información podría ser la versión, su nombre, etc. Así, mediante esta técnica es posible identificar los servicios que corren bajo un *firewall* y deducir, en consecuencia, cuáles son sus políticas de filtrado. [54]

2.2.3 Identificación del sistema operativo

Existen dos técnicas encaminadas a descubrir el sistema operativo que corre la máquina que está bajo ataque: la técnica activa y la pasiva. La activa opera bajo el principio de que cada sistema operativo responde diferente a una serie de paquetes mal formados. Lo que se hace es mandar a la máquina víctima estos paquetes y comparar su respuesta con una base de datos previamente hecha. Este método requiere de una conexión con la máquina víctima y su acción puede ser reconocida por un sistema *IDS*.

La identificación pasiva, por otra parte, captura paquetes provenientes del otro equipo con la ayuda de *sniffers*. Para determinar el sistema operativo del que se trata, esta técnica se basa en el principio de que todas las pilas *IP* se manejan de diferente manera para cada sistema operativo y así, analizando los paquetes capturados e identificando dichas diferencias, se podrá determinar el sistema operativo usado.

Existen áreas de los paquetes *TCP* capturados cuyo examen puede llevarnos a conocer el sistema operativo del que provienen, ejemplo de esto son:

- *TLL*. El número de saltos máximo que tiene un paquete antes de ser eliminado.
- Tamaño del *frame*.
- Si la bandera *DF (Don't Fragment)* está activada o no.

[54] "Firewalking" en http://www.webopedia.com/TERM/F/firewalking.html, 25/01/2010.

Analizando estos campos, es posible determinar el sistema operativo. Aunque no es 100% preciso, sí puede darnos un panorama bastante claro de lo que está corriendo en la máquina víctima.

Con esta técnica, puede evitarse el riesgo de ser detectado por un sistema *IDS*, además de que también pueden identificarse *firewalls* que trabajen como servidor *proxy*.

Esta técnica, como cualquiera, tiene ciertas limitaciones, como por ejemplo, el capturar paquetes de aplicaciones que construyen los suyos propios, un caso de esto es NMap. Y si se han modificado los valores del *TTL*, *DF*, entre otros, entonces no concordarán con ninguna base de datos, siendo esto una manera eficiente de protegerse contra dichos ataques. [55]

2.3 Explotación y obtención de acceso a sistemas y redes

La meta de todo atacante es el acceso al sistema víctima con los privilegios necesarios para la instalación y ejecución de software, o la modificación de archivos y su ubicación, así, hay diferentes técnicas que explotan las vulnerabilidades de aplicaciones, sistemas operativos, servicios, que nos permiten esto.

2.3.1 Robo de identidad

El robo de identidad o *phising* es básicamente un tipo de estafa en línea en el que se usan técnicas como el envío de correo no deseado (*spam*), sitios web falsos, mensajes instantáneos con el que pretenden engañar a los usuarios para que proporcionen información confidencial que puede ser desde datos de la tarjeta de crédito, cuentas bancarias, hasta contraseñas. (Véase figura 17.)

Existen varias recomendaciones para evitar ser víctima de este tipo de estafas:

- Si se recibe un e-mail o nos aparece un *pop-up* solicitando información personal o financiera, no responder ni hacer clic al enlace o vínculo del mensaje, puesto que, las compañías legítimas no solicitan este tipo de información vía e-mail.

- Por supuesto, utilizar antivirus y *firewalls* al navegar por la red.

- No enviar información personal o financiera a través del correo electrónico, puesto que no es un método seguro de transferencia de información.

- Revisar los resúmenes de las cuentas bancarias y tarjetas de crédito tan pronto como se reciban para verificar si se han imputado cargos no autorizados o no hechos.

[55] "Introducción a la seguridad informática" en http://es.kioskea.net/contents/secu/secuintro.php3, 12/09/2009.

- Tener cuidado al abrir o descargar documentos adjuntos a los mensajes recibidos, puesto que pueden contener *malware* que comprometan la seguridad de nuestro equipo. [56]

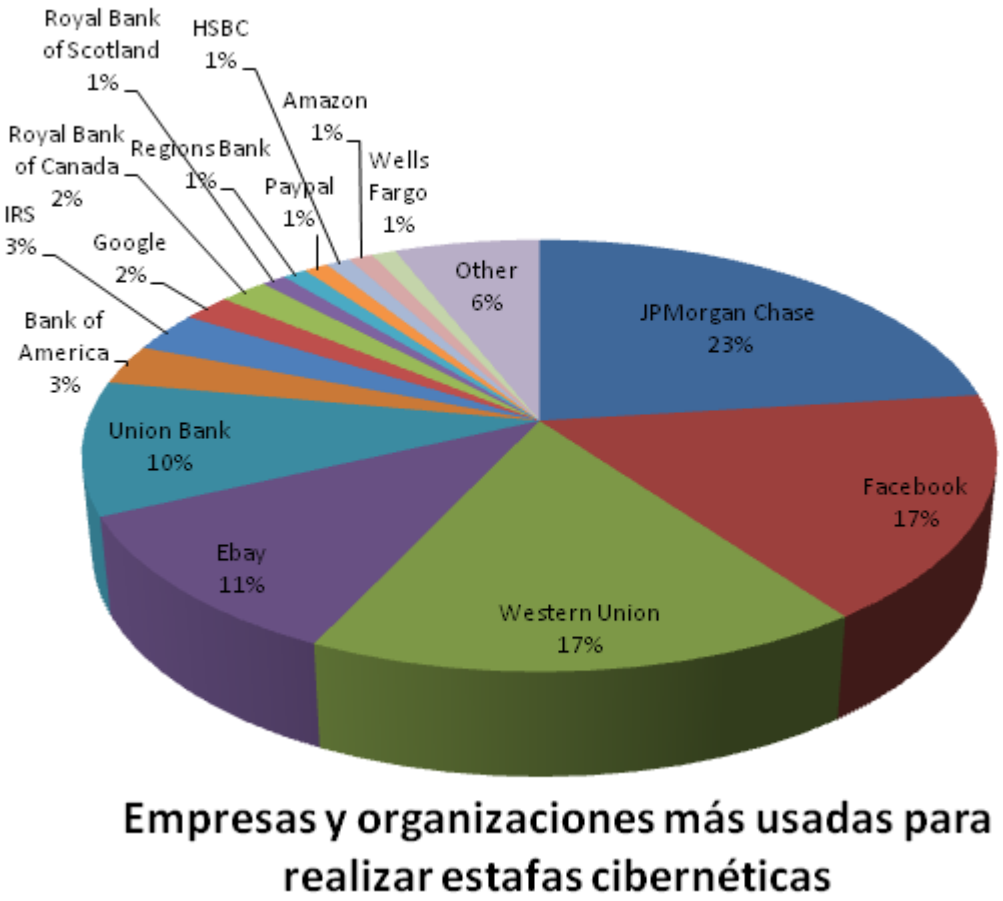


Figura 17. Relación de las organizaciones con presencia mundial, más utilizadas para realizar fraudes de *phising* en 2009. Diagrama de The Honey Pot Project. [57]

2.3.2 Engaño a firewalls e IDS's

La creación de "túneles" es una técnica popular utilizada para engañar la protección de un *firewall*. La técnica de *tunneling* consiste en encapsular un protocolo de red dentro de otro, permitiendo así superar los límites impuestos por algún *firewall.* (Véase figura 18.) Como ejemplo veamos dos aplicaciones que se comunican mediante el protocolo de transporte *TCP*. [58]

En el primer paso, las datos de la aplicación serán enviados al cliente "tunelizador" que será el encargado de encapsular estos datos dentro del protocolo *HTTP* (generalmente aceptado por la mayoría de los *firewalls*). La

[56] "¿Cómo evitar que te 'pesquen' en la red? Phising, un peligroso y moderno delito" en http://www.univision.com/content/content.jhtml?chid=9&schid=1860&secid=11068&cid=995274&pagenum=1, 10/10/2009.
[57] Tomada de http://www.pdatungsteno.com/2009/12/16/proyecto-honey-pot-colabora-en-la-lucha-mundial-contra-el-spam/
[58] "Secunia, stay secure" en http://secunia.com/, 13/10/2009.

aplicación y el cliente "tunelizador" no tienen que estar en la misma máquina, simplemente basta con que ésta sea accesible.

Después, el cliente "tunelizador" manda una petición *HTTP* estándar a un tercer agente, un "destunelizador" disfrazado como servidor WEB que tendrá el trabajo de desencapsular y mandar los datos a la aplicación destino.
Existen varias aplicaciones que permiten implementar esta técnica de evasión, por ejemplo: proxytunnel, HTTP-tunnel entre otros. para permitir crear un túnel usando el servicio de *SSH (Secure SHell)*, el cliente Putty puede servir. La herramienta NMap también implementa técnicas de evasión de *IDS's* y *firewalls*. [59]

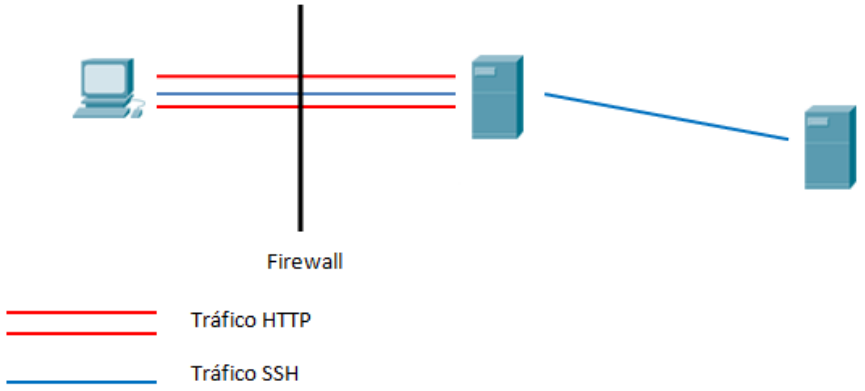


Figura 18. El tráfico *SSH* no permitido logra evadir al *firewall* encapsulado dentro de tráfico *HTTP* permitido.

2.3.3 Vulnerabilidades de software

Una vulnerabilidad de software puede definirse como un fallo o hueco de seguridad detectado en algún programa o sistema informático que puede ser utilizado bien por un virus para propagarse e infectar, o por *hackers* para entrar en los sistemas de manera no autorizada. De manera más sencilla, se trata de un fallo de diseño de algún programador que permite que un virus pueda realizar actividad maliciosa sin la necesidad de que el usuario actúe de forma directa. [60]

A continuación presentamos algunos casos de vulnerabilidad de software:

A. Buffer overflows

En un estudio generado por David Wagner, Jeffrey Foster, Eric Brewer y Alexander Aiken, mostraron que más del 50% de las vulnerabilidades explotadas eran de *buffer overflow*, también, el análisis demostraba que esta relación se incrementaba con el tiempo. [61]

[59] "Nmap – Free Security Scanner For Network Exploration & Security Audits" en http://nmap.org/, 23/10/2009.
[60] "Las vulnerabilidades de software" en http://www.pandasoftware.com/about/press/viewNews.aspx?noticia=4610, 27/09/2009.
[61] Aleph One, "Smashing the stack for fun and profit" en http://insecure.org/stf/smashstack.html, 29/09/2009.

Durante mucho tiempo se ha reconocido que los desbordamientos del *buffer* son un problema en lenguajes de bajo nivel. La esencia del problema es que los datos de usuario y la información del control de flujo del programa se mezclan en beneficio del desempeño, y los lenguajes de bajo nivel permiten el acceso directo a la memoria. C y C++ son los dos lenguajes más populares afectados por los desbordamientos del *buffer*.

Estrictamente hablando, ocurre un desbordamiento del *buffer* cuando un programa permite la escritura más allá del final del *buffer* asignado. Otra parte del problema ocurre cuando se permite que un atacante escriba en una ubicación arbitraria de memoria, fuera de la matriz de aplicación.

La pila

La pila es un tipo abstracto de datos. Una pila de objetos tiene la propiedad de que el último objeto posicionado en ella será el primero en ser removido. (Véase figura 19.)

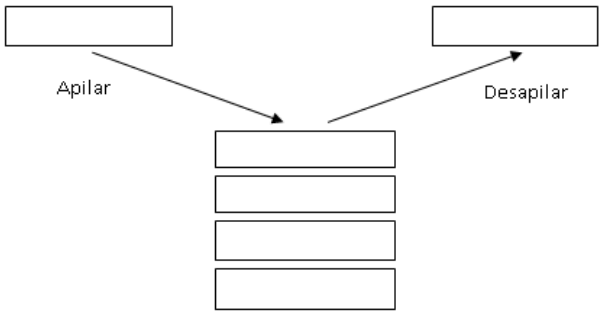


Figura 19. Representación de la organización de una pila.

Esta propiedad es comúnmente referida como último en entrar, primero en salir o LIFO en inglés (Last In, First Out queue).

Varias operaciones están definidas para la pila. Dos de las más importantes son PUSH y POP. La operación PUSH añade elementos al tope de la pila. Por el contrario, POP retira el último elemento del tope de la pila. [62]

Efectos de un desbordamiento de buffer

El efecto de un desbordamiento de *buffer* es amplio, desde la caída del sistema hasta la apropiación completa del control de la aplicación por parte del atacante; y si la aplicación se ejecuta como administrador o usuario raíz, entonces el atacante tendrá el control de todo el sistema operativo, así como de todos los usuarios que hayan iniciado sesión en el sistema. Si la aplicación desbordada es un servicio de red, el resultado del error podría ser un gusano. [63]

[62] An Zhiyuan, Liu Haiyan, "Realization of Buffer Overflow", 2010 International Forum on Information Technology an Applications en http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5635047, 30/04/2011.
[63] *Ídem.*

Explicación del desbordamiento del buffer

Al desbordamiento clásico del *buffer* se le conoce como "rompimiento de pila". En un programa compilado, la pila se utiliza para contener información de control como argumentos, a la que debe regresar la aplicación una vez que haya terminado con la función. Por desgracia, las variables que se asignan de manera local también se almacenan en la pila. Algunas veces, a estas variables se les identifica en forma imprecisa, como si se les asignara de manera estática, en lugar de la manera dinámica. La raíz del desbordamiento de *buffer* radica en que si la aplicación escribe más allá de los límites de una matriz asignada por la pila, el atacante tiene la posibilidad de especificar información de control. Y, esto es crítico para lograr su objetivo: el atacante busca la modificación de la información de control por valores que él mismo maneja.

Otra forma de *buffer overflow* que podría representar problemas de seguridad mucho más graves es el llamado *stack-smashing attack*, en el que un usuario malicioso puede tomar ventaja de un *buffer overflow* a través de la ejecución de código arbitrario. [64]

Métodos de prevención

- Reemplazar funciones peligrosas de manejo de cadenas.
- Reemplazar funciones inseguras como strcpy, strcat con las versiones válidas de cada una de estas funciones.

Reemplazar buffers de cadena de C con cadenas de C++

- Esto es más eficaz que sólo reemplazar las llamadas de C, pero es posible que cause gran cantidad de cambios en código existente, en particular si el código no está compilado como C++. [65]

B. Heap overflow

El *heap overflow* es un tipo de *buffer overflow* que se produce en un área de memoria denominada *heap*. La memoria *heap* se reserva dinámicamente con funciones tales como malloc() (subrutina para el ejercicio de asignación de memoria dinámica en los lenguajes de programación C y C++, es la abreviatura del inglés Memory Allocation) y puede ser sobreescrita de forma similar a la que se produce en los *buffer overflow*, es decir, en situaciones en las que el programador no verifica correctamente el tamaño de los datos que copia. [66]

Esta técnica es generalmente más difícil de explotar que el *buffer overflow*, por esta razón, es recomendable que los desarrolladores sigan como regla, no asignar de forma estática espacio para *buffer*.

---

[64] "Las vulnerabilidades de software" en http://www.pandasoftware.com/about/press/viewNews.aspx?noticia=4610 , 27/09/2009.
[65] *Ídem*.
[66] *Ídem*.

C. Vulnerabilidad de formato de cadena

Definición

La principal causa del error de formato de cadena es aceptar sin validar la entrada proporcionada por el usuario. En C es posible utilizar errores de formato de cadena para escribir en ubicaciones de memoria arbitrarias. El aspecto más peligroso es que esto llega a suceder sin la necesidad de manipular bloques de memoria adyacentes. Esta capacidad de diseminación permite a un atacante eludir protecciones de pila, e incluso modificar partes muy pequeñas de memoria. El problema también llega a ocurrir cuando las cadenas se leen a partir de una ubicación no confiable que controla el atacante. [67]

Localización de problemas de formato de cadena

Cualquier aplicación que tome una entrada del usuario y la pase a función de formateo está en riesgo. Un ejemplo muy frecuente de este problema sucede junto con aplicaciones que registran la entrada de usuario. Además, algunas funciones tal vez implanten formateo de manera interna.

En C habrá que buscar funciones de la familia printf. Entre los problemas que habrá de localizar se encuentran los siguientes:

```
printf (entrada_usuario);
fprintf (STDOUT, entrada_usuario);
```

Si se encuentra con una función con el siguiente aspecto:

```
fprintf (STDOUT, msg_format, arg1, arg2);
```

Se necesitará verificar dónde se almacenan la cadena a la que hace referencia msg_formato y si está protegida.

Métodos de prevención

El primer paso es nunca pasar entradas de usuario directamente a una función de formateo, además de asegurarse de hacerlo en cada nivel de manipulación de salida formateada. Como nota adicional, las funciones de formato tienen una sobrecarga de trabajo importante. Quizá sea conveniente escribir:

```
fprintf (STDOUT, buf);
```

la anterior línea de código no solamente es peligrosa, sino que también consume mucho ciclos de *CPU (Central Processing Unit)* adicionales.

[67] *Ídem*.

El segundo paso que habrá de darse consiste en asegurar que las cadenas de formato que utiliza la aplicación lean desde lugares confiables y que las rutas a las cadenas no sean controladas por el atacante. [68]

D. Race condition

"La condición de carrera se presenta cuando dos o más procesos leen y escriben en un área compartida y el resultado final depende de los instantes de ejecución de cada uno. Cuando una situación de este tipo se produce, y acciones que deberían ser ejecutadas en un periodo de tiempo fijo (atómicas) no lo hacen, existe un intervalo de tiempo durante el que un atacante puede obtener ciertos privilegios, así como leer y escribir en archivos protegidos, y en definitiva, violar las políticas de seguridad del sistema". [69]

Como ejemplo, veamos el código simplificado de un programa que almacena información en un archivo perteneciente a un usuario con privilegios de *root* para un sistema operativo Unix.

Ejemplo 1.

```
if (access ( archivo, W_OK) == 0) {
        open ();
        write ();
}
```

En una ejecución normal, si el usuario no tiene privilegios suficientes para escribir en el archivo, la llamada a acces () devolverá un -1, no permitiendo ni la apertura ni la escritura de archivo. Pero, ¿qué es lo que sucedería si el archivo cambiara entre la llamada a access()?. Imaginemos que después de la llamada a access(), y justo antes de que se ejecute la llamada a la función open(), el usuario borra el archivo original y lo enlaza por ejemplo a la dirección /etc/passwd: el programa estará escribiendo información en el archivo de contraseñas de los sistemas Linux.

En esta clase de situaciones, donde un programa comprueba la situación de un objeto, y luego ejecuta cierta acción asumiendo que la situación se mantiene cuando en realidad no es así, se denomina TOCTTOU (Time Of Check To Time Of Use). [70]

Qué hacer para evitar esto

Se pueden usar descriptores de archivo en lugar de nombres; para el ejemplo anterior, deberíamos usar una variante a la llamada access() que trabaje con descriptores en lugar de nombres de archivo. Con esto, se consigue que aunque se modifique el nombre del archivo, el objeto al que accedemos sea el mismo

[68] *Ídem*.
[69] Tanenbaum, Andrew. "Distributed Operating Systems" ED. Prentice Hall, 1995.
[70] Antonio Villalón, "Condiciones de carrera", *Seguridad en Unix y redes* en http://es.tldp.org/Manuales-LuCAS/SEGUNIX/unixsec-2.1.pdf, 30/09/2009.

siempre. Además, es conveniente invertir el orden de las llamadas, es decir, invocar primero a open() y después a la variante de access().

E. SQL injection

Se trata de una vulnerabilidad a nivel de la validación a las entradas a la base de datos de una aplicación. Es, en otras palabras, la posibilidad de inyectar sentencias *SQL (Structured Query Languaje)* arbitrarias en, por ejemplo, formularios estándar publicados en un sitio web. [71]

Para iniciar, demos un repaso a los comandos comunes a todas las distribuciones de *SQL*. (Véanse tablas 2, 3 y 4.)

Comandos DCL (Data Control Languaje)

Grant	Utilizado para otorgar permisos
Revoke	Utilizado para revocar permisos
Deny	Utilizado para denegar el acceso

Tabla 2. Comandos DCL, permiten al administrador controlar el acceso a los datos contenidos en una base de datos. [72]

Comandos DDL (Data Definition Languaje Statements)

Create	Utilizado para crear tablas, campos e índices
Drop	Empleado para eliminar tablas e índices
Alter	Utilizado para modificar tablas agregando campos o cambiando su definición.

Tabla 3. Comandos DDL, permiten a los usuarios llevar a cabo las tareas de definición de las estructuras que almacenarán los datos, así como los procedimientos o funciones que permitan consultarlos. [73]

Comandos DML (Data Manipulation Languaje Statements)

Select	Utilizado para conmutar registros de la base de datos para satisfacer un criterio determinado.
Insert	Utilizado para cargar lotes de datos a la base de datos en un única operación
Update	Utilizado para modificar los valores de los campos y registros especificados.
Delete	Utilizado para eliminar registros de una tabla de la base de datos.

Tabla 4. Comandos DML, permiten al usuario llevar tareas de consulta o manipulación de datos. [74]

[71] "Definición de Seguridad informática" en http://www.alegsa.com.ar/Dic/seguridad%20informatica.php, 12/09/2009.
[72] *Ídem.*
[73] *Ídem.*
[74] *Ídem.*

Cláusulas

Las cláusulas son condiciones de modificación utilizadas para definir los datos que se desean seleccionar o manipular. (Véase tabla 5.)

Cláusulas

From	Es utilizada para especificar la tabla de la cual se van a seleccionar los registros
Where	Utilizada para especificar las condiciones que deben reunir los registros que se van a seleccionar
Gruoup by	Utilizada para separar los registros seleccionados en grupos específicos.
Having	Utilizada para expresar la condición que debe satisfacer cada grupo
Order by	Utilizada para ordenar los registros seleccionados de acuerdo a un orden específico

Tabla 5. Relación de cláusulas y su uso. [75]

Operadores de comparación

Son operadores, en su mayoría binarios, que nos permiten comprar variables devolviendo un valor booleano a 1, si se cumple la condición y 0, en caso contrario. (Véase tabla 6.)

Operadores de comparación

<	Menor que
>	Mayor que
<>	Distinto de
<=	Menor o igual que
>=	Mayor o igual que
=	Igual que
Between	Para especificar un intervalo de valores
Like	Utilizado para comparar modelos
In	Utilizado para especificar registros en una base de datos.

Tabla 6. Los operadores de comparación verifican la relación entre dos variables. [76]

Veamos ahora, algunos ejemplos de sentencias *SQL*:

Ejemplo 1.

*SELECT * FROM Tabla;*

Lo que hace esta sentencia es devolver todos los registros que se hallen en la tabla llamada Tabla.

[75] *Ídem.*
[76] *Ídem.*

Ejemplo 2.

UPDATE Tabla SET password = '12345' WHERE user = 'admin'

En esta ocasión, se fijará como contraseña, para el usuario admin el valor 12345.

Ahora bien, es importante destacar la forma en que *SQL* ejecuta las instrucciones, puesto que es un punto fundamental si se quieren construir sentencias *SQL* a inyectar. Independientemente de la complejidad de la sentencia, ésta siempre se ejecutará por partes, es decir, secuencialmente una detrás de otra.

El aumento de páginas que requieran autenticación por parte del usuario, llenado de formularios, encuestas, suscripciones, por mencionar algunas, no hace más que ampliar extraordinariamente el espectro y potencial peligrosidad de esta clase de ataques, debido a que, en prácticamente el 100% de los casos, todos los argumentos recogidos serán pasados como una consulta a una base de datos, la cual será la encargada de mostrar al usuario los resultados de la misma.

A base de ingresar sentencias *SQL*, en el campo de autenticación de alguna página web, es que podemos terminar accediendo a la base de datos de la misma, con diferentes privilegios.

Un ejemplo simple de este tipo de ataque sería el siguiente:

En la parte donde se pide teclear el usuario de alguna página que requiera autenticación, podríamos introducir:

Usuario: '; drop table usuario- -

Con un poco de suerte, es decir, con los privilegios necesarios, la tabla "usuario" sería borrada, lo que traería como consecuencia que a partir de entonces, ningún usuario pueda autenticarse. Un ejemplo simple de ataque *DoS* usando *SQL* injection.

La comilla simple es interpretada por el manejador de bases de datos como un terminados de caracteres, lo que pasará es en el motor *SQL* interpretará que la cadena pasada como argumento empieza después de la primer comilla simple y termina con la segunda comilla.

Para ser más claros, veamos otro ejemplo.

Usuario: Gus'tavo
Password: 1234

Lo que sucederá es que el manejador interpretará los datos que se le han introducido como:

username: 'Gus'
password: 1234

e intentará ejecutar el resto de la consulta, es decir, el 'tavo', lo que dará por resultado, un error, puesto que no significa nada para *SQL*.

En nuestro ejemplo anterior, después de la comilla simple vino la sentencia drop table usuario, seguido de un doble guión: - -, esta sentencia le indica a *SQL* que lo que viene después, es un comentario, así, que lo ignorará y posiblemente evitaremos un error de sintaxis.

Así, con estos simples ejemplos es que podríamos ganar acceso a una base de datos mal protegida. Por supuesto, no son estas las únicas posibles sentencias inyectables, existen un sinfín más, incluso, se podría combinar con otras técnicas de ataque, como por ejemplo, el uso de *sniffers*.

Aunque en la mayoría de los lenguajes de programación, se pueden implementar medidas de seguridad contra este tipo de ataques, la realidad es que muchos de los desarrolladores no se preocupan por implementar seguridad en sus sistemas, resultando en pérdidas graves de información y activos.

2.4 Ataques a contraseñas

Hoy en día, el ataque a contraseñas es el ataque más recurrente puesto que no se requiere de grandes conocimientos técnicos para llevarlo a cabo. Además, la mayoría de los usuarios no suelen recurrir a contraseñas robustas, esto es, que incluyan letras minúsculas y mayúsculas, números y símbolos, sino que es muy común encontrarse como contraseña un nombre, una fecha significativa, una palabra sacada del diccionario, etc. y también, es común encontrar la misma contraseña, para varios pedidos de acceso, todo lo cual, simplifica en gran medida el trabajo al atacante.

Existen diversas prácticas recomendadas para elegir una buena y robusta contraseña, como las siguientes:

- Contraseñas largas.
- Mezclar mayúsculas y minúsculas.
- Incluir caracteres especiales, como por ejemplo, $, -, #, %,.
- Mezclar letras y números.
- Utilizar reglas nemotécnicas:

  i. Pensar en una frase: Este es mi proyecto de tesis.

  ii. Tomar las iniciales: Esmpdt.

  iii. Añadir complejidad: Esmpdt -> 3$mpd7

iv. Mezclar mayúsculas y minúsculas: 3$mPD7 [77]

2.4.1 Ataques por fuerza bruta y diccionario

Se denomina ataque de fuerza bruta a la forma de recuperar una clave probando todas las combinaciones posibles hasta encontrar la correcta. Existe una variedad de herramientas para todos los sistemas operativos que permiten este tipo de técnica. [78]

Los ataques por fuerza bruta, dado que utilizan el método de prueba y error, son muy costosos en lo que se refiere a tiempo computacional. Este tiempo es directamente proporcional a la complejidad con que la contraseña haya sido construida.

Además de dicho ataques, existen los llamados ataques de diccionario, que se centra en la práctica no recomendada que siguen algunos usuarios de elegir contraseñas que tengan algún significado, así, los ataques por fuerza bruta prueban usando cualquier combinación posible, mientras que los ataques de diccionario prueban con palabras conocidas.

Adicionalmente a estos dos ataques, existe un tercero que utiliza las características de los dos anteriores y es conocido como ataque híbrido, este tipo de ataques apuntan a contraseñas compuestas por una palabra tradicional seguida de una letra o número, por ejemplo, si alguien usa la contraseña 'perro123'. [79]

2.4.2 Herramientas

Existe una multitud de herramientas desarrolladas para llevar a cabo este tipo de técnicas, pero aquí veremos algunas de las más comúnmente utilizadas.

- Herramienta: Essential Net Tools.

  Contramedidas: Complejidad de contraseñas, *firewalls*.

  Descripción: Aunque el conjunto de herramientas de Essential Net Tools está completamente dirigido a realizar diagnósticos de redes y monitorización de conexiones, también pueden ser usadas para llevar a cabo test de penetración mediante ataques de contraseñas. Esta aplicación incluye varias herramientas indispensables para la administración y monitorización de redes. Entre ellas:

[77] "Ataques a contraseñas" en http://serdis.dis.ulpgc.es/~a013775/asignaturas/iiaso/curso0708/trabajos/seguridad/crack/ataquecontrasena.pdf , 03/10/2009.
[78] "Contraseñas" en http://es.kioskea.net/contents/attaques/passwd.php3, 03/10/2009.
[79] Saikat Chakrabarti, Mukesh Singhal, "Password-Based Authentication: Preventing Dictionary Atacks", University of Kentucky en http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4249815, 30/04/2011.

- NetStat. Muestra la lista de conexiones de red de una computadora, información sobre puertos *TCP* y *UDP* abiertos, direcciones IP y estados de las conexiones. (Véase figura 20.)

- NBScan. Es un escáner de *NetBIOS*. Esta herramienta ofrece una *interfaz* de usuario gráfica y de fácil administración del archivo lmhosts y funciones de escaneo paralelo, lo que permite comprobar una red clase C en un minuto.

- PortScan. Es un escáner de *TCP* que permite buscar la red en busca de puertos activos.

- Shares. Monitoriza y crea *logs* de conexiones externas de recursos compartidos, listas de compartidos locales, también permite conectar a recursos remotos.

- LMHosts. Es un editor de los archivos lmhosts integrados con NBScan.

- SysFiles. Un editor de archivos de sistema: servicios, protocolo, redes, hosts y lmhosts.

- NetAudit (*NetBIOS* Auditing Tool). Permite mejorar la seguridad de comprobación de la red.

- RawSocket. Ofrece la capacidad de establecer conexiones de bajo nivel *TCP* y *UDP* para resolver y comprobar los diferentes servicios.

- TraceRoute y Ping. Estas utilidades se presentan de forma personalizable y con resultados convenientemente mostrados.

- NSLookup. Permite convertir direcciones IP a *hostnames* y viceversa, obtener los alias, y mejorar las consultas *DNS* avanzadas.

- ProcMon. Muestra una lista de los procesos ejecutándose. Otras funciones incluyen generación de reportes en *HTML*, texto y formatos delimitados por comas, posibilidad de compartir direcciones de *IP* rápidamente entre diferentes herramientas, y mucho más. [80]

[80] "Laboratorios: Hacking – Técnicas y contramedidas – Ataques por fuerza bruta (Brute Force) II" en http://labs.dragonjar.org/laboratorios-hacking-tecnicas-y-contramedidas-ataques-por-fuerza-bruta-brute-force-ii, 04/10/2009.

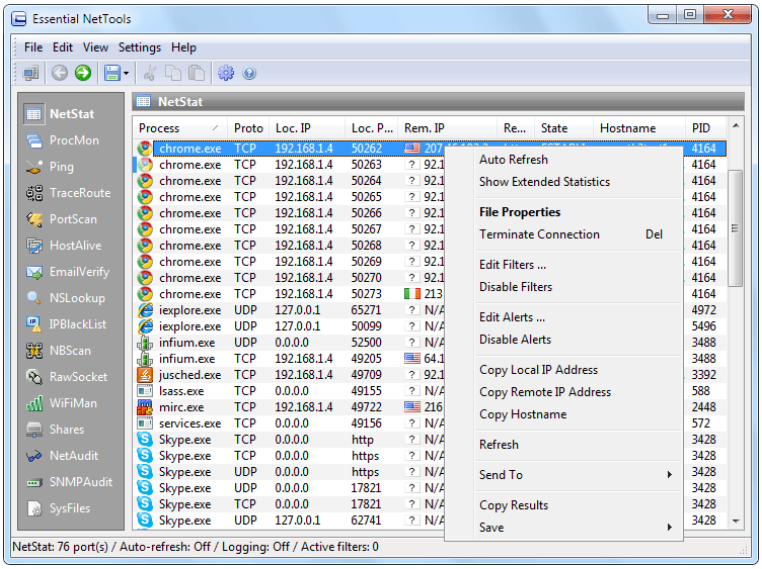


Figura 20. Imagen de la herramienta Essential NetTools, utilizando la herramienta NetStat. [81]

- Herramienta: John the Ripper.

  Contramedidas: Complejidad de contraseñas, políticas robustas de formación de contraseñas.

  Descripción: John the Ripper es un programa de criptografía que aplica fuerza bruta para descifrar contraseñas. Es una herramienta de seguridad muy popular, ya que permite a los administradores de sistemas comprobar que las contraseñas de los usuarios son suficientemente buenas. (Véase figura 21.)

  John the Ripper es capaz de auto detectar el tipo de cifrado de entre muchos disponibles, y se puede personalizar su algoritmo de prueba de contraseñas. Eso ha hecho que sea uno de los más usados en este campo.

  Características:

  - Optimizado para muchos modelos de procesador.
  - Funciona en muchas arquitecturas y sistemas operativos.
  - Ataques de diccionario y por fuerza bruta.
  - Muy personalizable (es software libre).
  - Permite definir el rango de letras que se usará para construir las palabras, y las longitudes.
  - Permite parar el proceso, y continuarlo más adelante.

[81] Tomada de http://www.tamos.com/htmlhelp/nettools/overview.htm

- Permite incluir reglas en el diccionario para decir cómo han de hacerse las variaciones tipográficas.

- Se puede automatizar.

John the Ripper usa un ataque por diccionario: tiene un diccionario con palabras que pueden ser contraseñas típicas, y las va probando todas. Para cada palabra, la cifra y la compara con el *hash* a descifrar. Si coinciden, entonces es la palabra correcta. [82]

Esto funciona bien debido a la poca cultura de seguridad de la mayoría de los usuarios. Además, este software también prueba con variaciones de estas palabras: les añade números, signos, mayúsculas y minúsculas, cambia letras, combina palabras, entre otras (ataque híbrido).

Además ofrece el típico sistema de fuerza bruta en el que se prueban todas las combinaciones posibles, sean palabras o no.

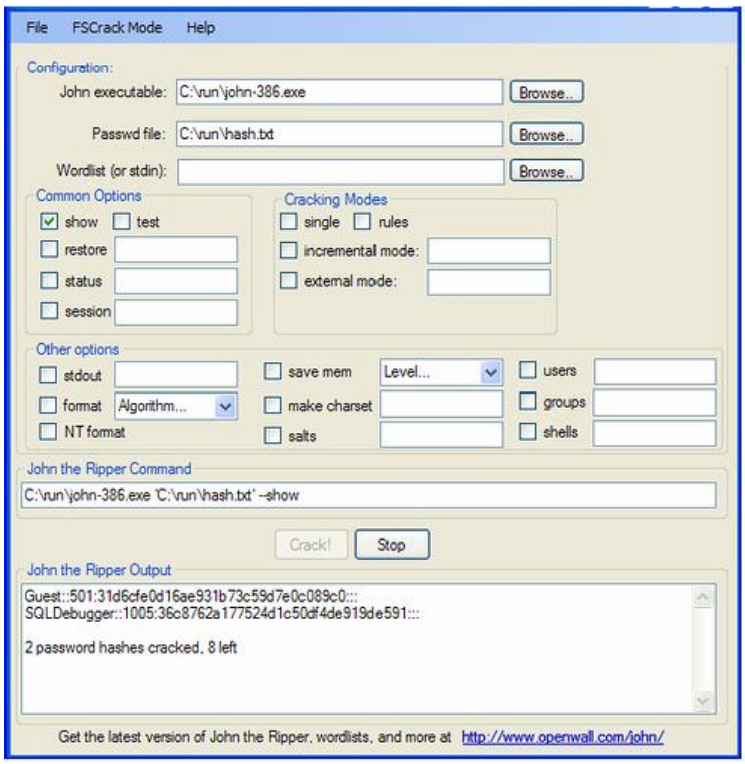


Figura 21. Pantalla principal de John the Ripper. [83]

- Herramienta: BrutusAET 2.

  Contramedidas: Complejidad de contraseñas.

[82] "Laboratorios: Hacking – Técnicas y contramedidas – Ataques por fuerza bruta (Brute Force) II" en http://labs.dragonjar.org/laboratorios-hacking-tecnicas-y-contramedidas-ataques-por-fuerza-bruta-brute-force-ii, 04/10/2009.

[83] Tomada de http://farm3.static.flickr.com/2311/2105464366_99b1b3d9a3.jpg

Descripción: Cuenta con una *interfaz* gráfica amigable, los protocolos soportados son, entre otros: *HTTP* (Autenticación básica), *HTTP* (HTML Form/CGI), *FTP*, *Telnet*, etc.

Características:

- Motor gradual de autenticación.
- Hasta 60 conexiones simultáneas por objetivo.
- Modos configurables de ataque por fuerza bruta.
- Secuencias altamente configurables de autenticación.
- Posibilidad de salvar la sesión de ataque para retomarla después.

BrutusAET2, al igual que John The Ripper, es capaz de utilizar ataques por diccionario, híbridos y de fuerza bruta. (Véase figura 22) [84]

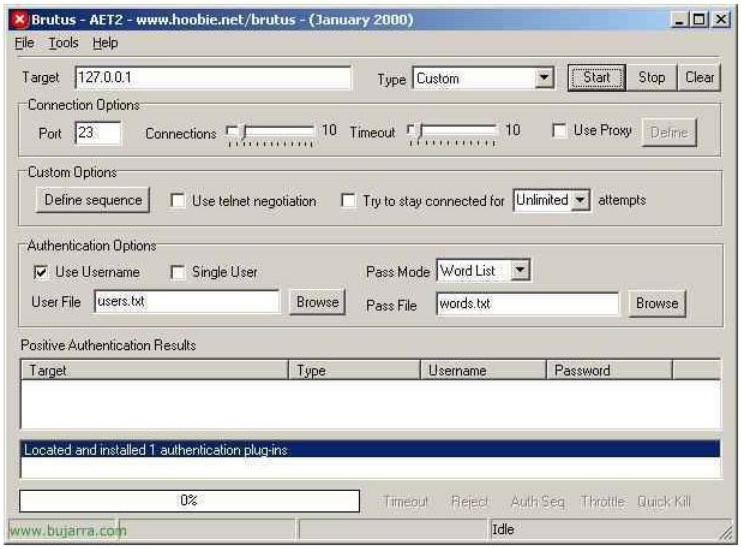


Figura 22. Pantalla principal de BrutusAET2 [85]

- Herramienta: THC-Hydra.

  Contramedidas: *Firewalls*, monitoreo de sesiones, complejidad de las contraseñas.

  Descripción: Dicha herramienta realiza diferentes ataques de fuerza bruta o diccionario a varios protocolos como son: *Telnet*, *HTTP*, *FTP*, entre muchos otros. [86] (Véase figura 23.)

[84] "Brutus" en http://www.bujarra.com/Brutus.html, 05/10/2009.
[85] Tomada de http://www.bujarra.com/Brutus.html
[86] Van Hauser, "THC Hydra" en http://freeworld.thc.org/thc-hydra/, 31/10/2009.

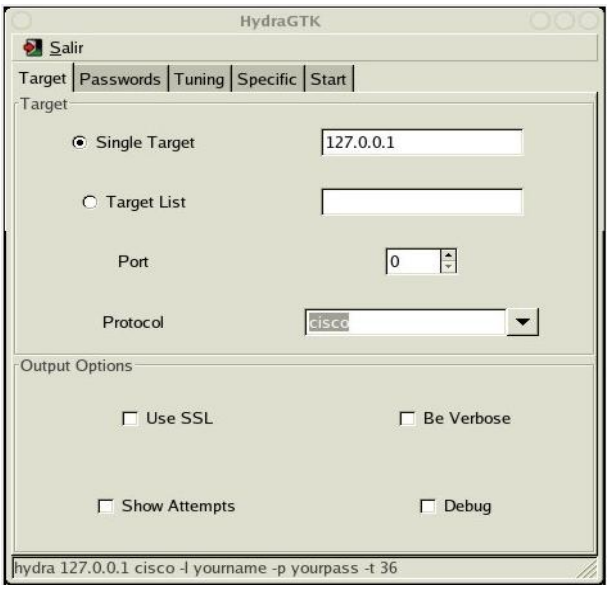


Figura 23. Pantalla de THC-Hydra [87]

2.5 Ataques a redes inalámbricas

Aunque las redes inalámbricas proveen gran facilidad de comunicación y ofrecen una gran movilidad entre los usuarios, representan también un riesgo de seguridad. Los atacantes pueden detectar y obtener acceso a redes corporativas aun cuando existen métodos para implementar seguridad, debido a que la mayoría de éstos son muy débiles y por tanto, fáciles de romper.

2.5.1 Denegación de servicio

Es un ataque en el que uno o más equipos intentan prevenir que una máquina objetivo pueda realizar un trabajo específico. La víctima puede ser un *servidor*, un *router*, un *hipervínculo*, una red completa, un usuario de internet, un proveedor de servicio de internet, un país o una combinación de varios de los casos anteriores. [88]

En el caso de las redes inalámbricas, este ataque puede implicar, entre otras cosas, la imposibilidad de un usuario para conectarse a una red o la pérdida de la conexión a ésta.

Un ejemplo de lo anterior es el llamado ataque de des-autenticación, en el cual el atacante envía una trama de des-autenticación al punto de acceso, provocando una des-autenticación del usuario en el mismo y, consecuentemente, la pérdida de la conexión. (Véase figura 24.)

[87] Tomada de http://www.mexicoextremo.com.mx/content/view/482/62
[88] Andrew Whitaker, Daniel Newman, *Penetration testing and network defense*, Estados Unidos, Cisco Press, 2006, p. 358.

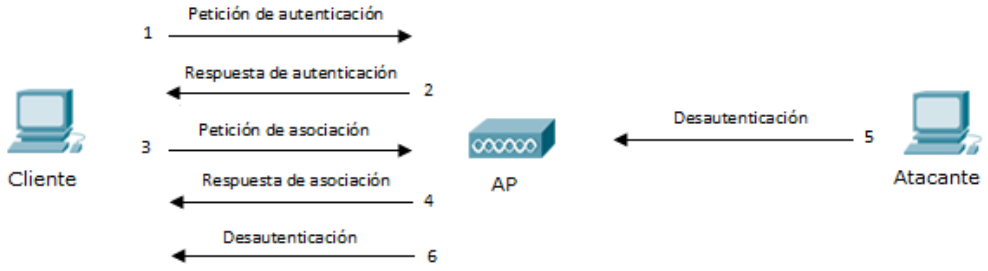


Figura 24. Ataque de des-autenticación.

Otro ataque similar es el *flooding* de autenticación, en donde se inunda el punto de acceso con peticiones que previenen que las genuinas sean atendidas.

Ataques de este tipo pueden ser detectados fácilmente utilizando herramientas inalámbricas *IDS* como Airdefense de Motorola o Airespace.

2.5.2 ARP poisoning

El protocolo *ARP (Address Resolution Protocol)* tiene como finalidad encontrar la dirección MAC que corresponde a una determinada dirección IP. Para ello se envía un paquete *ARP* request a la dirección *broadcast* de la red que contiene la dirección IP que se busca y, posteriormente, se espera a que dicha dirección IP responda con un *ARP* reply que incluye la dirección física que corresponde.[89] (Véase figura 25.)

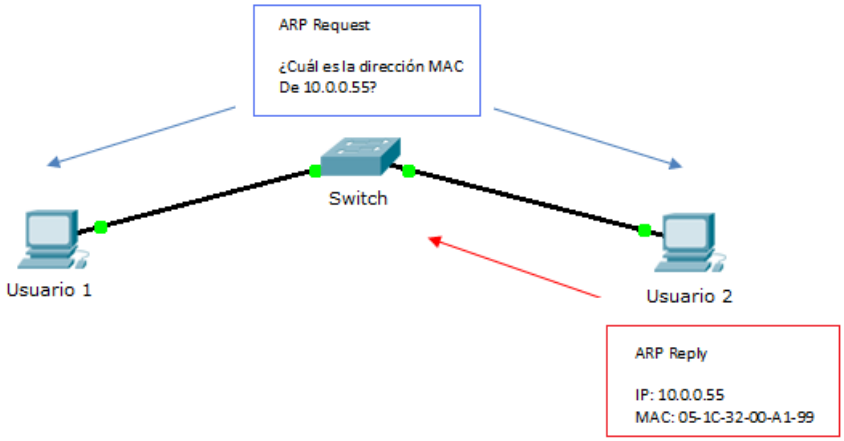


Figura 25. ARP reply.

El principio del *ARP* poisoning es enviar mensajes *ARP* falsos a la red. Normalmente la finalidad es asociar la dirección MAC del atacante con la dirección IP de otro

[89] *Idem*, p. 335, 336.

equipo (la víctima), como por ejemplo la puerta de enlace predeterminada. (Véase figura 26.) Cualquier tráfico dirigido a la dirección IP de esa máquina será erróneamente enviado al atacante, en lugar de a su destino real. El atacante puede entonces elegir entre reenviar el tráfico a la puerta de enlace predeterminada real (ataque pasivo) o modificar los datos antes de reenviarlos (ataque activo). El atacante puede incluso provocar una denegación de servicio contra una víctima, asociando una dirección MAC inexistente con la dirección IP de la puerta de enlace predeterminada de la víctima. [90]

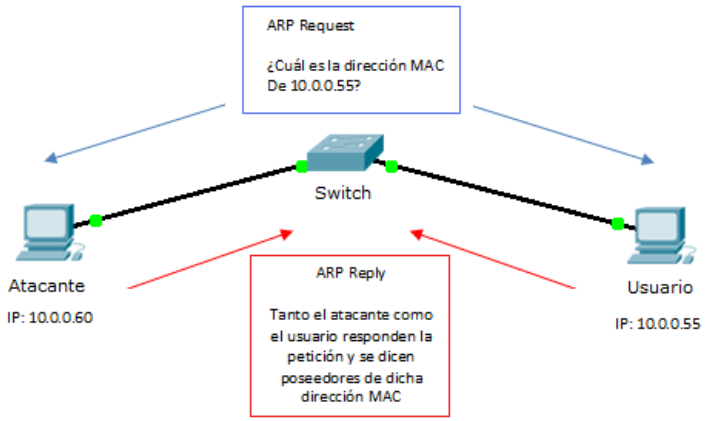


Figura 26. ARP poisoning.

2.5.3 Man in the middle

Es un ataque en el que se establecen conexiones independientes entre dos víctimas y se envían mensajes a ellas y/o entre ellas, haciéndoles creer que están efectuando una comunicación directa a través de una conexión privada cuando de hecho toda la conversación es controlada y manipulada por el atacante. Éste debe tener la posibilidad de interceptar todos los mensajes de la comunicación y hasta de inyectar nuevos en la misma. [91]

Este tipo de ataque es muy atractivo porque el *host* ya se encuentra autenticado con la víctima, siendo así, el atacante no necesita efectuar ningún tipo de *cracking*. No importa qué tan seguro sea un determinado proceso de autenticación porque la mayoría de los sistemas envían texto en claro una vez que tal proceso ha sido realizado con éxito. Esto hace que la mayoría de las computadoras sean vulnerables a este ataque.

Existen dos tipos:

- Activo. Es cuando el atacante se apodera de una sesión activa y compromete un objetivo específico.

[90] Gopi Nath, Shefalika Ghosh, "Different Flavours of Man-In-The-Middle Attack, Consequences and Feasible Solutions" en http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5563900, 30/04/2011.
[91] *Idem*.

- Pasivo. Es cuando el atacante se apodera de una sesión activa e intercepta todo el tráfico entre el *host* y el objetivo sin comprometer a ninguno de los dos. El tipo activo siempre inicia con uno pasivo.

Session replay y *Session hijacking* son considerados también ataques *Man in the middle*. [92]

A. Session replay

El atacante captura paquetes y modifica los datos antes de enviarlos al objetivo. (Véase figura 27.) [93]

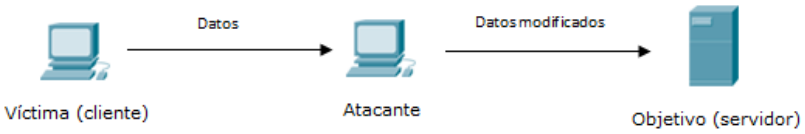


Figura 27. Session replay.

B. Session hijacking

El atacante se apodera de una sesión IP suplantando la fuente o el destino. Frecuentemente se realiza una denegación de servicio hacia el *host* mientras se suplanta a éste dentro de la red. (Véase figura 28.) [94]

Figura 28. Session hijacking.

2.5.4 Cracking WEP

WEP (Wired Equivalent Privacy) forma parte de los estándares 802.11 para conexiones inalámbricas. Utiliza una llave secreta que es compartida entre un cliente y un punto de acceso y es usada en conjunto con el algoritmo RC4 para cifrar todas las comunicaciones entre los mismos.

Puede operar con 40 o 104 *bits* de cifrado. Mientras más fuerte sea tal cifrado, más segura será la red.

---

[92] Andrew Whitaker, Daniel Newman, *Penetration testing and network defense*, Estados Unidos, Cisco Press, 2006, pp. 127-130.
[93] *Ídem*.
[94] *Ídem*.

El problema con *WEP* es el valor tan pequeño de su vector de inicialización que lo hace fácil de averiguar. El vector crea los primeros 24 *bits* de la clave *WEP*. Muchas implementaciones inician utilizando valores del vector iguales a cero y lo incrementan en uno por cada paquete que envían. 24 *bits* es igual a 16, 777, 216 valores, así que después de que se han enviado 16 millones de paquetes, el vector regresa a un valor de cero. Este comportamiento tan predecible de los primeros 24 *bits* de la clave *WEP* hace que ésta sea muy fácil de romper.

Además de lo anterior, en muchos casos no se modifica la clave *WEP* regularmente, provocando que los atacantes mantengan el acceso fácilmente.

Algunos ejemplos de herramientas que son comúnmente utilizadas para romper las claves *WEP* son:

- AirSnort. Es una utilidad de Linux que puede romper claves *WEP*. Requiere que el adaptador de red inalámbrico esté en modo de monitoreo. Captura paquetes de manera pasiva y luego intenta romper la llave cifrada. Con 5 o 10 millones de paquetes capturados, AirSnort puede romper la clave en menos de un segundo.

- WEPCrack. Es similar a AirSnort, pero no es tan popular. Está basado en el lenguaje Pearl.

- WifiSlax. Es una herramienta con utilidad de LiveCD basada en la distribución Slax de Linux, muy similar a las anteriores. [95]

2.6 Eliminación de evidencias

Casi al mismo tiempo que se desarrollara la técnica informática forense, se comenzaron a originar contramedidas para prevenir que los piratas informáticos fueran descubiertos, así, la eliminación de evidencias se hizo práctica común entre aquellas personas expertas en informática que no deseaban que sus acciones ilegales pudieran ser descubiertas.

A medida que se explora e investiga más sobre las técnicas anti-forenses, se han generado varias clasificaciones, como son:

- Destrucción de la evidencia.

- Ocultamiento de la evidencia.

- Eliminación de las fuentes de evidencia.

- Falsificación de la evidencia. [96]

[95] *Ídem*, pp. 352-357.
[96] Armando Botero, Iván Camero, Jeimy Cano, "Técnicas anti-forense en informática: ingeniería reversa aplicada a TimeStomp", Colombia, Pontificia Universidad Javeriana en http://www.fing.edu.uy/inco/eventos/cibsi09/docs/Papers/CIBSI-Dia3-Sesion6%283%29.pdf, 27/01/2010.

2.6.1. Destrucción de la evidencia

El principal objetivo de esta técnica es evitar que la evidencia sea encontrada por los investigadores y en caso de que esto no sea posible, disminuir radicalmente el uso que se le podría dar. Este método no busca que la evidencia sea inaccesible, sino irrecuperable.

Esto implica que se deben destruir, desmantelar o modificar todas las pruebas útiles para una investigación.

Existen dos niveles de destrucción de la evidencia:

- Nivel Físico: A través de campos magnéticos (discos duros, externos, por mencionar algunos).

- Nivel Lógico: Busca cambiar la posición de los datos, sobreescribirlos, eliminar la referencia a los mismos, entre otros.

Existe una variedad de herramientas para la destrucción de evidencia de las cuáles se pueden valer los intrusos para realizar este método. Un ejemplo de estas son: Wipe, Shred, PGP Secure Delete, Evidence Eliminator y Swap.

2.6.2 Ocultamiento de la información

Este método tiene como principal objetivo el hacer inaccesible la evidencia para el investigador. No busca manipular, destruir o modificar la evidencia, sino hacerla lo menos visible posible.

Esta técnica puede llegar a ser muy eficiente de ser bien ejecutada, pero conlleva muchos riesgos para el atacante, puesto que al no modificar la evidencia, de ser encontrada puede ser válida en una investigación formal y por lo tanto, servir para la incriminación e identificación del autor de dicho ataque.

Una de las herramientas utilizadas por los atacantes es la esteganografía, la cual trata de técnicas que permiten la ocultación de mensajes u objetos dentro de otros, llamados portadores, de modo que no se perciba su existencia. (Véanse figuras 29 y 30.) En el mercado se pueden encontrar muchas herramientas fáciles de usar tales como StegoArchive.

Figuras 29 y 30. Ejemplo de esteganografía. En este cuadro titulado "Los embajadores" de Holbein, llama la atención una mancha bastante rara entre los protagonistas en la imagen de la izquierda. Mirando el cuadro desde el ángulo correcto, aparece una calavera (imagen derecha). Este es un ejemplo de esteganogtrafía visual. [97]

2.6.3 Eliminación de las fuentes de evidencia

Este método tiene como principal objetivo neutralizar la fuente de la evidencia, por lo que no es necesario destruir las pruebas puesto que no han llegado a ser creadas. Por ejemplo, en el mundo real cuando un criminal utiliza guantes de látex para manipular el arma, lo que está haciendo es evitar dejar huellas dactilares.

Una de las acciones que los atacantes pueden llevar a cabo es la desactivación de los *logs* del sistema que se está atacando y la edición de la bitácora del mismo. [98]

2.6.4 Falsificación de la evidencia

Este método busca engañar y crear falsas pruebas para los investigadores forenses logrando así cubrir a el verdadero autor, incriminando a terceros y por consiguiente, desviando la investigación con lo cual sería imposible resolverla de manera correcta.

El ejercicio de este método se vale de una edición selectiva de las pruebas creando evidencias incorrectas y falsas que corrompen la validez de las pruebas de investigación forense, por lo cual no podrán ser tomadas en cuenta como evidencia. [99]

[97] Tomadas de http://danteslab.blogspot.com/
[98] "Armando Botero, Iván Camero, Jeimy Cano, "Técnicas anti-forense en informática: ingeniería reversa aplicada a TimeStomp", Colombia, Pontificia Universidad Javeriana en http://www.fing.edu.uy/inco/eventos/cibsi09/docs/Papers/CIBSI-Dia3-Sesion6%283%29.pdf, 27/01/2010.
[99] *Ídem*.

2.6.5 Herramientas utilizadas en las técnicas anti-forense

- TimeStomp. Es una herramienta anti-forense que puede leer y escribir los registros de tiempo o *time-stamps* de los archivos *NTFS (New Technology File System)*. Los registros de tiempo sirven para almacenar información de cuándo fue modificado, accedido, creado o modificado un archivo en el sistema *NTFS*. Según la clasificación anteriormente ofrecida, TimeStomp es una herramienta que destruye y falsifica información.

  Con este programa se consigue evitar que una analista forense obtenga una línea de tiempo de sucesos en un sistema. (Véase figura 31.)

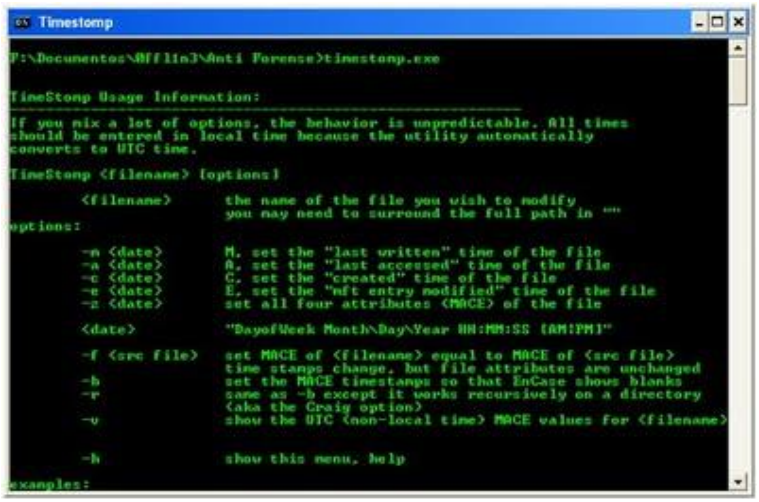


Figura 31. Pantalla principal de TimeStomp. [100]

- Shred. Es una utilidad disponible para sistemas Linux que sobreescribe los archivos especificados un número determinado de veces, 25 por defecto, con varios patrones de texto, convirtiendo así, el contenido del archivo original en otro totalmente distinto y llenándolo de información sin sentido, haciendo que la recuperación del mismo sea prácticamente imposible. (Véase figura 32.)

  Esta es una utilidad que destruye la información.

[100] Tomada de http://www.0ffl1n3.com/2009/12/tecnicas-anti-forenses.html

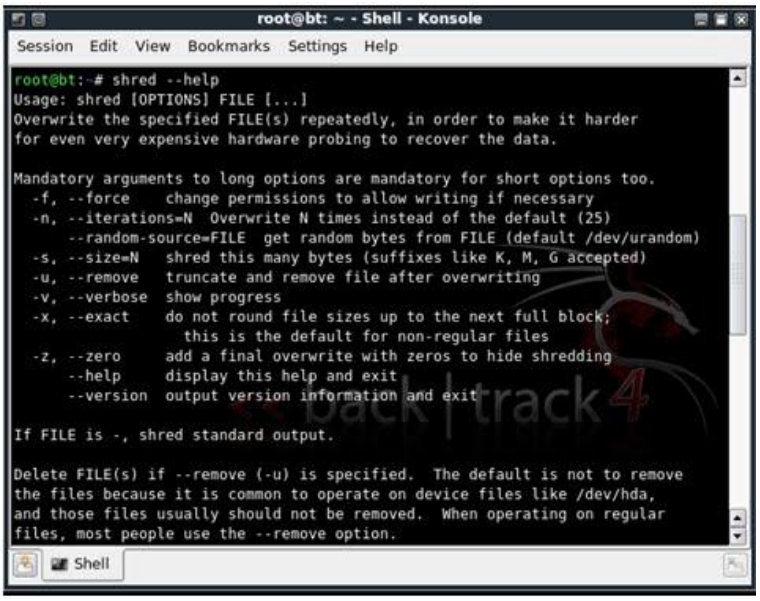


Figura 32. Vista de las opciones de la utilidad shred. [101]

- Evidence eliminator. Programa orientado al sistema operativo Microsoft Windows que permite eliminar información de forma casi permanente del disco duro. Sobre escribe la información a ser eliminada con el fin de que su recuperación sea prácticamente inviable. Esta aplicación destruye la información. [102] (Véase figura 33.)

Figura 33. Pantalla de funcionamiento de Evidence Eliminator. [103]

101 Tomada de www.forensicswiki.org/wiki/Timestomp
102 Armando Botero, Iván Camero, Jeimy Cano, "Técnicas anti-forense en informática: ingeniería reversa aplicada a TimeStomp", Colombia, Pontificia Universidad Javeriana en http://www.fing.edu.uy/inco/eventos/cibsi09/docs/Papers/CIBSI-Dia3-Sesion6%283%29.pdf, 27/01/2010.
103 Tomada de http://www.pctrackscleaner.com/images/evidence-eliminator-screen-12.jpg

Tema 3. Análisis de riesgo

3.1 Introducción

Como se ha mencionado, en un entorno informático existen una serie de recursos que están constantemente expuestos a diferentes tipos de riesgos: aquellos comunes a cualquier entorno, y los excepcionales, originados por situaciones concretas que pueden afectar a parte de una organización o a toda ella.

Para tratar de minimizar los efectos de un problema de seguridad se realiza un análisis de riesgos, que es un proceso necesario para responder a tres preguntas básicas sobre nuestra seguridad:

- ¿Qué queremos proteger?
- ¿Contra qué lo queremos proteger?
- ¿Cómo lo queremos proteger?

En la práctica existen dos aproximaciones para responder a estas cuestiones, una cuantitativa y otra cualitativa. La primera es la menos usada, ya que en muchos casos implica cálculos complejos o datos difíciles de estimar. Se basa en dos parámetros: la probabilidad de que un suceso ocurra y una estimación de las pérdidas en caso de que así sea. Aunque es posible conocer el riesgo de cualquier evento y tomar decisiones en función de estos datos, en la práctica la inexactitud en la estimación o en el cálculo de parámetros hace difícil esta aproximación.

El segundo método es el cualitativo, de uso muy común en la actualidad. Es mucho más sencillo que el anterior, ya que ahora no entran en juego probabilidades exactas sino sólo una estimación de pérdidas potenciales. Para ello se interrelacionan cuatro elementos: las amenazas, las vulnerabilidades, el impacto asociado a una amenaza y las contramedidas para minimizar las vulnerabilidades o el impacto. Con estos cuatro elementos podemos obtener un indicador cualitativo del nivel de riesgo en el que un activo determinado se encuentra, considerándolo como la probabilidad de que una amenaza se materialice sobre un activo y produzca un determinado impacto.

En este capítulo explicamos las fases necesarias para realizar con éxito un análisis de riesgos, desde su preparación y puesta en práctica hasta el correcto análisis de los resultados obtenidos.

3.2 Preparación del proyecto

En esta fase se recoge toda la información necesaria para la identificación de activos informáticos, así como sus vulnerabilidades y amenazas potenciales a los cuáles están sujetos. Se obtiene la probabilidad de ocurrencia de la explotación de una vulnerabilidad y el impacto que tendrá en el funcionamiento de la empresa o negocio si esto llegara a suceder.

Esta fase es fundamental para la construcción de una estrategia de seguridad perfectamente coherente con la operación y las necesidades de la organización.

Según la IEC 27001:2005 la evaluación del riesgo incluye las siguientes acciones:

- Identificación y valoración de los activos
- Identificación de las amenazas y vulnerabilidades importantes para los activos identificados.
- Evaluación del riesgo, de las amenazas y vulnerabilidades con cierta incidencia a ocurrir.
- Cálculo del riesgo.
- Evaluación de los riesgos frente a una escala de riesgo preestablecida. [104]

3.3 Identificación de activos

Un activo es un bien importante para el funcionamiento o manejo de la empresa y que la seguridad informática tiene como objetivo proteger. Los activos están conformados por los siguientes elementos:

- Información. Ya sea que esté guardada en un medio físico o electrónico, la información es el bien de mayor valor para una organización.
- Hardware. El equipo físico que conforma la estructura del sistema de comunicación.
- Software. Las aplicaciones y programas que son usadas en la empresa u organización.
- Usuarios. Los individuos que están en contacto con la información y que hacen uso del hardware y software. [105]

Para realizar la identificación de los activos es necesario contar con la presencia de personal calificado que tenga conocimiento del proceso informático de la organización.

3.4 Evaluación de activos

Una vez que los activos han sido identificados, se procede a evaluar su valor desde el punto de vista de la seguridad de la información y no sólo con base en

[104] "Análisis de riesgos. Seguridad Informática" en http://74.125.93.132/search?q=cache:izpLYC9qhlUJ:www.felaban.com/memorias_mayo_09/viernes_15_mayo/santiago_lioza_clain_v4.ppt+p%C3%A9rdida+esperada+seguridad+inform%C3%A1tica&cd=7&hl=es&ct=clnk&gl=mx , 22/10/2009.
[105] *Ídem*.

su valor intrínseco. El valor del activo se debe calcular según su misión crítica, costo, sensibilidad o una combinación de ambos valores. [106]

El valor de los activos es un factor importante en la decisión para modificar la operación o incrementar la protección a dichos elementos.

3.5 Impacto

El impacto puede definirse como las pérdidas que resultan de la explotación de una vulnerabilidad. Las pérdidas, por lo general, son expresadas en una o más áreas de impacto como destrucción, denegación de servicio, revelación o modificación. Aquí también toma parte el concepto de pérdida esperada, la cual es el impacto anticipado a los activos, resultado de la manifestación de una amenaza. [107]

Para poder considerar esto, es necesario tomar en cuenta los siguientes aspectos:

- Consecuencias de tipo financiero, es decir, pérdidas causadas sobre un activo físico o lógico determinado y las consecuencias de que éste no funcione y afecte la operación de la compañía.

- La importancia crítica a la organización de los datos y el sistema.

- Sensibilidad de los datos y sistema. [108]

3.6 Pasos del análisis de riesgo

El análisis y manejo del riesgo se resume mediante la interacción de los siguientes pasos:

- Determinación del alcance. Etapa en la que se define el motivo por el cual se está realizando el análisis, además, se definen los procesos que serán objeto de evaluación.

- Definición del equipo de trabajo. Como su nombre lo indica, se establecerá el personal involucrado en la elaboración del análisis.

- Fase de entrevistas. Etapa primordial del análisis de riesgo, puesto que permitirá conocer el bien visto desde el punto de vista del usuario de la información y de los dueños.

- Identificación de activos. En esta etapa se enumeran los activos con los que cuenta la empresa, se les asigna un valor considerando lo aprendido en la fase de entrevista con los dueños y usuarios. Este valor asignado es significativo puesto que será un indicador de la protección que necesitará

[106] Ídem.
[107] Ídem.
[108] Ídem.

dicho activo, y del impacto de su pérdida en caso de que un ataque ocurra.

- Identificación de amenazas. Consecuente con cada organización, lugar geográfico en que la empresa se encuentra, y la información anteriormente recabada, se deben definir las amenazas, ya sean externas o internas que pesan sobre los activos, se elabora además una estimación sobre la ocurrencia con que estas pueden presentarse.

- Priorización de amenazas. A partir de la estimación de ocurrencia anteriormente elaborada y siguiendo fórmulas sencillas descritas por organismos internacionales y aceptadas como normas, se deberá prestar especial atención a aquellas cuya ocurrencia sea inminente y sus consecuencias serias.

- Identificación de controles.

  i. Controles requeridos. Como su nombre lo indica, son todas aquellas normas que pueden fundamentarse en las reglas escritas, se espera que su cumplimiento reduzca la posibilidad de un ataque.

  ii. Controles discrecionales. Estos se aplican cuando el nivel de riesgo no se minimiza a un nivel aceptable siguiendo los controles requeridos, son aplicados por los administradores del sistema.

- Riesgo residual. Si tenemos que cuenta que todo sistema está sujeto a sobrellevar algún riesgo, el llamado riesgo residual invariablemente existirá, no importando qué tan estrictos sean nuestros controles o que tan capaces nuestros administradores. En esta fase se concluirá si el riesgo residual es aceptable se precisa de la implementación de controles adicionales.

- Informe del análisis. Cuando se completa el análisis, debe prepararse un reporte escrito que incluya, como mínimo, los siguientes aspectos:

  i. El nivel de vulnerabilidad en que se encuentra un activo.

  ii. Amenazas, estableciendo prioridad de acción y fijando su riesgo de ocurrencia.

  iii. Ambiente bajo el cual se realizó el análisis.

  iv. Estado de la conexión del sistema.

  v. Sensibilidad de los datos; se hará una lista de ellos clasificándolos según su importancia en la organización.

  vi. El invariable riesgo residual al que está expuesta la empresa.

vii. Cálculos de la expectativa de pérdida. [109]

3.7 Análisis costo-beneficio

En esta etapa dentro del análisis de riesgo se deberá elaborar además, un análisis costo-beneficio; esto se trata de una técnica que evalúa, en nuestro caso, el peso total del gasto que se desembolsará para llevar a cabo la salvaguarda de los activos, el valor de la totalidad que a los mismos que se les da, y el costo en tiempo y recursos para que un atacante logre pasar las medidas de defensa.

Dentro del análisis de riesgo deben considerarse tres costos principales:

- Costo del sistema (Ca). Valor de los activos a proteger.
- Costo de los medios (Cm). El costo que un atacante requiere para destruir las medidas de seguridad implementadas.
- Costo de las medidas de seguridad (Cs). El costo para proteger el sistema o activo.

Para que la implementación de las medidas de seguridad sea viable, debe cumplirse la siguiente relación:

$$Ca > Cm > Cs$$

Esto significa que el costo que significa atacar al sistema debe ser mayor que el valor o información al que se tendrá acceso si se tiene éxito. Además, la información no debe ser más costosa que la información protegida, de lo contrario, no convendría protegerla y sería mejor obtener la información de nuevo en caso de pérdida.[110]

[109] María Jaquelina López Barrientos, Cintia Quezada Reyes, *Fundamentos de seguridad informática*, México, Universidad Nacional Autónoma de México, Facultad de Ingeniería, 2006, pp. 163-168.
[110] *Idem*.

Tema 4. Hacking ético

4.1 Introducción

En el mundo digital de la actualidad, las empresas comienzan a tener dificultades para proteger la información confidencial propia o de clientes, al mismo tiempo que buscan mantener una presencia importante en el negocio de las tecnologías de la información.

Una alternativa para alcanzar un nivel aceptable de protección de sus activos más importantes son las pruebas de penetración, un campo de la seguridad informática que ha crecido mucho últimamente y que permite a *hackers* éticos realizar evaluaciones de seguridad con la responsabilidad y honestidad siempre en mente.

Como último preludio a la realización concreta de nuestro proyecto, a continuación explicamos a detalle los conceptos en los que se basa el mismo, las pruebas de penetración, sus diferentes variantes y los pasos a seguir para llevarla a cabo de manera exitosa.

4.2 Definición

El *hacking* ético o pruebas de penetración "*es la práctica que una entidad confiable realiza para intentar comprometer la red de computadoras de una organización, con el propósito de evaluar su seguridad*". [111] Mediante la simulación de un ataque en vivo, los administradores pueden ser testigos del daño potencial que un atacante puede provocar al ganar acceso, destruir datos o dañar los valores de la compañía.

Las pruebas de penetración pueden ser de tres tipos:

- Pruebas de caja negra. No se tiene conocimiento alguno de la red que se probará. Se puede, por ejemplo, contar con una dirección web o IP e intentar ganar acceso a la misma como si se fuera un atacante externo.

- Pruebas de caja blanca. Se tiene un conocimiento total de la red interna. Puede contarse con diagramas de la red, listas de aplicaciones y de sistemas operativos, etc. A pesar de que no es una simulación realista de un ataque externo, es el más exacto en relación con el "peor escenario", aquel en el que el atacante también cuenta con total conocimiento de la red objetivo.

- Pruebas de caja gris. Se simula ser un empleado. Se tiene una cuenta de acceso estándar a la red interna, por tanto, es una prueba acerca de las amenazas que un empleado dentro de la compañía puede generar. [112]

[111] Andrew Whitaker, Daniel Newman, *Penetration testing and network defense*, Estados Unidos, Cisco Press, 2006, p. 5.
[112] *Idem*, p. 6.

4.3 Pasos a seguir

Antes de realizar una prueba de penetración hay algunas cosas que deben definirse, entre ellas, el ámbito en la que ésta se llevará a cabo. Algunos de los factores a considerar para definir correctamente dicho ámbito son los siguientes:

- Definir los horarios en los que la prueba se llevará a cabo (durante o después de las horas de trabajo).
- Definir si las denegaciones de servicio serán permitidas.
- Definir si troyanos y *backdoors* pueden ser instalados en los sistemas.
- Definir si podrá intentarse atacar los sitios web.
- Definir si las bitácoras podrán ser borradas.
- Definir si el tipo de prueba será de caja negra, blanca o gris.
- Determinar si los administradores están conscientes de que las pruebas se llevan a cabo. No es recomendable que estén informados porque es probable que busquen endurecer las medidas de seguridad, lo que provocará que los resultados obtenidos no muestren lo que normalmente pasaría dentro de la red en un ataque real.
- Determinar qué sistemas serán los objetivos de evaluación.
- Definir si se permitirá el empleo de la ingeniería social.
- Definir si se permitirá la obtención y remoción de información de los sistemas evaluados.
- Determinar mediante qué medios será distribuido el reporte de la prueba y a quién. [113]

Una vez que se ha estructurado el ámbito de la prueba se puede comenzar a realizarla. De manera general, ésta puede dividirse en las siguientes cinco etapas:

- Reconocimiento. En esta fase se reúne la mayor cantidad de información posible acerca del objetivo. Este reconocimiento puede ser activo, en el que se utilizan diversas herramientas para la obtención de la información deseada, o pasivo, en el que se utiliza información que está disponible públicamente para descubrir información acerca de las tecnologías que posee la compañía.
- Escaneo. En esta fase se busca determinar qué servicios y qué sistemas operativos corren en el sistema objetivo. Se realiza de manera general un

[113] *Idem*, pp.6-8.

escaneo de puertos que además de ayudar a recabar la información antes mencionada, nos pueden permitir identificar vulnerabilidades para ganar acceso al sistema posteriormente.

- Ganar acceso. Una vez realizado el escaneo de debilidades en la red, se procede a explotar tales fallas.

- Mantener acceso. Cuando se realiza con éxito la penetración en el sistema, generalmente se busca mantener el acceso al mismo para futuros ataques. Esto se logra mediante la instalación de *backdoors*.

- Cubrir rastros. La última fase de la prueba es la eliminación de evidencias. Muchos ataques se realizan sin ser detectados, por lo que es conveniente evaluar cuáles de ellos pueden ser exitosos al cubrir los rastros dejados. [114]

4.4 Informe de observaciones

Una prueba de penetración es inservible si no se tiene algo tangible para dar al cliente. Un informe de observaciones debe incluir los resultados de las pruebas y si es el caso, deben documentarse las recomendaciones para asegurar sistemas de alto riesgo.

El informe debe contener las siguientes secciones:

- Resumen ejecutivo. Es una descripción general corta de la prueba, escrito para ejecutivos clave que quieren saber cómo afectan los resultados a su compañía y que probablemente no le darán mucha importancia a los detalles técnicos. Incluye además, un caso de negocio detallando el impacto de los resultados y los costos asociados a la reparación de las vulnerabilidades descubiertas.

- Ámbito del proyecto. Debe incluir el rango de direcciones IP probadas y factores como si se utilizó la ingeniería social, si se probaron redes públicas o privadas, si se utilizaron troyanos o *backdoors*, por mencionar algunos. Debe incluir además, un estimado del número de *exploits* utilizados y el tipo de cada uno de ellos.

- Análisis de resultados. Esta la parte esencial del informe. La extensión de esta sección puede variar dependiendo del ámbito y los detalles de la prueba. Debe utilizarse una plantilla que incluya lo siguiente:

  i. Dirección IP y dominio del equipo probado.

  ii. Puertos *TCP* y *UDP* abiertos.

  iii. Descripción de los servicios.

[114] *Idem*, pp. 35-37.

iv. Pruebas realizadas.

v. Análisis de vulnerabilidades.

- Resumen. El resumen ejecutivo al inicio del informe está dirigido hacia aquellos que toman las decisiones clave; el resumen final está dirigido hacia el personal técnico. Debe contener una lista de recomendaciones técnicas para el cliente.

- Apéndice. Finalmente, el informe debe incluir un apéndice con las siguientes secciones:

i. Información de contacto.

ii. Impresiones de pantalla.

iii. Registro de salida.

Las impresiones de pantalla y registros de salida son especialmente importantes. Se debe documentar todo lo que se hace durante la prueba para demostrar al cliente el trabajo realizado. [115]

[115] *Ídem*, pp. 40-45.

Tema 5. Implementación de pruebas

5.1 Introducción

En los capítulos anteriores hemos mencionado los fundamentos teóricos en los que basamos la realización de nuestro proyecto.

Explicamos las bases de la seguridad informática para hacer notar la relevancia y la justificación de nuestras pruebas. Posteriormente enumeramos los ataques más comunes a los que se ven expuestos los sistemas de información y las herramientas más populares para llevarlos a cabo, con la finalidad de desglosar los procesos y los efectos de las pruebas que realizaríamos y, finalmente, establecimos la estructura de nuestros planes basándonos en las metodologías de análisis de riesgo y *hacking* ético.

A continuación, presentamos la implementación real de todos los antecedentes recopilados. Detallamos nuestros planes, su cronología, las herramientas utilizadas, su ejecución, su documentación, el análisis de resultados y nuestras recomendaciones derivadas de todos los procesos.

5.2 Plan de pruebas

Plan de pruebas

Con la finalidad de evitar la ejecución de un proyecto de seguridad limitándonos a la utilización aleatoria de diferentes herramientas, presentamos este plan de pruebas metodológico con la finalidad de definir detalles acerca de qué hacer, cuándo hacerlo y cómo hacerlo.

El tipo de prueba que se plantea llevar a cabo es el de caja blanca, esto debido a que contamos con acceso a información vital de la red como diagramas, sistemas operativos y aplicaciones instaladas.

Siguiendo la teoría existente respecto a las pruebas de penetración habituales, los siguientes son los pasos a considerar:

1. Reconocimiento del *host* y escaneo. Tiene como finalidad la recolección de información acerca de la red. Cabe mencionar que la metodología empleada para tales objetivos es la de reconocimiento activo del *host*, que consiste en la utilización de herramientas técnicas para la obtención de información. Éste es un proceso que es fácilmente detectable, sin embargo, es el que arroja mejores y más precisos resultados.

   Ejemplos del tipo de información a recolectar en primera instancia son los siguientes:

   - Direcciones IP de los equipos en la red.
   - Puertos *UDP* y *TCP* accesibles en los sistemas.
   - Sistema operativo en los sistemas.

Herramientas:

NMap. Es una aplicación que corre en varias plataformas, incluyendo Linux y Windows (la versión de Linux arroja resultados más exactos), que permite el escaneo de grandes redes. Ayuda a determinar los *hosts* activos y qué servicios están ofreciendo éstos. Permite, además, implementar muchas de las diferentes técnicas de escaneo de puertos existentes.

Cuando los resultados arrojan puertos clave abiertos NMap es capaz de determinar el sistema operativo de los *hosts*, de lo contrario, puede utilizarse alguna herramienta específica de *fingerprinting*.

Xprobe2. Es una aplicación que corre en Linux y que permite la identificación del sistema operativo instalado en un objetivo. Contiene una base de datos con las diferentes firmas de los sistemas operativos más conocidos aunque también incluye una puntuación probabilística para adivinarlo.
Una vez recolectada la información anterior, puede dibujarse un diagrama de red que contenga los nombres de los *hosts*, las direcciones IP, el número de los puertos activos y los sistemas operativos instalados.

2. Penetración de la red.

Servidores y *Firewalls*

Herramientas:

Nessus. Es un escáner de vulnerabilidades de código abierto que corre en diversos sistemas operativos dentro de los que se encuentran Linux y Windows. Se encarga de verificar los servicios y vulnerabilidades que se suscitan a través de puertos estándar y no estándar. Puede mostrar sus resultados en formato *HTML* para una mejor visualización.

HTTPTunnel. Es una herramienta que utiliza la técnica de *tunneling* a través del puerto *TCP* 80. Se trata de una aplicación cliente/servidor que requiere que el segundo sea instalado en el sistema objetivo. HTTPTunnel permite redirigir puertos no abiertos en un *firewall* hacia el mencionado puerto 80, con lo que puede conseguirse la utilización de servicios que de otra manera no serían permitidos.

Switches y *Routers*

Herramientas:

WiFiSlax. Es una distribución GNU/Linux con funcionalidades *LiveCD* y *LiveUSB* que incluye una larga lista de herramientas de seguridad y auditoría listas para ser utilizadas entre las que destacan numerosos escáneres de puertos y vulnerabilidades, herramientas para creación y

diseño de *exploits*, *sniffers*, herramientas de *análisis forense* y herramientas para la auditoría inalámbrica.

Ettercap. Es una suite de herramientas disponible para sistemas Windows, Linux, entre otros. Incluye métodos de *sniffing*, inyección de caracteres, colección de contraseñas, filtrado y sustitución de paquetes, envenenamientos de la red, robo de puertos, por mencionar algunos. Muchas de las funcionalidades anteriores permiten la ejecución de ataques *man in the middle* en LANs (Local Area Network) con *switch*.

Wireshark. Es un analizador de protocolos muy utilizado para observar el tráfico de una red de datos y solucionar problemas suscitados en las mismas. Cuenta con una *interfaz* gráfica con varias opciones de filtrado y de organización de la información además de la capacidad de reconstruir el flujo completo de una sesión *TCP*.

3. Mantener acceso. Establecer un *backdoor* en el sistema para futuros ataques. A pesar de que es importante conocer acerca de virus y gusanos cuando se planea la manera de asegurar la red de una organización, no son herramientas comunes que se utilicen en las pruebas de penetración. Sin embargo, los caballos de Troya son utilizados normalmente para ganar y mantener acceso a sistemas comprometidos.

   Es ésta la manera en que realizaremos esta fase dentro de nuestra metodología.

   Herramientas:

   Back Orifice 2000 (BO2K). Es una herramienta cliente-servidor de administración remota disponible para plataformas Windows y Linux que permite actividades como la edición de registros, transferencia de archivos, creación de línea de comandos, control de procesos, apagado y reinicio remoto, captura de archivos de contraseñas, captura de pantallas, control de ratones y teclados, comunicación cifrada, entre otras.

   El *servidor* debe ser instalado en el equipo objetivo, por lo cual, es un proceso que se realiza una vez que éste ha sido comprometido. Tiene la ventaja de que puede ser ocultado del administrador de tareas de Windows aun estando en ejecución.

   La ejecución de todas las funcionalidades señaladas anteriormente dependen de la instalación de diversos *plug-ins*, agrupados en categorías como cifrado, autenticación, habilitación de servidores y de clientes, comunicaciones y misceláneo.

4. Destrucción de la evidencia. Borrar los archivos necesarios para ocultar el hecho de que la red ha sido vulnerada, y que en un posible *análisis forense* posterior, ésta no pueda ser usada en contra del atacante.

Para este efecto, la herramienta Back Orifice 2000 descrita arriba, brinda opciones útiles.

5.3 Documentación

5.3.1 Resumen

Este reporte detalla varias pruebas de intrusión recientes hechas en un laboratorio de cómputo de la UNAM, realizadas por los alumnos Luis Hugo Flores Román y Gustavo Gabriel Hernández Hernández, entre las fechas 19 de enero del 2010 y 26 de febrero de 2010. Tales pruebas tienen la finalidad de comprobar la seguridad de la red interna emulando las técnicas de un atacante malicioso. La combinación de pruebas ejecutadas contra la red incluyó el escaneo de puertos, escaneo de vulnerabilidades, ataques contra contraseñas entre otras, detalladas más adelante en el reporte.

Tras analizar los resultados de las pruebas realizadas se recomienda lo siguiente para mejorar la seguridad de la red:

- Actualizar de manera regular las versiones del *kernel* de los sistemas operativos Linux o los service pack de los sistemas operativos Windows instalados en los *hosts*.

- Actualizar las versiones de los programas que proveen los servicios instalados en los *hosts*.

- Actualizar de manera regular las definiciones de virus de los antivirus instalados en los *hosts*.

- Realizar una vigilancia más estrecha en las actividades que realizan los alumnos dentro del laboratorio.

- Actualizar las aplicaciones y servicios que corren bajo el *servidor*.

- No permitir la existencia de cuentas de cualquier servicio que funcionen con los valores por defecto.

- Cuidar la información que se publica en la página web del laboratorio.

- Aplicar un protocolo de cifrado al canal de comunicación con el *switch*.

- Uso de técnicas criptográficas en la autenticación de la red inalámbrica como, por ejemplo, *WEP*. Una mejor opción sería la utilización de claves dinámicas con *WPA* o *WPA2*.

- Desactivar la difusión del identificador de red inalámbrica o *SSID*.

- Administrar la autenticación de la red inalámbrica mediante un servidor RADIUS.

Se incluye dentro de este documento una explicación acerca del ámbito en el que el proyecto fue desarrollado, seguido de los resultados completos de las pruebas y el análisis de los resultados de las mismas.

5.3.2 Ámbito del proyecto

El tipo de *pentesting* realizado fue el de caja blanca, debido a que se tuvo acceso previo a información referente al objetivo, como diagrama de red, direcciones IP, entre otras.

La pruebas concernientes a este reporte fueron realizadas dentro de la red privada de un laboratorio de cómputo de la UNAM, con un rango de IP's comprendido entre la 172.16.1.1 a la 172.16.1.13, excluyendo únicamente la dirección 172.16.1.8. Adicionalmente, se corrieron pruebas en el *servidor* web, *router* inalámbrico y *switch*.

Se permitió el uso de troyanos y *backdoors* en el rango de IP's antes mencionado, más no en los demás dispositivos. Esto para no comprometer el desarrollo de las actividades diarias del laboratorio.

Alrededor de 19200 *exploits* fueron probados en contra del *servidor* y de la red. La mayoría de ellos pueden ser clasificados, pero no limitados, a las siguientes categorías:

- Escaneo de puertos.
- *Fingerprinting*.
- Vulnerabilidades de *servidor* web.
- Vulnerabilidades de *servidor FTP*.
- Ataques de diccionario.
- Vulnerabilidades de administración remota.
- Vulnerabilidades de *switch*.
- Vulnerabilidades de *router* inalámbrico.
- Vulnerabilidad a aplicaciones *backdoor*.

5.3.3 Análisis de resultados

A continuación presentamos un breve análisis de los resultados obtenidos, después de que se aplicaran las pruebas arriba mencionadas.

A. Reconocimiento del host y escaneo

Las pruebas llevadas a cabo en esta fase incluyeron escaneos del tipo *ARP* ping y SYN, además de la detección de sistemas operativos.

En primera instancia, se utilizó la herramienta Nmap dentro de un sistema operativo Windows (escaneo intensivo de todos los puertos TCP) aunque, para obtener resultados más exactos, posteriormente se hizo uso de la herramienta Xprobe2 dentro de un ambiente Linux para comparar los resultados de *fingerprinting*.

La información recopilada y las vulnerabilidades descubiertas se enlistan a continuación. (Véase figura 34.)

Listado de equipos

- IP: 172.16.1.1 (servidor, interna)
  Sistema operativo: Linux *Kernel* 2.6.0 – 2.6.7

  Puertos abiertos

Puerto	Servicio	Versión
21/TCP	ftp	ProFTPD 1.3.1
22/TCP	ssh	OpenSSH 3.6.1p2
80/TCP	http	Apache httpd 2.2.9
111/TCP	rpcbind	2
199/TCP	smux	Linux SNMP multiplexer
443/TCP	http	Apache httpd 2.2.9
1311/TCP		
3306/TCP	mysql	MySQL
5555/TCP		
8000/TCP		
32768/TCP	status	1
32774/TCP		

  Vulnerabilidades

Servicio	Descripción	Recomendación
FTP	Vulnerabilidad SQL injection en los módulos mod_sql_mysql y mod_sql_postgress, asociada a la versión 1.3.1 de proFTPD.	Actualización de la aplicación.
SSH	Vulnerabilidad DoS o ejecución arbitraria de código utilizando buffer_init y buffer_free en buffer.c, asociada a versiones anteriores a la 3.7.1 de OpenSSH.	Actualización de la aplicación.
HTTP	Vulnerabilidad DoS debida a un fallo en el manejo de respuestas excesivas de un servidor origen cuando se utiliza mod_proxy_http, asociada a la versión 2.2.9 de Apache httpd.	Actualización de la aplicación.

- IP: 172.16.1.2
  Sistema operativo: Linux *Kernel* 2.6.8

  Puertos abiertos

Puerto	Servicio	Versión
22/TCP	ssh	OpenSSH 5.0

  Vulnerabilidades

Servicio	Descripción	Recomendación
Linux Kernel 2.6.8	Vulnerabilidad DoS local debida a una falla al manejar conexiones SCTP. Vulnerabilidad de la función cpuset_task_read() dentro de /kernel/cpuset.c, debida a un error underflow, que puede ser explotada para leer la memoria del kernel. Vulnerabilidad debida a una falla del kernel en el manejo de las semillas generadoras de números aleatorios, debilitando así la seguridad de las aplicaciones que confían en el debida generación de los números antes mencionados.	Actualización del Kernel instalado en el host.

- IP: 172.16.1.3
  Sistema operativo: Linux *Kernel* 2.6.9 – 2.6.11

  Puertos abiertos

Puerto	Servicio	Versión
22/TCP	ssh	OpenSSH 4.3
111/TCP	rpcbind	2
42998/TCP	status	1

  Vulnerabilidades

Servicio	Descripción	Recomendación
SSH	Vulnerabilidad DoS de pre-autenticación debida a la posibilidad de incurrir en una condición de carrera en el manejador de señales, asociada a la versión 4.3 de OpenSSH.	Actualización de la aplicación.

- IP: 172.16.1.4
  Sistema operativo: Linux *Kernel* 2.6.9 – 2.6.11

Puertos abiertos

Puerto	Servicio	Versión
22/TCP	ssh	OpenSSH 4.3
111/TCP	rpcbind	2
57951/TCP	status	1

Vulnerabilidades

Servicio	Descripción	Recomendación
SSH	Vulnerabilidad DoS de pre-autenticación debida a la posibilidad de incurrir en una condición de carrera en el manejador de señales, asociada a la versión 4.3 de OpenSSH.	Actualización de la aplicación.

- IP: 172.16.1.5
  Sistema operativo: Linux *Kernel* 2.6.9 – 2.6.11

  Puertos abiertos

Puerto	Servicio	Versión
22/TCP	ssh	OpenSSH 4.3
111/TCP	rpcbind	2
54968/TCP	status	1

  Vulnerabilidades

Servicio	Descripción	Recomendación
SSH	Vulnerabilidad DoS de pre-autenticación debida a la posibilidad de incurrir en una condición de carrera en el manejador de señales, asociada a la versión 4.3 de OpenSSH.	Actualización de la aplicación.

- IP: 172.16.1.6
  Sistema operativo: Linux *Kernel* 2.6.8

  Puertos abiertos

Puerto	Servicio	Versión
22/TCP	ssh	OpenSSH 5.0

Vulnerabilidades

Servicio	Descripción	Recomendación
Linux Kernel 2.6.8	Vulnerabilidad DoS local debida a una falla al manejar conexiones SCTP. Vulnerabilidad de la función cpuset_task_read() dentro de /kernel/cpuset.c, debida a un error underflow, que puede ser explotada para leer la memoria del kernel. Vulnerabilidad debida a una falla del kernel en el manejo de las semillas generadoras de números aleatorios, debilitando así la seguridad de las aplicaciones que confían en el debida generación de los números antes mencionados.	Actualización del Kernel instalado en el host.

- IP: 172.16.1.7
  Sistema operativo: Linux *Kernel* 2.6.8

  Puertos abiertos

Puerto	Servicio	Versión
22/TCP	ssh	OpenSSH 5.0

  Vulnerabilidades

Servicio	Descripción	Recomendación
Linux Kernel 2.6.8	Vulnerabilidad DoS local debida a una falla al manejar conexiones SCTP. Vulnerabilidad de la función cpuset_task_read() dentro de /kernel/cpuset.c, debida a un error underflow, que puede ser explotada para leer la memoria del kernel. Vulnerabilidad debida a una falla del kernel en el manejo de las semillas generadoras de números aleatorios, debilitando así la seguridad de las aplicaciones que confían en el debida generación de los números antes mencionados.	Actualización del Kernel instalado en el host.

- IP: 172.16.1.9
  Sistema operativo: Microsoft Windows XP SP3

  Puertos abiertos

Puerto	Servicio	Versión
139/TCP	netbios-ssn	
445/TCP	microsoft-ds	Microsoft Windows XP microsoft-ds

Vulnerabilidades: Ninguna a destacar.

- IP: 172.16.1.10
  Sistema operativo: Linux *Kernel* 2.6.0, 2.6.2

  Puertos abiertos

Puerto	Servicio	Versión
22/TCP	ssh	OpenSSH 5.0

  Vulnerabilidades: Ninguna a destacar.

- IP: 172.16.1.11
  Sistema operativo: Linux *Kernel* 2.6.0, 2.6.2

  Puertos abiertos

Puerto	Servicio	Versión
22/TCP	ssh	OpenSSH 5.0

  Vulnerabilidades: Ninguna a destacar.

- IP: 172.16.1.12
  Sistema operativo: Linux *Kernel* 2.6.9 – 2.6.11

  Puertos abiertos

Puerto	Servicio	Versión
22/TCP	ssh	OpenSSH 4.3
111/TCP	rpcbind	2
51318/TCP	status	1

  Vulnerabilidades

Servicio	Descripción	Recomendación
SSH	Vulnerabilidad DoS de pre-autenticación debida a la posibilidad de incurrir en una condición de carrera en el manejador de señales, asociada a la versión 4.3 de OpenSSH.	Actualización de la aplicación.

- IP: 172.16.1.13
  Sistema operativo: Linux *Kernel* 2.6.9 – 2.6.11

  Puertos abiertos

Puerto	Servicio	Versión
22/TCP	ssh	OpenSSH 3.9p1
111/TCP	rpcbind	2
32769/TCP	status	1

Vulnerabilidades

Servicio	Descripción	Recomendación
SSH	Esta versión de OpenSSH (3.9p1) no registra la fuente de las conexiones rechazadas.	Actualización de la aplicación.

- IP: 180.20.20.1 (servidor, externa)
  Sistema operativo: Linux *Kernel* 2.6.9 – 2.6.11

Puertos abiertos

Puerto	Servicio	Versión
22/TCP	ssh	OpenSSH 3.6.1p2
80/TCP	http	Apache httpd 2.2.9
443/TCP	http	Apache httpd 2.2.9

Vulnerabilidades

Servicio	Descripción	Recomendación
SSH	Vulnerabilidad DoS debida al orden en que las claves son intentadas en una autenticación de llave pública, asociada a la versión 3.6.1p2 de OpenSSH.	Actualización de la aplicación.
HTTP	Vulnerabilidad DoS debida a un fallo en el manejo de respuestas excesivas de un servidor origen cuando se utiliza mod_proxy_http, asociada a la versión 2.2.9 de Apache httpd.	Actualización de la aplicación.

Diagrama de red

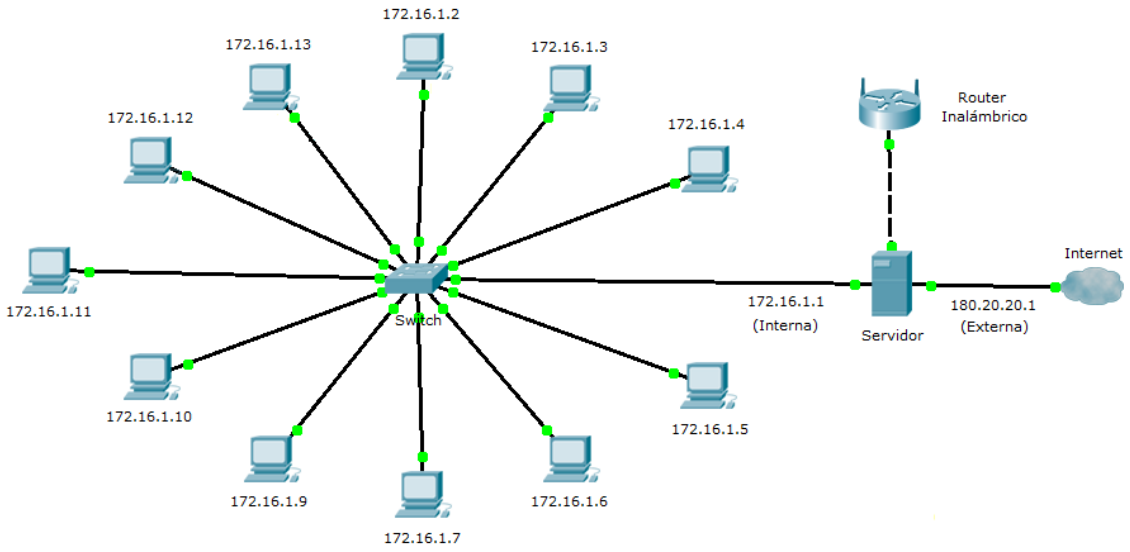


Figura 34. Diagrama de la red del laboratorio de cómputo analizado.

B. Penetración de la red

En esta fase se llevaron a cabo varios tipos de pruebas contra los diversos dispositivos del laboratorio, incluyendo escaneo de vulnerabilidades, ataques de diccionario, ataques *man in the middle*, *sniffing*, MAC *spoofing*, por mencionar algunos.

Una descripción más detallada de las pruebas realizadas en cada uno de los dispositivos y los resultados de las mismas se muestran a continuación.

Servidor

Previo a la realización de las pruebas en el *servidor* del laboratorio, se verificó que la cantidad de tráfico generado por el escáner de vulnerabilidades Nessus no fuera el suficiente para provocar un *DoS* en el dispositivo. Para lograr lo anterior, se instaló temporalmente un *servidor* (Apache 2.2.14) en uno de los *hosts* de la red interna y fue escaneado mientras se simulaba en él una carga de trabajo importante.

Una vez comprobado que la integridad de los servicios no corría riesgos, se escaneó la dirección IP externa del dispositivo, obteniendo como resultado 3 vulnerabilidades de alto riesgo, 19 de riesgo medio y 56 de bajo o riesgo nulo.

A continuación, se detallan los hallazgos más importantes. (Véase tabla 7.)

Servicio	Descripción	Riesgo	Recomendación
FTP	El servidor FTP remoto tiene configurada una cuenta con las credenciales establecidas de manera predeterminada.	Alto	Modificar la contraseña del servidor FTP remoto.
PHP	Diversas vulnerabilidades asociadas a versiones anteriores a la 5.2.7 del tipo buffer overflow en varias funciones, provocando la posibilidad de evadir restricciones de seguridad.	Alto	Actualizar la versión de PHP instalada en el servidor.
SSH	El servicio remoto soporta conexiones hechas con las versiones 1.33 o 1.5 de SSH, que ofrecen protocolos criptográficos inseguros.	Medio	Deshabilitar la compatibilidad con la versión 1 del protocolo.
HTTP	El servidor web remoto contiene un script PHP que permite el acceso a la función phpinfo(), que contiene información crítica acerca del servidor.	Medio	Remover el script PHP.
HTTP	Diversas vulnerabilidades asociadas a versiones anteriores a la 2.2.12 de Apache, por ejemplo, DoS, consumo excesivo de memoria, fallas en módulos de compresión	Medio	Asegurarse que los módulos en dónde se presenten las fallas no estén activados, o bien, actualizar la versión de

	de archivos y de flujo de datos.		Apache.
HTTP	El servidor web remoto soporta los métodos TRACE o TRACK, que son utilizados para depurar las conexiones del servidor.	Medio	Deshabilitar los métodos.
SSL	El servicio remoto soporta el uso de cifradores SSL de fuerza media o débil, es decir, aquellos con longitudes de llave de entre 56 y 112 bits o ningún cifrado en absoluto.	Medio	Reconfigurar la aplicación para evitar el uso de cifradores de fuerza media o débil.
SSL	Diversas vulnerabilidades de cifrado asociadas a versiones anteriores a la 3.0 de SSL, por ejemplo, ataques man in the middle o descifrado de comunicaciones entre los clientes y el servicio afectado.	Medio	Actualizar el servicio SSL a la versión 3.0
Dreamweaver	Dreamweaver produce archivos XML que contienen información de sincronización de archivos y directorios, lo que puede derivar en revelación de información crítica.	Medio	Desactivar la opción "Mantener información de sincronización" de la categoría "Información remota".

Tabla 7. Vulnerabilidades identificadas a través de Nessus.

Tras analizar las vulnerabilidades reportadas por Nessus, concluimos que la manera más viable de lograr un acceso no autorizado al *servidor* era mediante una conexión *FTP*.

Tras la instalación de la suite XAMPP en el dispositivo, llevada a cabo por otros compañeros tesistas, fue creada una cuenta *FTP* a la que no le fueron modificadas las credenciales predeterminadas. Lo anterior nos permitió acceder únicamente a una sección del servidor donde se alojaba la información de un proyecto de tesis y, aunque no fue posible escalar privilegios, pudimos observar los nombres de usuarios de cuentas con presumiblemente mayores facultades.

A través de la aplicación Hydra y de 5 diccionarios, buscamos *crackear* las cuentas observadas en el *servidor* y así poder alcanzar la escalación de privilegios, pero no fue posible debido a la fortaleza de las contraseñas asociadas a ellas.

Switch

En primera instancia, se empleó nuevamente el escáner de vulnerabilidades Nessus en la IP para configuración remota establecida en el dispositivo. Cabe mencionar que esta IP fue adquirida en el documento PDF de la práctica 2A del laboratorio de la materia administración de redes, por lo que es de conocimiento público.

A pesar de que los resultados arrojaron 8 vulnerabilidades de bajo o nulo riesgo, una fue muy relevante:

Vulnerabilidad	Descripción	Nivel de riesgo	Recomendaciones
Servicio de configuración remota sin cifrado	El servicio de configuración remota corre sobre un canal sin ningún método de cifrado, por tanto, los nombres de usuario, contraseñas y comandos viajan en texto claro.	Bajo o nulo.	Cifrar el canal sobre el que corre el servicio.

Conociendo lo anterior, se procedió a la utilización de ettercap y wireshark en un ambiente Linux para ejecutar los ataques *man in the middle* y *sniffing* contra el *switch* y el *host* desde el cual se realiza la configuración remota, con la intención de capturar la contraseña de administración del dispositivo.

La captura se realizó exitosamente mientras alumnos de la facultad experimentaban con la configuración del dispositivo. (Véase figura 35.)

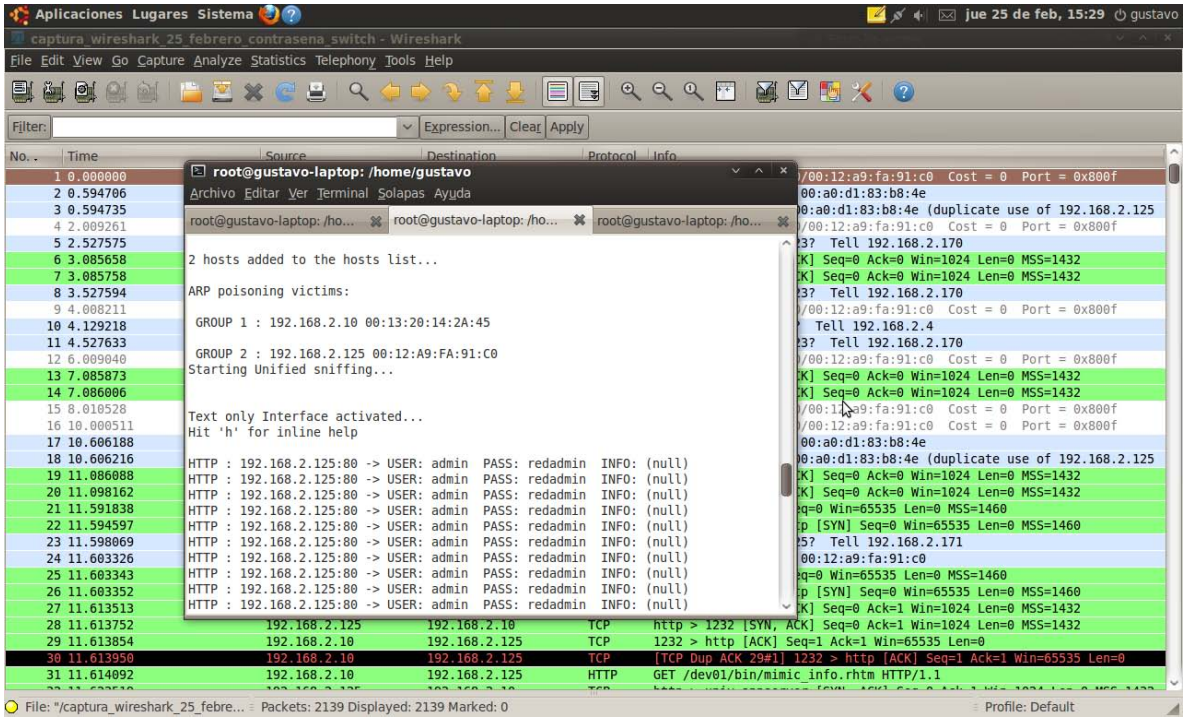


Figura 35. Captura de la contraseña de administración mediante ettercap y wireshark.

Teniendo posesión de la cuenta de administrador, cualquier otra actividad ilícita en el *switch* es fácilmente realizable.

Router

Se utilizó la distribución de Linux WifiSlax 3.1, herramienta especialmente diseñada para la auditoría de seguridad inalámbrica y lo descubierto luego de su utilización se detalla a continuación.

La red inalámbrica del laboratorio no cuenta con ningún método de cifrado en las comunicaciones, ya sea *WEP*, *WPA* o *WPA2* y se tiene habilitada la difusión del *ESSID* de la red. En cambio, sí cuenta con el método conocido como filtrado por MAC para impedir el acceso libre y no autorizado a la red del laboratorio. Sin embargo, ésta única precaución no brinda la seguridad suficiente como para que un usuario ilícito gane acceso a la red.

Haciendo uso de la distribución de Linux WifiSlax en su versión 3.1, se logró tener una amplia visión de las señales de redes inalámbricas captadas por nuestro adaptador, siendo la del laboratorio de redes y seguridad la que nos interesaba. Con la ayuda de la herramienta, pudimos obtener en cuestión de segundos el *ESSID* de la señal además de las direcciones MAC de las tarjetas inalámbricas pertenecientes al *router* y a los equipos legítimos que en ese momento se encontraban conectados a la red además de la potencia con que la señal llegaba. Estos datos no fueron los únicos proporcionados pero sí son los más significativos. Con esta información a la mano, el siguiente paso fue modificar la dirección MAC de nuestra computadora por alguna de las direcciones autorizadas mediante la aplicación para Linux llamada Macchanger. Una vez logrado esto, se pudo autenticarse en la red del laboratorio como usuario legítimo.

Vulnerabilidad	Descripción	Recomendaciones
La red no está protegida por ningún protocolo de cifrado	La utilización de protocolos de cifrado de datos para redes inalámbricas como lo son WEP y WPA son primordiales, ya que se encargan de codificar la información transmitida para proteger su confidencialidad y son proporcionados por los propios dispositivos inalámbricos (sólo los usuarios con contraseña pueden conectarse al punto de acceso).	Implementar algún medio de cifrado, siendo el de WPA o WPA2 los más recomendados.
El modo de difusión del ESSID está activado	Una característica de la conectividad inalámbrica es la capacidad de un adaptador de red inalámbrico de una computadora para buscar una red inalámbrica existente de forma automática. Desactivada la característica "Difusión ESSID", la única forma en que un ordenador puede unirse a la red es estableciendo manualmente en el SSID del ordenador el nombre específico de la red.	Desactivar el modo de difusión del ESSID.

C. Mantener Acceso

Por lo general, mantener el acceso se logra mediante la instalación de un programa *backdoor* dentro de las máquinas víctima y atacante. Dichas aplicaciones tienen dos componentes principales: el programa *servidor*, mismo que se instala en el equipo víctima, y el programa cliente, que actúa sobre la computadora del atacante. Por medio de estos programas, el atacante puede ejecutar remotamente en los sistemas infectados las mismas acciones que el administrador o usuario legítimo del equipo.

En esta parte de la prueba no se nos fue permitida la instalación de ninguna aplicación del tipo antes mencionado en ninguno de los dispositivos activos de la red, salvo en un par de computadoras que usan los alumnos para realizar sus prácticas.

Con estas limitaciones, nos dimos a la tarea de probar la aplicación llamada Back Orifice 2000 utilizando como máquina víctima al equipo con dirección IP 192.168.2.5 y como atacante al de IP 192.168.2.4. La aplicación no necesitó ser instalada en ninguna de las dos computadoras, simplemente se configuró el cliente en la máquina atacante y el *servidor* se configuró y ejecutó en la víctima.

Existen diversos *plug-ins* configurables en la aplicación y éstos pueden dividirse en las siguientes categorías:

- De cifrado.
- De autenticación.
- Propias del *servidor*.
- Propias del cliente.
- De comunicación.
- Misceláneos.

No se utilizó ni cifrado ni autenticación en esta prueba, puesto que no se consideraron necesarias. Se incluyeron todos los *plug-ins* relacionados al *servidor* ya que son estos los que permiten realizar la mayoría de acciones para el control remoto del mismo. En éstos se incluyen funciones como son el *keylogging*, la transferencia de archivos, el control de procesos, entre otros. Los *plug-ins* de comunicación son los encargados de especificar el protocolo propio de la capa de transporte del modelo *OSI* y el puerto a utilizar para la comunicación. Para nuestro caso, se utilizó el protocolo *TCP*, el puerto fue el utilizado normalmente por el programa que es el 54320. Finalmente, se encuentran los llamados misceláneos, e incluyen la utilidad BoPeep y LoveBeads. BoPeep permite realizar el secuestro del mouse y del teclado, esto es, controlarlos remotamente desde la máquina atacante además de que permite el despliegue de una pantalla en la que se muestra en tiempo real, la pantalla de la máquina en la que se está ejecutando

el *servidor*. LoveBeads se utiliza para controlar más de una computadora infectada aunque esta utilidad no fue implementada.

BO2K facilita también algunas opciones para que su funcionamiento permanezca lo más indetectable posible para la máquina infectada como, por ejemplo, activarse al iniciar sesión la computadora víctima esto con el fin de mantener el acceso al equipo. Otra opción es la de ocultar el proceso, ya que sin esto, la actividad será detectada por el *Windows Task Manager* de la computadora infectada como bo2k.exe. Con esta opción activada, el proceso simplemente no aparecerá en la ventana de actividad de Windows.

Para realizar la configuración del cliente, se incluyeron todos los *plug-ins*, a excepción de los concernientes al *servidor*; y ningún valor fue modificado.

Una vez que ambos, cliente y *servidor* fueron configurados y que el *servidor* fue ejecutado, simplemente se procedió a conectarse con la máquina víctima desde el cliente tecleando la dirección IP de la otra.

Una vez conectadas ambas, se procedió a probar los *plug-ins* instalados, pudiéndose listar los procesos que corrían bajo la computadora infectada, listar sus archivos, abrirlos y modificarlos, se probó la aplicación de *keylogger*, el secuestro del mouse y del teclado, la visualización del monitor y el desplegado de mensajes.

Una vez finalizadas las pruebas, todos los procesos involucrados fueron detenidos y los archivos borrados.

Vulnerabilidad	Descripción	Recomendaciones
Consideramos que lograr el uso de este programa no puede considerarse una vulnerabilidad, puesto que este fue consentido por el personal del laboratorio, sin embargo, un usuario mal intencionado podría fácilmente haber dejado corriendo el servidor en el equipo víctima.	Un usuario mal intencionado puede dejar corriendo el programa servidor en la máquina víctima.	Una vigilancia más estrecha por parte del personal con respecto a la actividad de los alumnos que hacen uso del laboratorio.

D. Destrucción de la evidencia

En vista de que gracias al programa BO2K se tiene pleno acceso al equipo víctima, se comprobó que el Registro de Eventos de Windows podía ser borrado con esta herramienta, y junto con él, toda la actividad ilegal producto del uso de este programa, sin embargo, esta acción no se nos fue permitida.

Vulnerabilidad	Descripción	Recomendaciones
Es posible borrar el registro de eventos de Windows mediante la herramienta BO2K	Este servicio graba cada actividad que ocurre en el equipo, como por ejemplo, la modificación o eliminación de archivos, la ejecución de aplicaciones, por mencionar algunas.	Más estrecha vigilancia de la actividad de los alumnos que hacen uso del laboratorio.

5.3.4 Resumen final

La posibilidad de recibir ataques informáticos es un problema que afecta a todo tipo de sitios y que los obliga a implementar políticas de seguridad eficientes, capaces de minimizar cualquier riesgo. El desafío es difícil, porque involucra seguridad física, lógica y capacitación en recursos humanos; difícil más no imposible.

A continuación se presenta un pequeño resumen con recomendaciones para minimizar el impacto de un ataque en cada dispositivo examinado.

A. Red interna

- Realizar actualizaciones de las aplicaciones y servicios que corren bajo los equipos que componen la red interna, instalar parches de seguridad con regularidad.

- Se recomienda realizar una vigilancia más estrecha a las actividades de los alumnos que utilizan el equipo de laboratorio así como al *software* que utilizan.

B. Servidor web

- Realizar actualizaciones de las aplicaciones y servicios que corren bajo el *servidor*, instalar parches de seguridad con regularidad.

- No permitir la existencia de cuentas de cualquier servicio que funcionen con los valores por defecto, ya sea para login, contraseña o ambas. Eliminarlas si no se utilizarán más, o modificar dichos valores.

- Tener cuidado respecto a la información que se publica en la página web del laboratorio.

- Realizar escaneos en busca de vulnerabilidades con regularidad.

C. Switch

- Aplicar un protocolo de cifrado al canal de comunicación con el *switch*.

- Cuidar la información que se publica en la página web del laboratorio.

D. Router

- Cifrar la señal propia del laboratorio, ya sea con protocolos tales como *WEP*, *WPA* o *WPA2*, siendo este último el más recomendable, implementar una contraseña robusta y fomentar la cultura entre el personal encargado del laboratorio del cambio de contraseña cada cierto periodo. Si al protocolo de cifrado se le añaden otras medidas de autenticación como por ejemplo el integrar un servidor *Radius*, sería una adición estupenda a la seguridad de la señal inalámbrica.

- Existen otras medidas deseables como por ejemplo, desactivar la difusión del *ESSID* de la señal.

Conclusiones

La realización de este trabajo nos ha exigido una comprensión sensiblemente más profunda del funcionamiento de las redes de datos y los dispositivos que la conforman, los protocolos que se emplean en ellas, la seguridad en las mismas y algunas de las técnicas más utilizadas para violar dicha seguridad.

Cada una de las fases del proyecto nos aportó no sólo una manera de alcanzar los objetivos que nos planteamos, sino también una experiencia muy valiosa en el desarrollo de nuestra profesión.

Las bases para afianzar nuestros propósitos no se limitaron a los conocimientos adquiridos a lo largo de nuestra formación profesional. La investigación previa nos permitió plantear el entorno y las bases de nuestras pruebas, además de ahondar en puntos finos que complementan temáticas ya conocidas y abordar nuevas problemáticas.

Por otro lado, al conocer más profundamente las metodologías de auditoría de seguridad existentes y elegir el concepto de *pentesting*, pudimos acercarnos a un gran número de herramientas técnicas, explorar sus funcionalidades, sus ventajas, y sus limitantes. Lo anterior no sólo nos permite percibir la manera en que las vulnerabilidades son explotadas, sino que también nos coloca en una mejor posición para prevenir los posibles ataques informáticos a los que nos enfrentemos en el futuro.

La ejecución de nuestras pruebas fue la culminación de una planeación concebida desde las investigaciones previas y la consolidación de conocimientos, hasta la adaptación y seguimiento de procesos propios de una auditoría de seguridad que, en última instancia, nos permitió plasmar por escrito nuestros descubrimientos, nuestras propuestas y la viabilidad de su aplicación.

Al analizar la gran cantidad de información obtenida pudimos apreciar varios posibles riesgos de seguridad con diversos niveles de importancia en la infraestructura del laboratorio.

Para mitigar las vulnerabilidades identificadas, se realizaron varias propuestas, de entre las que destacan la actualización permanente de sistemas operativos y programas, monitoreo más estricto de las actividades que los usuarios llevan a cabo dentro de las instalaciones, así como de la cantidad de información que es divulgada por diversos medios, el cifrado de algunos canales de comunicación y la protección de cuentas y contraseñas.

Recapitulando, el interés de las empresas que encargan este tipo de trabajos a compañías especializadas se centra en descubrir sus huecos de seguridad, qué tan viable es el explotarlos, y qué impacto tendría en la empresa en caso de que algún ataque la vulnerara. Si tomamos esto en cuenta, si suponemos que el laboratorio toma el lugar de la empresa contratante y si revisamos los resultados generados, puede observarse que los beneficios esperados señalados al inicio de la tesis fueron completados puesto que se descubrieron huecos de seguridad reales que comprometían en mayor o menor medida la integridad del laboratorio y pudieron ser implementadas soluciones que robustecieron las medidas de

seguridad ya consideradas anteriormente, lo que se traducirá en mejor desempeño de las funcionalidades del laboratorio y en la reducción de posibilidades que se genere una eventualidad en el mismo.

Glosario

ACK (Acknowledgement)	Tipo de señal de respuesta enviada entre procesos o computadoras que se están comunicando para indicar que la información fue recibida completa y libre de errores.
Análisis forense	Metodología de estudio ideal para el análisis posterior de incidentes, mediante el cual se trata de reconstruir el cómo se penetró en un sistema, a la par que se valoran los daños ocasionados.
API (Application Program Interface)	Se refiere al conjunto de rutinas, protocolos y herramientas necesarias para construir aplicaciones de software.
ARP (Address Resolution Protocol)	Es un protocolo de nivel de red responsable de encontrar la dirección física correspondiente a una dirección IP determinada.
Backdoor	Es una aplicación que permite el acceso remoto a una computadora, evitando una autenticación normal y manteniéndose oculta del sistema.
Banner	Se llama *banner* a la información que transmite un servicio cuando se trabaja con él. Dicha información puede incluir el nombre del servicio, la versión, por mencionar algunas.
Banner grabbing	Técnica de enumeración que trata de usar la información del banner como referencia para conocer los servicios que corren en un equipo protegido por un firewall.
BIOS (Basic Input Output System)	Código de software que localiza y reconoce todos los dispositivos necesarios para carga el sistema operativo en la memoria RAM.
Bit	Es el acrónimo de Binary DigIT. Es la unidad mínima de información empleada en informática. Con él, podemos representar dos valores cualquiera, sea verdadero o falso o 1 y 0.
Broadcast	Es un modo de transmisión de información donde un dispositivo emisor envía información a una multitud de dispositivos receptores de manera simultánea, sin necesidad de repetir la misma información equipo por equipo.
Buffer	Se refiere al área de almacenamiento temporal, reservada para uso en las operaciones de entrada-salida dentro de la cual los datos son leídos o escritos.
Buffer overflow	También conocido como desbordamiento de *buffer*, se refiere a un error de software que se produce cuando se copia una cantidad de datos sobre un área que no es lo suficientemente grande para contenerlos sobre escribiendo otras zonas de memoria.
Byte	Unidad de información compuesta por 8 bits.
Cadena	Secuencia ordenada de longitud arbitraria de elementos que pertenecen a un cierto alfabeto.
Cliente	Se llama así al dispositivo electrónico que recibe servicios de otro dispositivo conocido como servidor.
Cluster	Conjunto contiguo de sectores que componen la unidad más pequeña de almacenamiento de un disco. Los archivos se almacenan en uno o varios clusters, dependiendo de su tamaño de asignación. Sin embargo, si el archivo es más pequeño que un clúster, éste lo ocupa completo.
CPU (Central Processing Unit)	Se llama así al componente central de la computadora donde se realizan las funciones lógicas y aritméticas básicas.
Cracking	Irrumpir en un sistema informático de manera ilegal, quebrantando su seguridad.

DHCP (Dynamic Host Configuration Protocol)	Protocolo de red que permite a los nodos en una red IP, obtener sus parámetros de configuración automáticamente. Trabaja bajo la capa de aplicación del modelo OSI y está definido en el RFC 2131.
DNS (Domain Name System)	Base de datos distribuida y jerárquica que gestiona la conversión de direcciones de Internet expresadas en lenguaje natural a un dirección numérica IP.
DoS (Denial of Service)	Se refiere a una ofensiva diseñada específicamente para impedir el funcionamiento normal de un sistema y por consiguiente, impedir el acceso legal a los sistemas a los usuarios autorizados.
ESSID (Extended Service Set IDentifier)	Véase *SSID*.
Exploit	Es un software que tiene como finalidad automatizar el aprovechamiento de un error, fallo o vulnerabilidad de un programa o hardware.
Fingerprinting	Es el proceso de determinar el sistema operativo instalado en un equipo objetivo.
Firewalking	Técnica creada en 1998 con el objetivo de conocer las políticas de filtrado de un firewall.
Firewall	Dispositivo o conjunto de dispositivos configurados para permitir, limitar, cifrar, descifrar, el tráfico entre los diferentes ámbitos con los que interactúa sobre una base de normas y otros criterios.
Flooding	Es la acción de enviar una cantidad muy grande de información a alguien o a algo para intentar que se sature.
Formulario	Se llama formulario a una plantilla o página que cuenta con espacios en vacíos con el fin de que sean rellenados por un usuario.
Frame	También conocido como trama, es un bloque de datos de una transmisión.
FTP (File Transfer Protocol)	Protocolo de red para la transferencia de archivos entre computadoras conectadas a una red TCP. Se sitúa en la capa de aplicación del modelo OSI. Está descrito en el RFC 959.
Función	Se le conoce así a un subprograma que forma parte de un programa principal que permite resolver una tarea específica.
Gateway	También conocido como puerta de enlace, es un dispositivo activo que permite interconectar redes con protocolos y arquitecturas diferentes a todos los niveles de comunicación. Su propósito es traducir la información del protocolo utilizado en una red al usado en la red destino.
Hacker	Un especialista con grandes habilidades computacionales, que busca obtener acceso no autorizado a sistemas sin ninguna intención maliciosa.
Hash	Se refiere a una función o método para generar claves o llaves que representen de manera casi unívoca a un documento, registro, archivo, entre otras.
Heap	Estructura de datos del tipo árbol que contiene información perteneciente a un conjunto ordenado. Eta región queda disponible para las solicitudes de memoria dinámica al S.O. Su crecimiento va ligado a la disminución de la pila y viceversa.
Heap overflow	Error de software similar al *buffer overflow* que afecta al *heap,* permitiendo un acceso no autorizado a parte de la memoria.
Hipertexto	Es el nombre que recibe el texto que en la pantalla de un

	dispositivo electrónico conduce a otro texto relacionado. La forma más habitual de hipertexto en documentos es la de hipervínculos.
Hipervínculo	Elemento de un documento electrónico que hace referencia a otro recurso, por ejemplo, otro documento o punto específico del mismo.
Host	Término utilizado para referirse a las computadoras que, conectadas a una red, proveen y/o utilizan servicios a/de ella.
Hostname	Denominación otorgada por el administrador a una computadora.
HTML (HyperText Markup Languaje)	Se refiere al lenguaje de marcado predominante para la construcción de páginas web.
HTTP (HyperText Transfer Protocol)	Protocolo de comunicación que utiliza la WWW y que permite la interacción entre los servidores y el navegador. Trabaja en la capa de aplicación del modelo OSI y se encuentra definido bajo una serie de RFC's, siendo el más importante el 2616.
ICMP (Internet Control Message Protocol)	Es un sub protocolo de control y notificación del Internet Protocol (IP) usado para enviar mensajes de error, indicando por ejemplo que un servicio determinado no está disponible. Se encuentra definido en el RFC 792.
IDS (Intrusion Detection System)	Aplicación utilizada para detectar accesos no autorizados a un equipo o red determinados. Normalmente, esta herramienta se integra conjuntamente un firewall.
Interfaz	Punto de interconexión entre dos entidades, sistemas, equipos, conceptos, por ejemplo.
IP (Internet Protocol)	Protocolo de comunicación no orientada a conexión, funciona bajo la capa 3 del modelo OSI. Se encuentra descrito por el RFC 791.
IRC (Internet Relay Chat)	Protocolo de comunicación en tiempo real basado en texto. Se diferencia de la mensajería instantánea en que los usuarios no deben acceder a establecer la comunicación de antemano. Se encuentra definido por el RFC 1459.
Kernel	También llamado núcleo, es el responsable de facilitar el acceso seguro al hardware de la computadora a los distintos programas dentro de un sistema operativo.
keylogger	Tipo de software que se encarga de registrar las pulsaciones que se realizan en el teclado para posteriormente guardarla en un archivo. Generalmente, se usa con fines maliciosos.
LiveCD, LiveUSB	Es un sistema operativo almacenado en un medio extraíble (CD y USB, respectivamente) que puede ejecutarse desde éste sin necesidad de ser instalado. Utiliza memoria RAM como disco duro virtual y el propio medio como sistema de archivos.
Log	Término anglosajón equivalente a la palabra bitácora, se trata de un archivo que registra movimientos y actividades de un programa determinado (*log file*).
Malware	Del inglés malicious software, también conocido como *badware*, término general utilizado para definir una variedad de software malicioso, entre ellos virus informáticos, software espía, troyanos y amenazas similares.
Multicast	O multidifusión, se refiere al envío de la información en una red a múltiples destinos simultáneamente. En oposición al *multicast,* se encuentra el envío de un punto a otro denominado *unicast*.

NetBIOS(Network Basic Input Output System)	Es una API que complementa la BIOS de DOS al agregar funciones especiales para redes locales.
***NTFS* (New Technology File System**	Sistema de archivos de Windows NT incluido en las versions de Windows 2000, Windows XP, Windows Server 2003, Windows Server 2008, Windows Vista y Windows 7. NTFS permite definir el tamaño del cluster a partir de 512 bytes de forma independiente al tamaño de su partición.
OSI (Open System Interconnection)	Es el modelo de red descriptivo creado por la International Standarization Organization (ISO) en 1984 con el objetivo de convertirlo en estándar internacional de arquitectura de redes de computadoras.
Paquete	Cantidad mínima de datos que se transmiten en una red o entre dispositivos. Su longitud varía según el protocolo que los construya.
Pentesting	Es la acción que un hacker ético lleva a cabo para buscar vulnerabilidades potenciales en una infraestructura de red.
Phising	Término informático que denomina un tipo de delito encontrado dentro del ámbito de las estafas cibernéticas caracterizado por obtener información confidencial de forma fraudulenta suplantando a una entidad, generalmente bancaria.
Pila	Véase *buffer*.
Plug-in	Pequeños programas que se agregan a otro ya existente para ofrecer una nueva función, por ejemplo, parches para el navegador que permiten escuchar música o ver videos.
Pop-up	Mensaje que se despliega automáticamente en la pantalla sin antes ser solicitado expresamente por el usuario. Un *pop-up* puede ser una página completa o pequeñas ventanas.
Proceso	Se conoce así a un programa en ejecución.
Protocolo	Conjunto de reglas usadas por computadoras para comunicarse unas con otras a través de una red.
Proxy	Servidor conectado normalmente al servidor WEB que almacena en caché la información recibida por los usuarios, así, si un usuario accede a través del proxy a un sitio anteriormente visitado, recibirá la información del servidor proxy y no del sitio real.
Puerto	Forma genérica de denominar a una interfaz a través de la cual los diferentes tipos de datos se pueden enviar y recibir. Dicha interfaz puede ser de tipo físico o lógico, a nivel de software, en cuyo caso se puede utilizar el término puerto lógico.
RAM (Random Acces Memory)	Se refiere al área de almacenamiento temporal de una computadora desde donde el procesador recibe instrucciones.
Root	En sistemas operativos del tipo Unix, *root* es el nombre convencional de la cuenta de usuario que posee todos los derechos en cualquier modo. Es también conocido como superusuario.
Router	Dispositivo que opera en la capa 3 del modelo OSI, para la interconexión de redes informáticas, que permite asegurar el enrutamiento de paquetes o las rutas que deben tomar los mismos.
Servidor	Se llama así a un dispositivo electrónico que, formando parte de una red, provee servicios a otros dispositivos a los que se les denomina clientes.

Sintaxis	Conjunto de reglas que definen las secuencias correctos de los elementos de un lenguaje de programación.
Sniffer	Programa que monitoriza los paquetes de datos que circulan a través de una red. Tienen diversos usos como la detección de fallos de una red, o la interceptación de mensajes electrónicos, el robo de contraseñas por mencionar algunos.
Spam	Se denomina *spam* o correo basura, a todo tipo de comunicación no solicitada, realizada vía electrónica. De este modo, es *spam* cualquier mensaje no solicitado que tiene el fin de ofertar, comercializar o tratar de despertar el interés respecto de un producto, servicio o empresa.
Spoofing	Se refiere al uso de técnicas de suplantación de identidad, generalmente con fines maliciosos o de investigación. Existen varios tipos: i. IP spoofing, que consiste en sustituir una dirección IP. ii. ARP spoofing, que se refiere a la suplantación de identidad mediante la falsificación de la tabla ARP. iii. DNS spoofing, suplantación de identidad por nombre de dominio. iv. Web spoofing, suplantación de una página web real. v. Mail spoofing, suplantación de la cuenta de correo electrónico de alguien.
SQL (Structured Query Languaje)	Es un lenguaje del tipo declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ésta. Una de sus características es el manejo del álgebra y cálculo relacional, lo que permite efectuar consultas con el fin de recuperar información específica de forma sencilla, o realizar cambios sobre ella.
SSH (Secure SHell)	Dícese del protocolo que facilita las comunicaciones seguras entre dos equipos usando una arquitectura cliente/servidor. Trabaja bajo la capa de aplicación del modelo OSI. Se encuentra referido por el RFC 4251.
SSID (Service Set IDentifier)	Es un código incluido en todos los paquetes de red inalámbrica para identificarlos por parte de una red. A menudo, al SSID se le conoce como nombre de la red. Existen algunas variantes del SSID. Las redes ad-hoc que consisten en máquinas cliente sin punto de acceso, utilizan el BSSID (Basic Service Set Identifier); mientras que en las redes en infraestructura que incorporan un punto de acceso, se utiliza el ESSID (La E significa extendido).
Switch	Dispositivo que opera en la capa 2 del modelo OSI, para la interconexión de redes informáticas, que permite interconectar dos o más segmentos de red, pasando datos de un segmento a otro de acuerdo a la dirección MAC de destino.
TCP (Transmissión Control Protocol)	TCP es un protocolo de comunicación orientado a conexión perteneciente a la capa 4 del modelo OSI. Está descrito en el RFC 793.
Telnet (Telecommunication NETwork)	Protocolo de comunicación entre computadoras del tipo cliente-servidor que permite acceder mediante una red a otra máquina con el fin de controlarla remotamente. Actualmente en desuso por su carencia de cifrado en su comunicación. Trabaja bajo la capa de aplicación del modelo OSI y está referenciado por el RFC 854.

Three-Way Hand-Shake.	Proceso propio del protocolo TCP que ocurre entre un cliente y un servidor cuando se inicia una conexión TCP, esta se realiza en tres pasos: i. El sistema cliente envía un paquete con la bandera SYN al sistema destino. ii. El equipo destino responde con un paquete con las banderas SYN-ACK confirmando la recepción del paquete SYN inicial. iii. Para finalizar, el cliente reconoce la recepción del SYN enviado durante el segundo paso mediante el envío de un paquete ACK. En este momento queda establecida la conexión y la transferencia de datos puede iniciar.
Timestamp	Es una secuencia de caracteres que denotan la hora y fecha en la cual ocurrió un evento. Son típicamente usados para el seguimiento de eventos, siendo cada uno de ellos un log marcado.
Trama	Véase Paquete.
Tunneling	Técnica consistente en encapsular un protocolo de red dentro de otro creando lo que se conoce como un túnel dentro de una red de computadoras. El uso de este técnica puede perseguir diferentes objetivos, como por ejemplo la comunicación *multicast*, redirección de tráfico o el violar las políticas de seguridad de una red.
UDP (User Datagram Protocol)	Es un protocolo mínimo perteneciente al nivel de transporte del modelo OSI documentado en el RFC 768. Generalmente es usado junto a protocolos tales como DHCP y DNS.
Unicast	Se refiere a la comunicación establecida entre un solo emisor y un solo receptor.
WEP(Wired Equivalent Privacy)	Es el sistema de cifrado incluido en el protocolo IEEE 802.11 para redes Wireless. Está basado en el algoritmo RC4, que utiliza claves de 64 bits o de 128 bits.
Windows Task Manager	Utilidad que provee información acerca de programas y procesos que se encuentran corriendo en la computadora, entre otras cosas.
WPA (Wifi Protect Access)	Mecanismo de control de acceso a una red inalámbrica, pensando en la idea de eliminar las debilidades de WEP.
WWW (World Wide Web)	WWW o Web es el servicio de información distribuido, basado en hipertexto, cuya información puede ser de cualquier formato (texto, gráfico, audio, imagen fija o en movimiento) y fácilmente accesible a los usuarios mediante navegadores Web. Fue creado como tal en 1990 por Tim Berners-Lee y Robert Cailliau.

Fuentes

Bibliográficas

- López Barrientos, María Jaquelina y Quezada Reyes, Cintia. *Fundamentos de seguridad informática*, México, Universidad Nacional Autónoma de México, Facultad de Ingeniería, 2006.

- Tanenbaum, Andrew. *Distributed Operating Systems*, Prentice Hall, 1995.

- Whitaker, Andrew y Newman, Daniel. *Penetration testing and network defense*, Estados Unidos, Cisco Press, 2006.

Electrónicas

- "Amenazas y vulnerabilidades" (14/09/2009)
  http://sistemas-de-seguridad-en informatica.blogspot.com/2008/07/amenazas-y-vulnerabilidades.html/

- "Análisis de riesgos. Seguridad Informática" (22/10/2009)
  http://74.125.93.132/search?q=cache:izpLYC9qhlUJ:www.felaban.com/memorias_mayo_09/viernes_15_mayo/santiago_lioza_clain_v4.ppt+p%C3%A9rdida+esperada+seguridad+inform%C3%A1tica&cd=7&hl=es&ct=clnk&gl=mx/

- "Análisis y captura de tráfico de red. Interpretación de segmento TCP. Establecimiento de conexión TCP" (28/01/2010)
  http://seguridadyredes.nireblog.com/post/2008/01/29/analisis-capturas-trafico-de-red-interpretacion-segmento-tcp-ii-establecimento-conexian-tcp

- Anand, Anish y Gadge, Jayant. "Port Scan Detection", IEEE, 2008. (30/04/2011)
  http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4772622/

- "Apuntes de la clase de Seguridad Informática II"
  M. C. Cintia Quezada Reyes. Semestre 2009-2.

- "Ataques a contraseñas" (03/10/2009)
  http://serdis.dis.ulpgc.es/~a013775/asignaturas/iiaso/curso0708/trabajos/seguridad/crack/ataquecontrasena.pdf/

- " BO2K" (02/11/2009)
  http://www.bo2k.com/

- "Borrado seguro de archivos con Shred" (27/01/2010)
  http://www.linuxtotal.com.mx/index.php?cont=info_seyre_008/

- Botero, Armando, Camero, Iván y Cano Jeymi. "Técnicas anti-forense en informática: ingeniería reversa aplicada a TimeStomp", Colombia, Pontificia Universidad Javeriana. (27/01/2010) http://www.fing.edu.uy/inco/eventos/cibsi09/docs/Papers/CIBSI-Dia3-Sesion6%283%29.pdf/

- "Brutus" (05/10/2009) http://www.bujarra.com/Brutus.html

- Chakrabarti, Saikat y Singhal, Mukesh. "Password-Based Authentication: Preventing Dictionary Attacks", IEEE, 2007. (30/04/2011) http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4249815/

- "COBIT Framework for IT Governance and Control" (01/05/2011) http://www.isaca.org/Knowledge-Center/COBIT/Pages/Overview.aspx/

- "¿Cómo evitar que te 'pesquen' en la red? Phising, un peligroso y moderno delito" (10/10/2009) http://www.univision.com/content/content.jhtml?chid=9&schid=1860&secid=11068&cid=995274&pagenum=1/

- "Configuración de una red wireless" (26/01/2010) http://www.adslayuda.com/Redes-configurar_wireless.html/

- "Conoce a tu enemigo: Identificación pasiva" (10/10/2009) http://his.sourceforge.net/honeynet/papers/finger/

- "Contraseñas" (03/10/2009) http://es.kioskea.net/contents/attaques/passwd.php3/

- "Debian Security Advisory" (17/01/2010) http://www.debian.org/security/2007/

- "Definición de Seguridad informática" (12/09/2009) http://www.alegsa.com.ar/Dic/seguridad%20informatica.php/

- "Estándares de evaluación para sistemas de cómputo seguros" (19/09/2009) http://mixtli.utm.mx/Estandares_de_Evaluacion_para_Sistemas_de_Computo_Seguros.ppt/

- "Etercap" (29/10/2009) http://ettercap.sourceforge.net/

- "Firewalking" (25/01/2010) http://www.webopedia.com/TERM/F/firewalking.html/

- "Gestión de riesgos de los sistemas de información" (08/10/2009) http://www.mailxmail.com/curso-gestion-riesgos-sistemas-informacion/identificacion-vulnerabilidades-impactos/

- Ghosh, Shefalika y Nath, Gopi. "Different Flavours of Man-In-The-Middle Attack, Consequences and Feasible Solutions", IEEE, 2010. (30/04/2011) http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5563900

- Haiyan, Liu y Zhiyuan, An. "Realization of Buffer Overflow", IEEE, 2010. (30/04/2011) http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5635047/

- "Information Technology Security Evaluation Criteria (ITSEC)" (18/09/2009) http://www.ssi.gouv.fr/site_documents/ITSEC/ITSEC-uk.pdf/

- "Introducción a la seguridad informática" (12/09/2009) http://es.kioskea.net/contents/secu/secuintro.php3/

- "ISO 27000" (21/09/2009) http://www.iso27000.es/iso27000.html/

- "Laboratorios: Hacking – Técnicas y contramedidas – Ataques por fuerza bruta (Brute Force) II" (04/10/2009) http://labs.dragonjar.org/laboratorios-hacking-tecnicas-y-contramedidas-ataques-por-fuerza-bruta-brute-force-ii/

- "Las vulnerabilidades de software" (27/09/2009) http://hispamp3.yes.fm/2004/01/22/las-vulnerabilidades-de-software/

- "[Linux - security] ALERT: Again: possibility of remote root exploit in openssh" (16/01/2010) http://www.sfu.ca/~siegert/linux-security/msg00002.html/

- "Nmap – Free Security Scanner For Network Exploration & Security Audits" (23/10/2009) http://nmap.org/

- One, Aleph. "Smashing the stack for fun and profit" (29/09/2009) http://insecure.org/stf/smashstack.html/

- "Open SSH" (17/01/2010) http://www.openssh.com/txt/

- "ProFTPD" (16/01/2010) http://www.proftpd.org/

- "RFC 826 – Un protocolo para la resolución de Dirección Ethernet" (01/10/2009) http://www.rfc-es.org/rfc/rfc0826-es.txt/

- "RFC 4732 – Internet Denial Of Service Considerations" (30/09/2009) http://tools.ietf.org/html/rfc4732/

- "Robo de identidad o phising" (10/10/2009) http://www.symantec.com/es/mx/norton/security_response/phishing.jsp

- "Secunia, stay secure" (13/10/2009) http://secunia.com/

- "Seguridad de red inalámbrica Wi-Fi (802.11o WiFi)" (12/01/2010) http://es.kioskea.net/contents/wifi/wifisecu.php3/

- "Seguridad y control informático. Seguridad Canadiense" (21/09/2009) http://cesarisazab.googlepages.com/CLASE2.pdf/

- "Tecnología Inalámbrica. Protección de las redes inalámbricas" (13/01/2010) http://www.cisco.com/web/LA/soluciones/comercial/proteccion_wireless.html/

- "Tenable Nessus" (28/10/2009) http://www.nessus.org/

- "THC-Hydra" (31/10/2009) http://freeworld.thc.org/thc-hydra/

- "Troyanos Backdoor, la nueva generación de vandalismo cibernético" (26/01/2010) http://persystems.net/sosvirus/general/backdoor.htm/

- "Trusted Computer System Evaluation Criteria, Orange Book" (17/09/2009) http://nsi.org/Library/Compsec/orangebo.txt/

- "Trusted Computer System Evaluation Criteria (TCSEC)" (17/09/2009) https://www.ccn-cert.cni.es/publico/2008/401/es/t/tcsec.htm/

- "TSGrinder 2.03" (06/10/2009) http://www.bujarra.com/ProcedimientoTSGrinder.html/

- "Tutorial de Seguridad Informática" (20/10/2009) http://redyseguridad.fi-p.unam.mx/proyectos/tsi/capi/Cap5.html/

- Villalón, Antonio. "Seguridad en Unix y redes", España, 2000. (30/09/2009) http://es.tldp.org/Manuales-LuCAS/SEGUNIX/unixsec-2.1.pdf/

- "What is ITIL?" (01/05/2011) http://www.itil-officialsite.com/AboutITIL/WhatisITIL.aspx/

- "Wireshark" (29/10/2009)
  http://www.wireshark.org/

- "Xprobe2" (25/10/2009)
  http://xprobe.sourceforge.net/