NOTAS PARA UN CURSO DE DOCTORADO SOBRE "SEGURIDAD INFORMÁTICA EN REDES DE ORDENADORES"- V 1.1[1]

Jesús Calabozo Morán

Departamento de Ingeniería Eléctrica y Electrónica, Universidad de León
Escuela Superior de Ingenierías Industrial e Informática,
Campus de Vegazana, s/n, E–24071 Léon, E-mail: diejcm@unileon.es

Contenido

1. INTRODUCCIÓN /MOTIVACIÓN

Este curso se integra en el Programa de Doctorado "Sistemas Inteligentes En La Ingeniería" propuesto por el Departamento de Ingeniería Eléctrica y Electrónica de la Universidad de León para los bienios 2001/02; 2002/03; 2003/04 y con ánimo de continuidad, evidentemente con las adaptaciones que se consideren oportunas en función de la evolución de su interés y el cumplimiento de los objetivos propuestos.

Por ello nos permitimos señalarlo como Versión 1.1.

[1] Este trabajo está soportado por el proyecto de investigación DPI 2001–0105 del MCT.

El grupo de Investigación dirigido por el Dr. Enrique López al que honro en pertenecer, considera prioritario el estudio de la Seguridad en el proceso de tratamiento de la información, las TIC, comenzando su andadura en este campo con la solicitud y posterior concesión de un Proyecto de Investigación por parte del Ministerio de Educación (Español) en relación con la "Detección de Intrusos" como expresión del concepto de seguridad extrapolable a otras áreas del conocimiento, concretamente la de ..Gestión Económica – Contabilidad ....

Esto justifica la composición interdisciplinar del grupo en su búsqueda de la generación de conocimiento mediante la "imbricación disciplinar".

Por otra parte, la demanda tecnológica y social y en el campo de la seguridad de la información, obliga a nuestro entorno académico a considerar esta necesidad dando respuesta a esa demanda en sus vertientes de formación teórica y aplicada, y con la esperanza de creación de un ente en el que concepto de "seguridad" sea el centro de su actividad prioritaria.

A continuación, y a lo largo de todo el texto, se irán realizando algunas disquisiciones sobre la seguridad informática con la sola intención de arropar el frío contenido del programa.

El concepto de seguridad en el proceso de tratamiento automático de la información (informática) no cobra especial interés hasta que ésta se "personaliza"-PC, ya que el mismo concepto de "computación" daba por segura o alcanzada la meta de "fiabilidad", esta personalización hará tomar conciencia de las debilidades en privacidad y un primer paso en la autenticación como medio de acceso al sistema, y claro, los estudiosos ya realizaban, en ensamblador, su primer "*malware*"

El perfeccionamiento de las Interfaces de acceso al los sistemas y de acceso a la programación, las Meta-herramientas para cualquier actividad de procesamiento y sobre todo la RED, nos llevan a la situación actual de máxima capacidad de computación y transmisión pero, sino de mínima, si de baja seguridad.

¡Y la RED! , esa que fue creada para mantener segura la "conexión" pero no tanto la "información"

Si los profesionales ya se hacían preguntas como:

¿esta información es la correcta o ha sido modificada?

¿proviene de donde me dicen?

¿llega a donde quiero?

¿sólo yo tengo acceso a ella o alguien y quien mas?

....

Estas preguntas no son asumidas por la "gente" hasta que la respuesta a ellas no implica una pérdida/ganacia manifestada en credibilidad, papel moneda o privacidad.

La Confidencialidad, Autenticación, Integridad y No-repudio constituyen los objetivos de la seguridad y a ellos nos referiremos a lo largo de este texto.

El concepto de "no repudio" es totalmente inherente a la generalización de las transacciones en Red.

Es la seguridad informática ¡una asignatura imposible!?

A continuación pensaré en voz alta sobre "cuales" son las materias, temas, asuntos... que incluye el concepto de "seguridad en informática", o "qué" se debe conocer para tratar de este tema?

Partimos de la primera pregunta en los estudios de informática

¿qué elementos constituyen un "Sistema Informático" (S.I.) "útil"?

- primera respuesta: Hardware + Software

y claro está, el S.I. existe pero no funciona según lo esperado.

- segunda respuesta: Hardware + Software + Usuario

y ahora la respuesta del S.I. se acerca más a lo esperado o si no ¿qué ocurre con el último PC dotado de la última tecnología y versión de SO y con unas aplicaciones de última generación puestos en las manos de un/a usuario/a sin noción de su utilización?.

Esta primera abstracción nos muestra ya los tres amplísimos mundos inherentes a la informática y que si bien caracterizan la complejidad de su estudio, la especialización la hace llevadera, pero el experto en seguridad deberá bregar de forma enconada en todos ellos.

Situándonos en la visión de la "gente", el S.I. se conforma por... varios Mainframes, PC´s, y otras cajas negras con sus Chips de diferentes fabricantes, sus diferentes SO´s y sus múltiples Aplicaciones que han sido Programadas con también diferentes Lenguajes y que hemos Conectado mediante cables/placas/equipamiento diverso que se comunican mediante

Protocolos, uno de ellos predominante, pero que cada implementador diseña o programa de forma muy similar ¡pero distintas! Y todo ello programado, montado y administrado por diversas personas, cada una con buenas o malas intenciones y con posibilidad de equivocarse……

Pues bien, veamos algunos asuntos que se incluyen en el punto anterior desde la BIOS a las aplicaciones:

- Lenguajes de programación: perl, c, php, asp, xml, java…
- Programación y gestión de SO´s en profundidad (comunicación entre procesos, gestión de memoria, E/S, parámetros de configuración del SO…), consecuencia de inconsistencias, desbordamientos, puertas traseras…
- Administración y gestión de diversas plataformas y entornos (Mainframes IBM, Cray; entornos personales: Microsoft, Mac, Unix, Tipo-Unix, y otros 90 en constante desarrollo y con sus continuas evoluciones…)
- Protocolos a nivel de bit, de red (*Tcp/ip, IPX/SPX, NetBIOS, NetBEUI, ApleTalk, familiaTCP/IP, Wap…)* y de bajo nivel (*Ethernet, Token Ring*).
- Configuración de "cajas negras" como Routers, Switches, Firewalls…
- Criptografía
- Cracking (programación a bajo nivel)
- Utilidades - Herramientas "top 50, 60,…"
- Actualización "diaria" en bugs y vulnerabilidades y consecuente Parcheo de SO y Aplicaciones.
- Técnicas de ataque, clásicas, actuales y…futuras.
- Wireless, ¡otro problema!
- Virus, troyanos, gusanos, spyware, keyloggers
- Configuración de complejas aplicaciones servidoras (Apache, IIS, Sendmail …) y clientes.
- ….etc..etc..etc…

Sirva lo anterior para justificar la gran cantidad de contenidos que quedarán fuera de esta exposición y la reafirmación de que el concepto de la seguridad es un proceso en constante evolución, inabarcable por una

sola persona ni por una sola tecnología y que requiere de estudio y actualización continuados.

2. METODOLOGÍA

Aunque muchos hemos estudiado, visto, incluso practicado, diversas y originales metodologías docentes, el hecho real, que además viene impuesto por el sistema educativo (véanse las inspecciones al aula), se manifiesta en la impartición de clases teóricas presenciales, el desarrollo de unas practicas (en el centro o con olor hogareño) complementado todo ello generalmente con la ejecución de un trabajo personal. Pues este y no otro es el simplísimo y clásico modelo metodológico que se sigue en este curso.

La evaluación .....(asistencia, trabajo).

Y ¿qué guión seguir?

Pues bien, partimos de la Dualidad:

Hombre/mujer Bueno/a ←→ Hombre/mujer Malo/a

Hacker ←→Administrador

(el orden de los factores no altera las definiciones)

De todos es conocido que el "conocimiento" no se puede categorizar en Bueno o Malo, pero si lo es su "utilización"

Siendo la finalidad del curso lograr la protección máxima del sistema (en su acepción más amplia), será requisito previo el dominio de todos y más de los conceptos expuestos y aquellos olvidados y aquellos otros de próxima aparición. Sin embargo, en el ámbito de la Seguridad, habrá que hacer un esfuerzo (algunos ya lo llevan implícito y no requieren de ello) en situarse en posición de Agresor-Malo-Atacante y utilizar todos los conocimientos, herramienta e "ingenio" del que dispongamos con la finalidad de asaltar nuestra propia fortaleza-sistema.

Por ello, el núcleo de la secuencia expositiva a seguir consistirá en 3 pasos:

- `determinación del objetivo`
- `ataque`
- `defensa`

(con este tipo de letra hemos logrado enfatizar el título de la película)

El Entorno o Plataforma en el que se pretende aplicar lo explicado es... ¡cualquiera!, aunque seguramente aparecerá más a menudo la expresión Windows pero no faltará Linux, siendo esta una buena oportunidad para aquellos que todavía no se habían acercado a este sistema.

Como curso de Doctorado, o tercer ciclo que también se denomina ;), es posible la asistencia al mismo por parte de personas interesadas en el tema pero no necesariamente especialistas, aunque generalmente proveniente de áreas afines, lo que facilita la transferencia de conceptos. Esta paradoja del sistema educativo Español (máximo nivel vs. conocimiento relativo) es justificable al considerar el doctorado como "la apertura al mundo de la investigación reglada".

Todo lo anterior viene a justificar que la exposición-presentación efectiva de los temas expuestos en el temario deberá ser realizada de forma ampliamente contextualizada, centrándose en ejemplos paradigmáticos y definiendo trabajos asociados a cada tema de factible ejecución práctica personal. ya!

Asociado a cada tema se propone al alumno la realización de un trabajo personal práctico, con unos contenidos que pretenden la aplicación directa de lo expuesto en el aula. Será la propia afición o interés del alumno el que supervise y juzgue esta actividad.

Aunque a lo largo del curso se citarán un elevado número de herramientas, recomendando encarecidamente su utilización para un mejor conocimiento, en la práctica se utilizarán las herramientas que considero con mayor "eficiencia formativa" aunque para algunos no coincidan con las mas indicadas o destacables en el mundo del análisis de redes...Nmap, Ethereal, Hping2, Iptools, Netcat, Snort, Nessus, Whisker/Nikto.

Hemos comentado que el alumno escoge y desarrolla un trabajo de recopilación y síntesis de información en un determinado ámbito de la seguridad. Este es un aspecto importante ya que los alumnos deberán efectuar una presentación dinámica (ej.: power_point) del trabajo realizado ante los demás compañeros, lo que abre la perspectiva del curso con el acercamiento a unos 20 temas diferentes además de los expuestos en clase.

3. TEMARIO

Tema-1º. Conceptos

Se expone al mundo la inmensidad y complejidad del área a tratar, realizando un amplio desarrollo de lo expuesto anteriormente, con el fin de evitar el desánimo ante déficits en conceptos de seguridad pretendiendo al

mismo tiempo incorporar la amplitud y capacidades de investigación en este campo.

Práctica: Sobre un complejo esquema de sistema de información se deberán identificar la mayoría de posibles puntos y acciones vulnerables.

Pocas personas se han parado a conocer/probar las utilidades de red que vienen con su S.O., aquí haremos este ejercicio.

Tema-2º. Protocolos

El conocimiento de los protocolos a nivel a bit es lo que diferencia al "Hacker" del "Lamer" o de otra forma, al verdadero administrador de sistemas del mago o coleccionista de programitas y/o scripts con alguna aviesa intención.

Este trabajo de aprendizaje de las cabeceras de los protocolos puede resultar, y a veces resulta, desmoralizante pero ¿cómo diseñar una llave sino se conoce el mecanismo de las cerraduras? O ¿cómo entender la sucesión de bits que resultan de una captura de Nmap sin conocer lo anteriormente expresado? O ¿cómo diseñar llaves ¡perdón! Paquetes con Hping2 que tengan un determinado significado? O ¿cómo entender el mecanismo del "*ping of death*"? Y descubriremos como mediante ICMP pueden realizarse escaners muy efectivos y generar túneles que muy posiblemente atravesarán el firewall...

Bueno, pues es aquí donde se dedica un tiempo a introducir el significado de las cabeceras TCP, UDP, IP, ICMP, ARP... y se dejan al alumno para su estudio personal todo el conjunto de la familia de protocolos TCP/IP enviándoles a los RFQ y recomendándoles el "Red Book" de IBM.....

Claro que previamente se ha realizado una rápida exposición comparativa de las capas OSI y TCP/IP, los conceptos de Puerto-Socket y el *"...3-way handshake..."*

Practica: Es evidente que aquí hay que dedicarse a curiosear, Olfatear o realizar *Sniffing* de red y realizar la labor de interpretar los paquetes capturados, es ahora cuando el posible desánimo producido por el estudio de cabeceras comienza a ser sustituido por un interés creciente.

Tema-3º. Búsqueda y determinación del Objetivo

En esta fase se intenta obtener la mayor cantidad de datos relativos al objeto del ataque y como en cualquier guerra ¡todo vale!

Esta sección es especialmente interesante por cuanto además de satisfacer la capacidad de cotilleo o espionaje de casi todo humano y evidentemente permitir el aprendizaje de técnicas que hoy las TIC ponen a nuestra disposición, nos harán percatarnos de cómo nuestra desidia, ignorancia y en muchos casos vanidad, nos hacen expandir multitud de datos personales por ese campo totalmente abierto al público (indiscriminado) que es Internet. ¿Cuánto "currículum vitae" con infinidad de datos personales? y esa actas de reuniones, incluso de organismos oficiales, que se colocan en Red sin ningún mecanismo básico de protección de acceso, para mayor comodidad de su lectura, pero con nombres y apellidos, direcciones y fechas, la información de lo hablado y ...

Pues si, vamos a buscar información utilizando nuestros conocimientos técnicos principalmente pero también nuestro ingenio, perspicacia, osadía, habilidades sociales, si eso, la llamada "ingeniería social"... busque en las papeleras, tome unas copitas con algún jefe..., y ¿cuántos "*Keyloggers*" hay instalados en las máquinas de esos buenos profesionales que utilizan perfectamente el Word y el E-mail pero no conocen nada mas del PC que utilizan?

Pues bien, aquí se verán herramientas de trazado de ruta, pero no en plan *lamer* sino fijándonos en el bit TTL de ICMP.

Interrogaremos a los servicios Whois, Finger, News, busca en Web con las opciones avanzadas, (sobre todo se analizará en la propia web de la organización), extraer información de los DNS, Bases de datos de dominios: RIPE, ESNIC, Listas de distribución, Foros (hay que ver la cantidad de cosas que se cuentan los especialistas!)

Veremos la estructura de la red objetivo mediante herramientas visuales de gestión de red (NDG_software) e introducimos el concepto SNMP. Hay que escanear subredes para ver las IP (Nmap) y claro ver donde están los servidores y que servicios-puertos tienen activos (Hping).

Y como es la primera vez que escaneamos puertos, es una muy buena oportunidad para repasar el *handshake* de la conexión TCP/IP y apreciar, además de comprender lo que hacemos, el rastro que vamos dejando de todo lo que hacemos.

Quede claro que estas herramientas de escáner los hombres-buenos las utilizan para analizar el comportamiento y estado de la red con el fin de optimizar su rendimiento etc. Vale?!

Y ¿cómo determinar el SO de un objetivo? Pues esta es una buena muestra de la complejidad del estudio de la seguridad pues cada diseñador-fabricante implementa las cosas como quiere (pseudoestandarizacion?) y aunque se atengan a los RFC, cada cual implementará la pila TCP/IP a su modo, dando lugar a que la respuesta a determinadas señales no sea la misma en un SO que otro, con lo que se aprovecha este truco para su determinación ¡nada sofisticado, mas bien vulgar, pero efectivo! Y lo llaman "*fingerprinting*". Y si lo aplicamos a lo que responden las aplicaciones al intentar conectar dándonos su nombre y número de versión entonces lo denominan "*footprinting*".

Claro está que podríamos encontrar *passwords* "esnifando" el tráfico de la red, pero eso lo consideraremos en la sección de ataques.

Practica: Las consultas señaladas arriba, insistiendo en News.

Comandos "tracert" y "traceroute" y aplicaciones gráficas de trazado como "NeoTrace"

El software Boy de NDG es sumamente explícito.

Se diseña una petición ARP con *Hping* o *Lcrzoex.*

Tema-4º. Ataques

El interés que muestra el alumno al llegar a esta fase del curso es desbordante pero ciertamente quedará frustrado ante la imposibilidad de poder presentar todas las posibilidades y técnicas que esta sección ofrece.

Nuevamente se insiste en que los ataques están íntimamente relacionados con la tecnología de lo atacado, de ahí la necesidad de su estudio y conocimiento… como ya se atisbó, las mil y una tecnologías asociadas a la informática y comunicaciones (TIC).

Estas agresiones pueden realizarse contra la Red, el sistema o la aplicación, y con el fin de simplemente "curiosear" (con la finalidad que sea) la información o con el fin de "modificar" o "destruir" la información o reducir la eficacia del sistema o mas radicalmente destruir el objetivo (red, host o aplicación), en definitiva un amplia variedad de posibilidades.

Todo profesional de lo ajeno que se precie intentará "ocultar su identidad" y "borrar las huellas-evidencias" de sus intentos. Esta ocultación se realizará robando la personalidad de otro mediante técnicas de *spoofing (MAC-ARP IP, DNS, Web, FakeMail,…), bouncign (proxies, netcat…*) mientras

que el borrado de huellas, del mismo modo que la “escalada de privilegios”, obligará al atacante a un detallado conocimiento del SO, ficheros de configuración y *logs* del sistema, aunque claro está se aprovechará de la ayuda que le ofrecen los *rootKits*.

El tirar (*nuke*) o rebajar la eficacia de un sistema o red, la provocación de “denegación de servicio” (DoS, DdoS), se basa generalmente en realizar una “inundación” (*flooding*)de información, a veces una simple señalización, de forma que se produzca la saturación del ancho de banda disponible o como ya se adelantó ¡dar mas alimento del que se puede digerir! es decir, forzar a la utilización excesiva de un recurso o entretener al sistema en una sola labor.

Otra opción consistiría en ¡proporcionar un alimento que no se puede digerir! O enviar paquetes mal conformados que despistan el funcionamiento del protocolo como el *PingOfDeath.*

Aquí se debe hacer hincapié en que la utilización de herramientas como *Hping* o *Lcrzoex* además de proporcionar un profundo conocimiento de protocolos, nos permitirá analizar la consistencia de la implementación de una determinada pila TCP/IP en un sistema al poder comprobar su funcionamiento ante paquetes especialmente configurados.

Una forma más actual de producir un DoS consiste en explotar vulnerabilidades algorítmicas en muchos de los algoritmos empleados en el software a atacar. La idea clave del ataque es que muchas de las estructuras de datos y algoritmos empleados habitualmente, tienen un buen comportamiento en los casos "normales", pero pueden degenerar a casos "críticos" (en consumo de memoria o de CPU) ante situaciones patológicas.

Es curioso observar que el hecho de que un sistema A sea amigo o tenga una “relación de confianza” con un sistema objetivo B, le puede costar a A ser atacado con el fin de robar su identidad y poder acceder a B evitando así el paso de autenticación (*hijaking*).

El lanzamiento de peticiones de respuesta contra una dirección de *broadcast* generará una respuesta de “todos” los sistemas implicados a la dirección de envío, generalmente una dirección *spoofeada*, y si ese “todos” resulta ser demasiados, provocará una caída del sistema afectado, este es un típico ataque denominado *Smurf.*

Otra forma común de provocar una DoS, o simplemente caida, o como medio para obtener escalada de privilegios, consiste en generar una ruptu-

ra o "desbordamiento de buffer" ya sea por sobrepasar su capacidad o por interferir en su contenido (*format_string*) etc. Como se sabe, la mayoría de las acciones en informática requieren de un Buffer o zona de memoria para almacenamiento de datos ya sea de forma temporal o menos temporal, en general si se logra romper las protecciones ofrecidas por el SO para lograr interferir en las diferentes zonas de memoria implicadas en la ejecución de un proceso (instrucciones, datos, stakc´s) será relativamente sencillo provocar un determinado "*buffer overflow*", muchos de los "gusanos" explotan (*exploit*) este tipo de vulnerabilidad generalmente asociada a servicios como IIS, Apache tampoco se libra, sendmail, fingerd…¡cada día aparece uno nuevo y no se libra nadie, incluso en el propio Ssh!

Comentemos ahora algo sobre los llamados "Bugs de CGI" o "errores en la programación" de unos programas llamados *"Common Gateway Interface"* que nos permiten generar y obtener resultados de una forma dinámica de un servidor Web.

En definitiva el peligro está en que "se ejecuta un programa en el servidor" con los datos que se dan desde el cliente a través de una página HTML. Este programa CGI puede ser escrito con cualquier lenguaje de programación (c/c++, Fortran, Perl, Visual Basic, algun shell de Unix, appelScript, java…) pudiendo por lo tanto ser interpretado o compilado con sus ventajas e inconvenientes ¡cuidado dónde se deja el intérprete pues permitiría ejecutar cualquier programa, además de los cgi´s, si fuera posible acceder a el en el servidor!.

Pues bien, la cantidad de lenguajes utilizados, la dificultad y en la actualidad las prisas por generar código, dan lugar a cientos errores de programación que son explotados por pacientes Hackers. La cantidad de Bugs de cgi que existe es tal que hacen que la comprobación manual de nuestras aplicaciones sea imposible, lo que lleva a que se clasifiquen en grandes bases de datos. Para comprobar que nuestro sistema se encuentra libre de estos errores lo normal es comprobar, a través de esa BdD, la existencia de errores y claro, para ello existen herramientas como Nessus, SARA, Whisker-Nikto etc. los cuales realizarán este trabajo por nosotros de forma automatizada

La mayoría de las veces los ataques pretenderán la obtención de información, privada o privilegiada, pero estos ficheros y transferencias estarán generalmente protegidos por "contraseñas" o *passwords* por lo que nos

va a ser completamente necesario conseguir estas claves y descifrarlas. Los *keylogers* y sniffers pueden ayudarnos en esta labor, pero una vez logrado el acceso a los ficheros de claves el enviar estos a nuestro sistema requerirá previamente de la presencia de una aplicación *ftp* o *tftp* en el objetivo (pues claro, para que la conexión la inicie el objetivo y pueda pasar el *firewall*) o mejor utilizar una de los llamadas "herramientas de administración remota" (RAT) como *BacckOrifice, NetBus, SubSeven*... que nos permitirán realizar todo el trabajo de forma integrada, aunque los actuales anti_virus y anti_spyware suelen dar buena cuenta de estos.

Y ahora a desencriptar-romper claves (*crack*), lo lógico será aplicar el algoritmo a la inversa y como esto generalmente no es muy efectivo lo que hacemos es aplicar el algoritmo de encriptación "a lo bruto" o *brute_force* a un gran número de palabras o "diccionario" hasta que acertamos con la que funciona, al ser muchísimas las palabras lo lógico será automatizar este proceso y eso es lo que hacen *JohnTheRipper* y *Brutus* entre otros.

La ruptura de técnicas anticopia, obtención de números de serie etc. siempre que se realicen dominando las "herramientas de depuración" (*debuggers*) con el consiguiente conocimiento del ensamblador y la estructura de los microprocesadores, puede resultar una actividad muy gratificante en cuanto que ofrece confirmación de esos conocimientos, todo lo contrario a la obtención de passwords o seriales a través de *keygen´s*, web´s o BdD de números de serie que se actualizan quincenalmente.

No hay que olvidarse de los buenos resultados que se obtienen en esto de las claves-passwords con técnicas de ingeniería social, que logran hacer superfluos muchos conocimientos informáticos.

Y que decir de los "Virus", utilizando el concepto en su acepción mas amplia, pues no se va a decir casi nada, solo que cada día son más efectivos, que son verdaderas joyas de programación y que la disponibilidad de herramientas para su creación va a dar muchos problemas.

Temario:

Todo lo anterior quedará reducido al presente Temario, comenzando por presentar un Amplio Esquema de posibles ataques y vulnerabilidades y continuando con:

Estudio de diversos tipos de escaneado con Nmap

Tuneling mediante ICMP (loki)

Spoofing, repaso TCP y ARP

DoS, Flooding SYN

Estudio del Hijaking

Configuraciones de Netcat

Ejemplos simples de Bugs de unicode y Cgi´s

Librería Pcap y Snifers asociados (Ethereal, Zxsniffer, Analyzer)

Práctica: WinNuke, Mail Bomber, Ipspoofing,

NetCat,

Radmin vs. BackOrifice, NetBus, Subseven

Nessus, Whisker/Nitko,

JohonTheRipper, Brutus

Tema-5º. Defensa

Los contenidos de este apartado deben surgir por si mismos como consecuencia de lo analizado en los capítulos anteriores.

Esto es así hasta el punto de que uno de los procedimientos de analizar la seguridad de nuestro sistema consistirá en la realización de ataques contra nosotros mismos y con las mismas herramientas que utiliza el hacker (ahora en posición de administrador de sistemas) el llamado *Penetration_test.*

Por ello se utilizarán esos magníficos programas que son Nmap y Nessus y que ya se emplearon en ataque.

Pues bien, haremos un rápido recorrido por los procedimientos que permiten conseguir los objetivos que plantea la seguridad. De forma sintética podemos decir que para lograr la "confidencialidad" de los datos-información, vamos a tener que ocultarlos mediante técnicas criptográficas (o escondiéndolos con técnicas *esteganograficas*), para lograr la integridad de la información deberemos generar un "resumen" o *hash* (*Md5, Sha-1*…) de la misma que nos confirme la no modificación o alteración de los contenidos, para lograr la "autenticación" (o confirmación de que somos quien somos) deberemos dejar una característica, a poder ser incopiable, de nosotros mismos como los "sistemas bimétricos" (retina, huella dactilar,

muestra ADN (¿)...) o pedir a un tercero "Autoridad de Certificación" que certifique que yo soy yo mediante un pseudo_carnet de identidad o "Certificado", y si además ese Certificado está generado con procedimientos seguros y reconocidos por otra autoridad que también se atreve a certificar a la primera autoridad (la que nos certificó a nosotros) uf¡¡... entonces la información a la que se adjunte este valiosísimo y especial certificado tendrá la consideración de "No Repudiable" con todas las implicaciones y o responsabilidades jurídicas que ello conlleva.

Parece que el mundo de la informática, con esto de la seguridad, se ha convertido en el mundo de las palabras_clave, palabras_de_paso, contraseñas, seriales o de forma general: *passwords*. Y por cierto, ¿con que clave protegemos nuestro conjunto/s de passwords ?...bueno, acabemos con la paranoia.

Pues sí, es el mundo de la claves. Veamos..., nuestra clave de identificación para acceder al CPD, la clave del BIOS, la clave de acceso a la sesión, determinados ficheros de datos tendrán su/s claves, la de nuestro PGP, la clave de un disco o zona del disco muy privada, las de nuestros programas y juegos preferidos, la otra media docena de claves para lo mismo pero ahora aplicadas al portátil, la del correo, las de las web´s: e_bay, paypal, web_calendar,... y todas esas otras webs que cada cual conoce, además de todas las contraseñas de nuestras varias cuentas de correo (he tenido que dejar alguna cuenta porque me he olvidado la contraseña).

En definitiva, parece que nuestro surtido de *passwords* nos va a acompañar por algún tiempo.

Ya que ha surgido el nemónico PGP "*Pretty Good Privacy*" o su versión GNU "*GnuPG*", se aprovecha aquí la ocaqsión para recomendar encarecidamente su utilización, pues si criticable es que un trabajador no responda al rendimiento esperado de su actividad laboral, debido a la utilización de los recursos de la empresa de forma inapropiada, mucho más criticable es la intromisión de los jefes en la vida privada del trabajador olisqueando sus e-mails, navegación, discos etc.

En cuanto a la protección de las "comunicaciones" y sus "servicios" asociados, solo nos queda comprobar que todo funciona correctamente o mejor "seguramente" y lo haremos con las mismas herramientas que los hackers (aunque quizás ahora, dado que paga la organización, serán *tools*

comerciales) de forma que cuando descubramos un agujero o vulnerabilidad apliquemos el parche correspondiente... aunque lo normal es que sean otros quienes descubran los agujeros, por lo que será conveniente que nos suscribamos a un "servicio de alertas".

Es evidente que cuantos menos programas-servicios se ejecuten en una máquina, menos posibilidades de ejecutar, producir errores y abrir puerta/os tendremos, entonces, aplicando esta evidencia, conviene ejecutar solo los servicios estrictamente necesarios para la organización.

Si como ya se comentó, la confidencialidad de la información pasa por la codificación y/o cifrado de la misma, convendrá crear "túneles" virtuales por los que la información pase cifrada y que fuercen a una autenticación en la conexión, para ello se utilizarán los protocolos seguros que mas convengan (*ssl, ssh, ipsec...*), *stunnel* es sumamente didáctico y práctico! Lo anterior complicará o hará inútil la escucha del "hombre en medio" (*Man_ in_ the_ midle*)

Y ahora comentaremos escuetamente esa barrera de contención o "cortafuegos" (*firewall*) en la que muchos de los administradores ponen todas sus esperanzas de seguridad, las cuales, evidentemente, se verán frustradas por ataques, algunos de ellos de lo mas primitivo.

La configuración de un cortafuegos, como se sabe, no es tarea simple y por ello hay que dedicarle atención y consecuentemente tiempo. Quizás sea esta dificultad en la definición de reglas lo que proporciona al administrador esa sensación de seguridad, una relación absurda pero que existe.

El tratamiento de este tema en este curso, se reduce a la presentación da varios esquemas de situación en la red y a la explicación de un ejemplo de regla "iptables/ipchains"**.**

Lo expuesto, no debe restar importancia a la gran responsabilidad que el/los firewalls tienen en la cadena de seguridad de los sistemas informáticos.

Es sumamente recomendable prevenir la "troyanización" de procesos de SO, aplicaciones y las modificaciones no supervisadas de los ficheros de configuración del sistema u otros datos o BdD..., la solución mas empleada pasa por realizar un Hash del archivo en cuestión y mantener alguna aplicación, tipo *Tripware* o *Chkrootkit,* que compruebe que esa firma no ha sufrido modificación a lo largo del tiempo.

La definición de la Política en el uso de los recursos asociados a las TIC en la organización aparece como requisito indispensable al pretender conseguir objetivos de seguridad. Estas normas, que serán de obligado cumplimiento, proporcionarán un marco de utilización de los recursos y por lo general limitarán los accesos y utilización indiscriminada de los mismos. Esta Política facilitará el ajuste y refinamiento del funcionamiento de los demás elementos que intervienen en la seguridad.

Podemos concluir afirmando que en la actualidad no es posible conseguir los objetivos de seguridad (autenticación, integridad, confidencialidad y no repudio) con la incorporación de una única tecnología (firewall, IDS, auto_test, certificados, protocolos seguros...), y resulta requisito indispensable la definición y estricto cumplimiento de una "Política" de seguridad en la organización.

Temario:

Presentación general de los conceptos arriba expresados, con incidencia en seguridad de la informática personal, dado que la mayoría no son administradores de sistemas se pretende al menos, una inmediata incorporación de los conceptos adquiridos es este curso.

PGP y firma electrónica

Certificados y sistema PKI

Protocolo SSL e Instalación de Stunnel como cliente y servidor.

Análisis de dos /tres reglas Ipchains

Redacción de una Política

Práctica: Se complementa el conocimiento de herramientas ya empleadas en el apartado de ataques.

Seguridad personal:

Anonimato en conexiones, navegador, mail, telnet... proxies (Proximitron) y cifrado (Ddcript, PGP)¡¡ sobre anonimato se cita una de las mejores webs al final!...la "steganografia" es también muy práctica (Camouflage)

Firewall (FW-1, Kerio, ZoneAlarm)

Antivirus

Anti_keyLoggers y Anti_Spyware (Adaware, Spybot)

Tema-6º. Sistemas de Detección de Intrusos (IDS)

Por ser este el tema de investigación que dio pie a la escritura de este documento, antes de exponer el resumen del temario del curso referido a los IDS, me permitiré hacer algunas consideraciones en relación con estos sistemas.

No es mi intención hacer aquí un estudio y clasificación de los IDS, para ello les remito a la página:

http://www.networkintrusion.co.uk/ids.htm

Incluiremos en el nemónico IDS todas variantes de este concepto: a nivel de red (NIDS), host (HIDS)y aplicación, de respuesta activa o pasiva, detectores de anomalías o abusos etc.

Pues bien, ¿Cuántos tipos de intrusiones-abusos-anomalias-vulnerabilidades se conocen? ¿Cuantos se generan diariamente? ¿Cuántas posibilidades existen?...

La respuesta a estas cuestiones y en concreto la suma de todas ellas nos hace pensar que con el estado actual de la tecnología la detección de intrusos en TR, aun siendo factible, no parece funcional.

Vemos proyectos del máximo interés que se manifiestan exitosos pero ello se produce cuando su aplicación se realiza en ámbitos *restringidos* ya sean vulnerabilidades asociadas a un determinado protocolo, SO, servicio o aplicación etc.

En absoluto se pretende aquí realizar una critica destructiva de estos sistemas, al contrario la cantidad y complejidad de problemas-ataques-comportamientos a prevenir es de tal magnitud que la constante investigación y desarrollo en este campo puede generar resultados imprevisibles siempre que ello se realice de forma coordinada y ateniéndose a una norma o estandarización.

Esta estandarización se puede considerar absolutamente necesaria al menos en lo concerniente a los formatos de E/S de la información que trate "cada" sistema, no importando el procedimiento-tecnología de procesamiento de cada uno de ellos.

Y digo "cada" porque solo puedo comprender el SDI como un conjunto de "subsistemas-módulos" integrables en algún punto y que permitan su adaptación y configuración en función de los requisitos específicos de cada entorno de aplicación.

Esto refuerza la idea de la necesidad de definir una "política" en los entornos de los Sistemas de información de cada organización.

Me atrevo a presentar un primer modelo con una estructura de capas (similar al estudio de redes) que evidentemente puede y debe ser perfeccionado pero que permita transmitir la idea de módulo-subsistema:

1	Aplicación/ Host/ Red	
2	sensor-detector	Agentes...
3	Filtrado-3	Redes neuronales...
4	recolector	.....
5	transmisión	Tcp/ip.....
6	almacenamiento	BdD.....
7	Filtrado-7	.....
8	análisis-procesamiento	DataMining....
9	Salida	.....
10	alertas	Sist.experto,
11	Filtrado-11	.....
12	actuador	.....

Observaciones:

- La interfases de E/S las constituyen las líneas divisorias entre capas y sus formatos deberían estar estandarizadas.
- Hay capas que pueden o no existir, por ejemplo puede no requerirse la 6 si el procesamiento se realiza en TR, o la 5 si el procesamiento lo realiza la misma estación recolectora, o la 12 si no se requiere respuesta activa por parte del IDS, etc.
- Los filtros, también opcionales, permitirían regular-afinar la adaptabilidad y respuesta, en definitiva, el funcionamiento global del sistema.
- La adaptabilidad del sistema podría conseguirse incorporando realimentación entre diferentes capas, lo que ciertamente añadiría un alto grado de complejidad pero quede aquí expuesta la idea.

Este modelo permitiría integrar todos los métodos de análisis actuales y futuros ya se basen en detección de umbrales (uso de CPU, saturación de red...), medidas estadísticas a partir de históricos de valores, medidas basadas en reglas con sistemas expertos, algoritmos no lineales basados en sistemas bioinspirados (redes neuronales, sistemas inmunes o inmunológicos, algoritmos genéticos, hormigas...), extracción de conocimiento basándose en sistemas de Data-Mining, mecanismos de transición de estados, redes de petri... estudiando su aplicación en las diferentes capas y comprobando sus rendimientos.

Finalizo esta exposición citando las limitaciones actuales de los IDS, basándome en un boletín del "National Institute of Standards and Technology." (NIST) redactado por Rebecca Bace y Peter Mell:

- No tiene escalabilidad para redes de organizaciones muy grandes o distribuidas.
- Tiene una gestión difícil, con un control complicado.
- Los IDS comerciales raramente interactúan entre si.
- Los IDS comerciales generalmente no interactúan con otros paquetes de gestión de res o seguridad.
- Producen un significante número de errores, especialmente de falsos_ positivos, lo que produce una considerable pérdida de tiempo y recursos.
- No pueden compensar la existencia de deficiencias significativas en la organización como: estrategias de seguridad, política o arquitectura de seguridad.
- No pueden compensar las debilidades propias de los protocolos de red.
- No pueden sustituir a otros mecanismos de seguridad (como la identificación o autenticación, encriptación, certificación, firewalls o control de acceso.

Ante la Imposibilidad de determinar el éxito del ataque por parte de los IDS, es posible complementar estos con la utilización de los llamados "*Honeypots*" con la muy gráfica traducción de "tarros de miel", los cuales al permitir al atacante acceder a los recursos cebo o trampa que se han proporcionado y ejecutar sobre ellos todas sus artimañas, brindan la posibilidad de realizar un análisis de los métodos y técnicas de ataque y de obser-

var perfiles comportamientos los cuales podrán servir a la ora de ajustar los IDS y definir políticas de seguridad corporativa.

Estos sistemas demandan tiempo para el análisis de los datos obtenidos, lo que puede suponer un grave inconveniente al suponer un coste excesivo en recursos humanos y técnicos.

Práctica: En este tema, tras una presentación de lo expuesto, tanto la teoría como la práctica se basarán en el estudio e instalación de Snort y su IDScenter, su configuración y diseño de reglas.

4. BIBLIOGRAFÍA, RECURSOS, etc.

¿Qué bibliografía recomendar?

Si de algo disponemos hoy (al menos en esta parte del mundo) es de información, mucha información y MUY BUENA información y además ¡gratis!

Posiblemente vosotros tengáis mejores referencias que las que yo cite pero... os aseguro que estas son muy buenas. ¡No las desaprovechéis!

- Pues primero lo básico, no por fácil, sino por fundamental, y la podemos encontrar en la Web de IBM, sus "Red Books" en concreto:

  "TCP/IP Tutorial and Technical Overview"

  http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/gg243376.html?Open

  - También la web de CISCO nos proporciona información fundamental.
  - No olvidar la Web de Microsoft.
  - Y aquí un ejemplo de lo que ofrece O´Rreilly:

    "CGI Programming on the World Wide Web"

    http://www.oreilly.com/openbook/cgi/

- Y dentro de los múltiples y buenos textos que existen, seleccionaré alguno que me parece mas completo y entretenido:
  - "SEGURIDAD EN UNIX Y REDES"
    www.rediris.es/cert/doc/unixsec/unixsec.pdf

  - o "Guía de Administración de Redes con Linux" http://es.tldp.org/Manuales-LuCAS/GARL2/garl2/
  - o "Análisis de Seguridad de la Familia de Protocolos TCP/IP y sus Servicios Asociados" http://es.tldp.org/Manuales-LuCAS/doc-seguridad-tcpip/Seguridad_en_TCP-IP_Ed1.pdf
  - o "Libro Electrónico de Seguridad Informática y Criptografía en Diapositivas Versión v 3.1" http://www.lpsi.eui.upm.es/SInformatica/descarga/SItemas.zip

- Ataques, Vulnerabilidades, Malware etc...
  - o The Common Vulnerabilities and Exposures List (CVE) http://cve.mitre.org./cve/
  - o En la Metabase ICAT existen enlaces a más de 6100 vulnerabilidades: http://icat.nist.gov/icat.cfm
  - o Una Web que nos informa de ataques/virus/toyanos/spyware referidos a puertos: http://www.iss.net/security_center/advice/Exploits/Ports/default.htm
  - o Lista de Intrusiones detectada por NetworkICE http://www.iss.net/security_center/advice/Intrusions/
  - o TCP/IP Attacks http://cs.baylor.edu/~donahoo/NIUNet/hacking.html

- Webs de recursos que os guiará en este viaje:
  - o CERIAS, The Center for Education & Research in Informattion Assurance & Security http://www.cerias.purdue.edu/
  - o No me agradezcáis esta dirección, pero sí a su creador http://www.ausejo.net/

- Herramientas
  - o Top 50 security tools

  - Tools & Utilities¡¡¡¡¡¡¡¡¡ te la doy impresa en Anexo.
    http://www.devhell.org/~rix/tools/index.en.html#Network_security
  - Las 75 Herramientas de Seguridad Más Usadas
    http://www.insecure.org/tools/tools-es.html
  - EPIC Online Guideto Practical Privacy Tools
    http://www.epic.org/privacy/tools.html#surf
    http://www.privacy.net/remailer/
  - Ejemplo de herramientas on-line…y más cosas…
    http://www.hackerscenter.com/onlinetools/index.htm

- Para Aplicaciones:… pues las web´s de cada aplicación

- ¿Cuantos SO´s conoces? ¡demasiados?!
  - Mira en esta dirección
    http://www-hiraki.is.s.u-tokyo.ac.jp/members/nobukuni/full.html

- Hay que estar alerta:
  - Cert
    http://www.cert.org/advisories/
  - Hispasec
    http://www.hispasec.com/

- Y como final de la parte de referencias, ante cualquier duda, a "goo-glear" ¡!!

  … aunque "TEOMA" también nos será útil.

ANEXO- I:

Relación de trabajos surgidos como aplicación de dicho programa.

1. Sistemas de Detección de Intrusos basados en Agentes y Sistemas Expertos.
2. Seguridad en redes de ordenadores: metodologías, técnica y herramientas.
3. Auditoría informática
4. Herramientas de auditoría de red.
5. Troyanos y puertas traseras.
6. Iptables, cortafuegos en Linux.
7. Puertos y protocolos vs. ataques y defensas.
8. Ipv6_IPSec y proyectos sobre Ipv6.
9. Redes privadas virtuales (VPN).
10. Protocolos IPSec, SSL, Smime.
11. Autonomoues Agents for Intrusión Detection (AAFID).
12. Escáner de vulnerabilidades NESSUS.
13. Firewalking.
14. SATAN.
15. Sistemas de Detección de Intrusos: SNORT.
16. Vulnerabilidades, ataques y sistemas de defensa asociados a los protocolos ICMP e IGMP.

ANEXO-II

Una Página de Recursos y Herramientas impresa:

...pretendo reflejar lo expresado al comienzo de este texto sobre la cantidad de materias que requiere el dominio de la seguridad informática...

http://www.devhell.org/~rix/tools/index.en.html#Auditing_-_Debugging_-_Reversing_Debugging_Memory

TOOLS & UTILITIES

Auditing - Debugging-Reversing:

Code browsers:

CScope (*Unix*, *Windows*): Command line C/C++ code browser, who helps to browse using references, declarations and functions (functions called by another function, or functions calling the current function). Can be integrated with Emacs and VI.

Hypersrc (*Unix*): C/C++ code brower, through a nice GUI interface.

Source Navigator (*Unix*, *Windows*): Powerfull tool supporting a lot of languages, who permits to browse symbols, hierarchies, classes, cross-references, includes, to realizate grep,... It also supports different version control systems, debugging and compiling shortcuts, and contains a SDK.

Understand for C++ (*Unix*, *Windows*) [*Closed source*, *Non-free*]: Powerfull tool to browse, document, and develop C/C++ source code, through a nice GUI.

Code scanners:

FlawFinder (*Unix*): Analyze C++ code to find vulnerabilities.

LCLint (*Unix*): C code static analyzer/debugger, who helps to detect wrong programmation mechanisms and vulnerabilities. The user can help the analysis by providing comments in the source code.

RATS (*Unix*): Analyze C/C++ code to find dangerous function calls.

Splint (*Unix*, *Windows*): Tool to statically test and detect vulnerabilities in C programs. The user can help the analysis by commenting program sources, to give informations to the analyser.

Debugging:

Data Display Debugger (*Unix*): Graphical front-end who can be associated to lots of debuggers (gdb, dbx, jdb, Perl & Python debuggers, ...) and who permits to print data structures as graphs, or to easily follow variables evolution.

IceDump (*Windows*): Plugin who extends SoftIce with new commands and features: memory dumping, memory loading, process dumping, process killing, thread suspension, thread resume, clipboard manipulations, automatic tracing, ...

Insight (*Linux*): Nice graphical front-end to GDB, who permits lots of interesting manipulations.

Private Ice (*Linux*): Powerfull system/kernel debugger, like SoftIce, but for Linux.

SoftIce (*Windows*) [*Closed source*, *Non-free*]: Famous system/kernel debugger for Windows, who permits to debug and to realize device drivers.

W32DASM (*Windows*) [*Closed source*, *Non-free*]: Windows Debugger/desassembler, easy to use.

Hijacking:

Detours (*Windows*): Tool who permits to statically or dynamically hijack functions calls, or to add data in an executable.

Logging:

CyberSensor (*Windows*) [*Closed source*]: Tool who permits to spy Win32 APIs calls, eventually remotely. Also permit to develop your own plugins.

Fenris (*Linux*): Debugging tool who permits to log all system calls realized by a process, to print parameters for those calls, and the evolution of data.

STrace (*Linux*, *Windows NT - 2000*): Debugging tool who permits to log all system calls realized by a process.

System Call Tracker (*Linux*): Debugging tool who permits to log specific system calls on all the system.

Memory:

LordPE (*Windows*) [*Closed source*]: Tool to edit/view parts of PE files, dump them from memory, optimize them, ...

MemSpy (*Windows*) [*Closed source*]: Dump, search and watch a process's memory, using an explorer like interface.

Allocation:

Debauch (*Linux*): Help to dynamically detect memory allocation/deallocation problems, by hooking originals function calls to the C standard library.

Dmalloc (*Unix*): Library who permits to debug and analyze memory allocations and deallocations on the heap.

MemWatch (*Unix*): Tool who permits to detect memory problems: free(), overflows, segmentation faults, ...

Decompilers - Desassemblers:

BIEW (*Windows*, *DOS*, *Linux*, *FreeBSD*): Text/hexadecimal editor + desassembler, who supports lots of executable formats.

W32DASM (*Windows*) [*Closed source*, *Non-free*]: Windows Debugger/desassembler, easy to use.

WingDis (*Java*) [*Closed source*, *Non-free*]: Really powerfull Java decompiler.

IDA Pro:

2pelf (*Windows*): Plugin who permits to create FLIRT signatures for i386 ELF objects.

Desquirr (*Windows*): Plugin who permits to decompile x86 code and to generate C code. ? ?

IDA Pro (*Windows*) [*Closed source*, *Non-free*]: Famous interactive desassembler, supporting lots of file formats and processors.

OBJRec (*Windows*): Plugin which helps in the reconstruction of unknown structures and classes in x86 binaries.

x86grph (*Windows*): Utility that generates flowgraphs from x86 code. It is capable of graphing non-contiguous functions.

Executables editors:

BIEW (*Windows*, *DOS*, *Linux*, *FreeBSD*): Text/hexadecimal editor + desassembler, who supports lots of executable formats.

ELF:

ELFkickers (*Linux*): Set of tools to create and manipulate non conventional ELF executables.

ELFsh (*Unix*): Interactive shell who permits to manipulate ELF executables.

PE:

Detours (*Windows*): Tool who permits to statically or dynamically hijack functions calls, or to add data in an executable.

LordPE (*Windows*) [*Closed source*]: Tool to edit/view parts of PE files, dump them from memory, optimize them,...

PEexplorer (*Windows*) [*Closed source*, *Non-free*]: Explorer/editor who permits to analyze and modify informations contained in PE executables, for example by allowing to comment functions calls in the code, or by allowing to modify resources.

Resources:

ExeScope (*Windows*) [*Closed source*, *Non-free*]: Utility to analyze, display and rewrite resources of executables files (EXE, DLL, OCX,...).

Resource Hacker (*Windows*) [*Closed source*]: Resource editor directly in Windows executables.

Scanners:

Language 2000 (*Windows*) [*Closed source*]: Utility to detect compilers and compressors.

Coding:

Assemblers:

GASP (*Unix*): Utility who "compiles" a tool to easily manipulate different records of a data structure.

Intel2GAS (*Unix*): Utility who convert MASM, TASM, and NASM assembler sources to GNU Assembler (GAS) sources.

NASM (*Unix*, *Windows*, *DOS*): Freeware portable assembler for Intel processors. NASM supports lots of executable formats on different plateforms, and respect the classical Intel syntax. It's firstly an assembler near your processor, and independant of the plateform. It can also be used as a backend for the LCC compiler.

Assertions:

Nana (*Unix*): Tool who permits to realize C/C++ assertion tests, using GCC compiler.

Comparisons:

Delta (*Linux*, *Windows*, *DOS*) [*Closed source*]: Powerfull files and directories comparison tool, in a nice text interface.

Merging:

Araxis Merge (*Windows*) [*Closed source*, *Non-free*]: Powerfull files/directories comparison and merging tool (2 or 3 files), with a nice interface, command line possibilities, and an API.

Compilers:

C++ Builder Compiler (*Windows*) [*Closed source*]: Borland's famous C ANSI compiler, in a really interesting free version.

LCC (*Unix*, *Windows*): Retargetable C compiler, who generate ASM code for Alpha, Sparc, MIPS and x86.

Documentation:

DOC++ (*Unix*, *Windows*): Automated documentation system for C, C++, IDL and Java langages, based on an analysis of commentaries available in the source code. C/C++ IDE for Gtk/Gnome.

IDE:

Anjuta (*Linux*): C/C++ IDE for Gtk/Gnome.

Open Perl IDE (*Windows*): Perl IDE for Windows.

Source Navigator (*Unix*, *Windows*): Powerfull tool supporting a lot of languages, who permits to browse symbols, hierarchies, classes, cross-references, includes, to realizate grep,... It also supports different version control systems, debugging and compiling shortcuts, and contains a SDK.

Understand for C++ (*Unix*, *Windows*) [*Closed source*, *Non-free*]: Powerfull tool to browse, document, and develop C/C++ source code, through a nice GUI.

Emulation:

IA-64 Linux Developer's Kit (*Unix*): Emulator, environment, Linux kernel and developer's kit to manipulate IA64 architecture.

Plex86 (*Unix*): PC emulator based on virtualization, who permits to install differents OS.

Simics (*Unix*) [*Closed source*, *Non-free*]: Powerfull simulator for PC (x86 and x86-64), SPARC V9, PowerPC, and Alpha architectures. Also permits to emulate multiprocessor systems, as well as clusters and networks of systems.

VMWare (*Unix*, *Windows*) [*Closed source*, *Non-free*]: Powerfull PC emulator based on virtualization, who permits to install lots of OS, virtual networking, and options for hardware virtualization. For example, you can choose to commit ou cancel modifications to hard disks.

Libraries:

a386 (*Unix*): C programming library providing a virtual machine who is an abstraction of an Intel 386 running in protected mode.

Linux:

Linux a386 (*Unix*): Port of Linux to the a386 architecture. It permits to run Linux kernel as a normal Unix process.

Encryption:

GNUPG (*Unix*): PGP encryption.

MCrypt (*Unix*): Encryption library to replace crypt, and who supports lots of powerfull modern algorithms.

Executables:

BurnEye (*Linux*, *Windows*) [*Closed source*]: Utility to encrypt ELF executable files.

SecurePE (*Windows*) [*Closed source*]: Utility to encrypt PE executable files.

Steganography:

Invisible Secrets (*Windows*) [*Closed source*, *Non-free*]: Windows Debugger/desassembler, relatively powerfull and easy to use. Powerfull tool, who permits to insert data into JPG, PNG, BMP, HTML and WAV files.

MP3stego (*Unix*): Steganography at encoding time, using MP3 files.

OutGuess (*Unix*): Steganography using PNM and JPG files.

Snow (*Unix*): Steganography using ASCII files.

StegFS (*Linux*): Steganographic file system.

StegHide (*Unix, Windows*): Steganography using JPEG, BMP, WAV and AU files.

Host security:

Trinux (*Linux*): Linux distribution on some disquettes, who offer lots of hacking/forensic/ security tools.

Wipe (*Unix*): Delete files in a nonreversible way.

Files integrity:

AIDE (*Unix*): Files and directories integrity analyzer, based on Tripwire.

CryptoMark (*Linux*): Analyze files integrity by verifying signatures at run time.

Samhain (*Unix*): File integrity and host-based intrusion detection system, with kernel modules detection, and centralized monitoring to databases.

Tripwire (*Unix, Windows NT - 2000*) [*Closed source, Non-free*]: Files and directories integrity analyzer.

Tripwire Open Source (*Linux*): Files and directories integrity analyzer.

Logging:

BackLog (*Windows NT - 2000*) [*Closed source*]: Utility who permits to store Windows events logs to a distant syslog Unix demon.

LogCheck (*Unix*): Famous logs analyzer.

HIDS:

CyberSensor (*Windows*) [*Closed source*]: Tool who permits to spy Win32 APIs calls, eventually remotely. Also permit to develop your own plugins.

Samhain (*Unix*): File integrity and host-based intrusion detection system, with kernel modules detection, and centralized monitoring to databases.

Resources:

FPort (*Windows NT - 2000 - XP*) [*Closed source*]: Utility who reports all open TCP and UDP ports and maps them to the owning application.

Vision (*Windows NT - 2000 - XP*) [*Closed source*]: Utility to view resources attached to a process. For example, it can detect which process owns a particular port.

Passwords:

ASMcrack (*Windows*): Unix passwords cracker under Windows, really fast.

Crack (*Unix*): Famous Unix passwords cracker.

John The Ripper (*Unix*, *Windows*, *DOS*): Famous Unix passwords cracker.

L0pht Crack (*Windows*) [*Closed source*, *Non-free*]: Famous Windows passwords cracker. It also permits to get sources from different technics: sniffing, registry, files, ...

Opie (*Unix*): One-Time passwords system.

Patchs - protections:

GRSecurity (*Linux*): Set of patchs who inserts protections at lofs of level, to enforce the security of a Linux system.

IMsafe (*Linux*): HIDS based on some artificial intelligence mechanisms, to determine system calls realized by a process who are differents from its normal behavior.

Ksec (*BSD*): Tool who permits to react against host oriented attacks, by analyzing the inconsistencies in the kernel's structure (modules, syscalls, interfaces,...).

Kstat (*Linux*): Tool who permits to react against host oriented attacks, by analyzing the inconsistencies in the kernel's structure (modules, syscalls, interfaces,...).

LIDS (*Linux*): Kernel patch for Linux, who permits to enforce a system's security and to limit root's possibilities.

SubDomain (*Linux*): Kernel patch for Linux who limits gains of privileges.

Buffer overflows - Format strings:

BOwall (*Windows NT*): Protection tool against buffer overflows under Windows, who patch critical DLLs.

FormatGuard (*Linux*): Add protections into GCC against some format strings attacks.

LibSafe (*Linux*): Library intercepting functions calls to some library functions known as vulnerables.

OpenWall (*Linux*): Kernel patch for Linux who gives better security for the system: non executable stack, modified libc's base address, FIFOs and hard links protection, restricted access to /proc, handlers protections, limitations of the number of processes, shared memory segments destruction, IP aliases protections, ...

PaX (*Linux*): Kernel patch who permits to parametrize memory pages as non executables.

RSX (*Linux*): Patch who permits to remap ELF files, and avoid to run executable code in data segments.

StackGuard (*Linux*): Add protections to GCC against some stack smashing attacks, by using non modifiable values.

StackShield (*Linux*): Add protections to GCC against some stack smashing attacks, by saving the return address.

Rootkits:

Adore (*Linux*): Famous rootkit offering lots of hiding possibilities.

ChkRootkit (*Unix*): Tool to detect lofs of famous rootkits.

Scanners:

TARA (*Unix*): Analyze the security of a Unix system. Based on Tiger.

Tiger (*Unix*): Analyze the security of a Unix system, by observing the architecture, access rights, ...

Network security:

Backdoors:

Cd00r (*Unix*): Permit to start listening on a port after receiving a specific sequence of IP paquets on a specified interface.

iCmd (*Windows NT - 2000*) [*Closed source*]: Multiconnections telnet server, extremely tiny.

OpenSSH-Reverse (*Unix*): Patched version of OpenSSH where the server connects to the client.

Cracking:

Brutus (*Windows*) [*Closed source*]: One of the best tools for remote password cracking, who permits to crack passwords for HTTP, POP3, FTP, NetBios, Telnet protocols, and who support SOCKS proxies.

ObiWaN (*Unix*, *Windows*): HTTP servers brute forcing, with intermediate proxies support,...

Encryption:

FreeSWAN (*Linux*): IPSec implementation.

STunnel (*Unix*, *Windows*): Permits to encapsulate arbitrary TCP connections in a SSL tunnel.

VTun (*Unix*): Permits to create virtual tunnels by using lots of protocols, like Ethernet, PPP, SLIP, IP, TCP, UDP, and Unix pipes.

Yavipin (*Linux*): Permits to create secure virtual tunnels.

SSH:

OpenSSH (*Unix*): Famous secure protocol.

OpenSSH-Reverse (*Unix*): Patched version of OpenSSH where the server connects to the client.

PuTTY (*Windows*): Freeware Telnet, SSH and SCP client.

WinSCP (*Windows*): Graphical SCP client, who offers two different interfaces, really friendly.

Filtering:

GFCC (*Linux*): Graphical interface who permits to create and control rules from the Linux packets filter.

IPChains (*Linux*): Paquets filter for 2.2 kernels.

IPfilter (*FreeBSD*): Paquets filter.

IPTables (*Linux*): Paquets filter for 2.4 kernels. Permit stateless/statefull packet filtering, NAT and packet mangling (user mode packets manipulations).

Packet2SQL (*Linux*): Utility who permits to convert IPChains logs to SQL insertions in a database, to realize analysis and correlations.

PKTfilter (*Windows 2000 - XP*): Configuration tool for Windows 2000 filtering, with a syntax similar to IPfilter.

Sinus Firewall (*Linux*): Linux packets filter with dynamic filtering.

Honeypots:

Deception Toolkit (*Unix*): Tool who permits to simulate vulnerable services, and to observe the manipulations that hackers realize.

LaBrea (*Unix*, *Windows NT*): Tool who permits to simulate hosts by using non used IP adresses on a network, to generate decoys to scanners.

Libraries:

LibCurl (*Unix*, *Windows*): Library who permits to easily download and upload files by using different protocols: FTP, HTTP, HTTPS, Telnet, LDAP,... Also supports proxies, cookies, authentification, resumes, and lots of languages: C, C++, Perl,...

C:

Hijacking Suite (*Unix*): Library and tools who permits to easily create hijacking attacks, for example against IRC and HTTP protocols.

LibNet (*Unix*, *Windows NT - 2000 - XP*): Functions library who permits to send arbitrary packets at different levels: Link Layer, ARP, RARP, IP, TCP, UDP... This library is used by lots of tools available on this page.

LibNIDS (*Unix*, *Windows*): Emulate Linux 2.0.x TCP/IP stack, to give an environment to analyze TCP/IP packets.

LibPCap (*Unix*, *Windows 9X - NT - 2000*): Functions library who permits to capture paquets, by supporting Berkeley Packet filter. This library is used by lots of tools available on this page.

LibRNet (*Unix*): Library who permits to build and send arbitrary packets, alternative to LibNet.

State Threads (*Unix*): Offers a threading API for structuring an Internet application as a state machine.

TCP/IP Library (*Windows 2000 - XP*): Library who permits to generate arbitrary TCP/IP packets.

Perl:

NetPacket (*Unix*): Perl library who offerts an interface to capture common TCP/IP protocols.

NetPCap (*Unix*): Perl library who permits to capture packets through LibPCap.

NetPCapUtil (*Unix*): Perl library who offerts a really simple interface to LibPCap.

NetRawIP (*Unix*): Perl library who permits to send and receive arbitrary packets.

NIDS:

Despoof (*Unix*): Command line utility who permits to detect spoofing, by analyzing packets TTL.

IDSWakeUp (*Unix*): Scripts who permits to generate lots of attacks against NIDS.

IPlog (*Unix*): TCP/IP traffic logger who detects lots of scans and attacks.

ISIC (*Unix*): Random traffic generator, by specifying percentage of particularities (fragmentation, options...). Usefull to realize tests against firewalls, NIDS, fingerprinting...

LibNIDS (*Unix*, *Windows*): Emulate Linux 2.0.x TCP/IP stack, to give an environment to analyze TCP/IP packets.

Network Flight Recorder (*Unix*) [*Closed source*, *Non-free*]: Famous NIDS, using filters descriptions in a proprietary langage.

Prelude (*Unix*): NIDS easily parametrizable to run on different hosts, by hosting NIDS sensors or a server generating reports.

Shadow (*Unix*): NIDS with Perl modules, who uses a Web interface.

Snort (*Unix*, *Windows*): Famous NIDS, fully personnalisable, and who runs using rules. Different plugins, scripts and tools permits to add more and more functionalities.

Promiscuous detection:

Anti Anti Sniffer (*Linux*): Kernel patch to avoid detection of a sniffer by classical tools.

AntiSniff (*Unix*, *Windows NT - 2000 - XP*) [*Closed source*, *Non-free*]: Utility who permits to detect network card in promiscuous mode on an Ethernet, by differents technics.

Neped (*Unix*): Utility who permits to detect network card in promiscuous mode on an Ethernet.

Sentinel (*Unix*): Utility who permits to detect network card in promiscuous mode on an Ethernet, by 3 different technics.

Scan detectors:

PortSentry (*Unix*): Scan detector.

ScanLogD (*Unix*, *Windows*): TCP scan detector.

Protocols:

LibNet (*Unix*, *Windows NT - 2000 - XP*): Functions library who permits to send arbitrary packets at different levels: Link Layer, ARP, RARP, IP, TCP, UDP... This library is used by lots of tools available on this page.

LibRNet (*Unix*): Library who permits to build and send arbitrary packets, alternative to LibNet.

NetRawIP (*Unix*): Perl library who permits to send and receive arbitrary packets.

Bouncing - Redirectors:

Bouncer (*Unix*, *Windows*): Tunnel connections through an SSL proxy, with a lot of filtering possibilities, and SOCKS server simulation.

FPipe (*Windows*) [*Closed source*]: TCP/UDP redirector, who permits to specify the source port to use.

GMAclick (*Unix*, *Windows*): Permits to download specified files from the web by using Wingates.

ProxyHunter (*Windows*) [*Closed source*]: HTTP and SOCKS proxies scanner. Permit to test them, and to be used itself as a proxy server who use a different proxy for each request.

Pudding (*Unix*): HTTP proxy who modifies HTTP requests to use anti-IDS technics.

RInetD (*Unix*, *Windows*): Permits to create TCP redirections by using precise rules.

SocksChain (*Windows*) [*Closed source*, *Non-free*]: Utility who permits to easily chain a set of SOCKS or HTTP proxies.

SocksHTTP (*Java*) [*Closed source*]: Converts classical SOCKS client requests to a request to an HTTP proxy.

HTTP:

Corkscrew (*Unix*, *Windows*): Tunnel SSH connections through an HTTP proxy.

Curl (*Unix*, *Windows*): Utility who permits to easily download and upload files by using different protocols: FTP, HTTP, HTTPS, Telnet, LDAP, ... Also supports proxies, cookies, authentification, resumes, ...

DesProxy (*Unix*, *Windows*): Tunnel TCP connections through an HTTP proxy, eventually by converting SOCKS requests.

FizzBounce (*Unix*): TCP redirector through HTTP proxies.

HTTPort (*Windows*) [*Closed source*]: Tunnel TCP connections through the HTTP protocol, by simulating a SOCKS server, and by eventually using an intermediate server.

HTTPTunnel (*Unix*, *Windows*): Bidirectionnal tunnel through HTTP requests, eventually through an HTTP proxy.

LibCurl (*Unix*, *Windows*): Library who permits to easily download and upload files by using different protocols: FTP, HTTP, HTTPS, Telnet, LDAP, ... Also supports proxies, cookies, authentification, resumes, and lots of languages: C, C++, Perl...

MultiProxy (*Windows*) [*Closed source*]: HTTP proxies tester. MultiProxy can be used as a proxy server who use a different proxy for each request.

Numby (*Unix*): Scanner for HTTP vulnerables proxies.

Proxomitron (*Windows*) [*Closed source*]: Scanner and redirector through HTTP proxies, who can also delete or modify informations contained in HTML transferred pages. For example, this permits to easily filter automatic popups, DHTML or JavaScript.

ProxyTools (*Unix, Windows*): Set of Perl utilities, who permits to use, sort, test and search for HTTP proxies.

TransConnect (*Unix*): Transparently tunnel TCP connections through an HTTP proxy.

Zylyx (*Unix*): permits to access to files through HTTP proxy caches.

IRC:

BNC (*Unix*): Simple IRC redirector.

EZBounce (*Unix*): IRC redirector with powerfull administration, DCC proxying, connections detaching, VHosts support, logging...

Muh (*Unix*): IRC redirector with connections detaching...

PsyBNC (*Unix*): IRC redirector with powerfull administration, DCC proxying, connections detaching, VHosts support, proxys support, logging, party line, scripting...

TIAtunnel (*Linux*): IRC tunnel who permits to connect from an IPv4 to an IPv4, IPv6, or IPv4 with SSL support server.

Ethernet:

ARP0c (*Linux, Windows*): TCP hijacking using ARP spoofing and bridging.

ARPing (*Unix*): Tool who permits to realize pings using ARP requests.

ARPtool (*Unix*): Arbitrary ARP paquets creation.

Fake (*Linux*): Tool who permits to realize IP take-over by using an interface and ARP spoofing.

Ghost Port Scan (*Linux, BSD*): Scanner who permits distribution and spoofing by using ARP poisoning.

GrabItAll (*Windows 2000 - XP*): Performs traffic redirection by sending spoofed ARP replies.

Hunt (*Unix*): Tool who permits to realize lots of operations on an Ethernet: sniffing, hijacking, desynchronisations, ARP relaying... Hunt can also work despite eventual switchs.

Smit (*Unix*): ARP hijacking tool, who gives some technichs to sniff despite of eventual switchs.

Hijacking:

ARP0c (*Linux*, *Windows*): TCP hijacking using ARP spoofing and bridging.

ButtSniff (*Windows*) [*Closed source*]: Little sniffer, who runs on a Telnet compatible interface, and who permits to reset connections and to hijack.

DSniff (*Unix*, *Windows*): Set of tools who permits to sniff lots of protocols, like files and mail sending, SSH and HTTPS protocols in special conditions. DSniff also offer technics to capture network traffic if switchs are used.

Ettercap (*Unix*): Console sniffer, whith a really nice interface based on NCurses, who permits to write pluggins, and offer technics against switch, to realize passive fingerprinting, and some hijacking possibilities.

Fake (*Linux*): Tool who permits to realize IP take-over by using an interface and ARP spoofing.

GrabItAll (*Windows 2000 - XP*): Performs traffic redirection by sending spoofed ARP replies.

Hijacking Suite (*Unix*): Library and tools who permits to easily create hijacking attacks, for example against IRC and HTTP protocols.

Hunt (*Unix*): Tool who permits to realize lots of operations on an Ethernet: sniffing, hijacking, desynchronisations, ARP relaying, ... Hunt can also work despite eventual switchs.

IRChijack (*Unix*, *Windows*): IRC session hijacker.

NetSed (*Unix*): Tool who permits to intercept and modify packets.

Smit (*Unix*): ARP hijacking tool, who gives some technichs to sniff despite of eventual switchs.

HTTP:

Achilles (*Windows*) [*Closed source*]: Permit to capture and modify HTTP and SSL requests, by installing it as a proxy between the client and the server.

Proxomitron (*Windows*) [*Closed source*]: Scanner and redirector through HTTP proxies, who can also delete or modify informations contained in HTML transferred pages. For example, this permits to easily filter automatic popups, DHTML or JavaScript.

Pudding (*Unix*): HTTP proxy who modifies HTTP requests to use anti-IDS technics.

TCP-IP:

APsend (*Unix*): TCP/IP arbitrary packets generator, who contains lofs of ready attacks.

FragRoute (*Unix*): Tool to use like a router, but who permits to fragment packets with different technics.

HPing (*Unix*): Command line utility who permits to generate arbitrary packets, to scan, fragment, fingerprint, ... Also permit to realize IP ID scan.

IDSWakeUp (*Unix*): Scripts who permits to generate lots of attacks against NIDS.

IPlayer (*Unix*): Permits to capture and regenerate packets captured using TCPDump's format, through NASL script or through SendIP.

Iris (*Windows*) [*Closed source, Non-free*]: Probably one of the best existing sniffers, who permits to fully rebuild sessions (HTTP, telnet, POP3, ...) and to print them in different formats, to rebuild packets, ...

ISIC (*Unix*): Random traffic generator, by specifying percentage of particularities (fragmentation, options, ...). Usefull to realize tests against firewalls, NIDS, fingerprinting, ...

MPac (*Unix*): Utility who permits to generate TCP/IP arbitrary packets by using configuration files.

Nemesis (*Unix*): Utility who permits to generate TCP/IP arbitrary packets by using scripts.

NetDude (*Unix*): Graphical interface who permits to interprete and modify TCPDump logs, and to powerfully filter.

NetSed (*Unix*): Tool who permits to intercept and modify packets.

PSH (*Solaris*): Shell who permits to generate arbitrary TCP/IP packets by using Tcl/Tk.

Rain (*Unix*): Permits to easily generate TCP, UDP, ICMP and IGMP packets on the command line, and to simulate well-known attacks.

SendIP (*Unix*): Command line utility who permits to generate arbitrary TCP/IP packets with payload.

TCP/IP Library (*Windows 2000 - XP*): Library who permits to generate arbitrary TCP/IP packets.

TCPreplay (*Unix*): Tool who permits to regenerate network traffic artificially, by using logs generated by TCPDump.

WInject (*Windows 9X - 2000*): Tool who permits to generate TCP/IP arbitrary packets, by using a graphical interface.

DNS:

Snoof (*Unix*): DNS spoofer.

Zodiac (*Unix*): Full GNU tool who permits to manipulate DNS: sniffing, decoding, flooding, spoofing, ...

FTP:

Curl (*Unix*, *Windows*): Utility who permits to easily download and upload files by using different protocols: FTP, HTTP, HTTPS, Telnet, LDAP, ... Also supports proxies, cookies, authentification, resumes, ...

FTPbounceCommander (*Unix*): Utility who permits to send files by using the FTP BOUNCE vulnerability. attacks, CGI scanning, ...

LibCurl (*Unix*, *Windows*): Library who permits to easily download and upload files by using different protocols: FTP, HTTP, HTTPS, Telnet, LDAP, ... Also supports proxies, cookies, authentification, resumes, and lots of languages: C, C++, Perl, ...

HTTP:

BabelWeb (*Unix*): Scanner who permits to obtain lots of informations about a Web server: allowed pages, available CGIs, existing files types, ...

Curl (*Unix*, *Windows*): Utility who permits to easily download and upload files by using different protocols: FTP, HTTP, HTTPS, Telnet, LDAP, ... Also supports proxies, cookies, authentification, resumes, ...

ELZA (*Unix*, *Windows*): Scripting langage who permits to realize lots of manipulationes related to websites, as: simulating browsers, automatic answer to Forms, dictionary attacks, CGI scanning, ...

GMAclick (*Unix*, *Windows*): Permits to download specified files from the web by using Wingates.

HTTPush (*Unix*): Tool who permits to easily generate HTTP requests.

LibCurl (*Unix*, *Windows*): Library who permits to easily download and upload files by using different protocols: FTP, HTTP, HTTPS, Telnet, LDAP, ... Also supports proxies, cookies, authentification, resumes, and lots of languages: C, C++, Perl, ...

Pudding (*Unix*): HTTP proxy who modifies HTTP requests to use anti-IDS technics.

Zylyx (*Unix*): permits to access to files through HTTP proxy caches.

ICMP:

AICMPsend (*Unix*): ICMP arbitrary packets generator, who contains lots of ready attacks.

ICMPEnum (*Unix*): Utility who permits to use DDoS technics to generate and receive packets (sending and receiving on different hosts).

ICMPush (*Unix*): Utility who permits to generate ICMP arbitrary packets, and to analyze received answers, to realize ICMP fingerprinting.

Sing (*Unix*): Command line utility who permits to generate arbitrary ICMP packets.

Mail:

Anubis (*Unix*): Tool who permits to send anonymous mails, with support for anonymous remailers, SOCKS, Wingates, ...

SNMP:

NetSNMP (*Unix*): Powerfull toolset, who permits to manipulate SNMP protocol.

Sockets:

NetCat (*Unix*, *Windows*): Famous tool who permits to manipulate TCP and UDP to realizate lots of operations: server, client, scanning, scripting, ...

Socket Workbench (*Windows*) [*Closed source*, *Non-free*]: Nice GUI tool to manipulate TCP sockets.

TCP:

Phoenix (*Linux*): TCP connections reseting by IP and MAC spoofing.

Reverb (*Unix*): Permits to adapt TCP connections, to allow 2 "client" connections or 2 "server" connections to communicate.

Tunnelling:

Bouncer (*Unix*, *Windows*): Tunnel connections through an SSL proxy, with a lot of filtering possibilities, and SOCKS server simulation.

Corkscrew (*Unix*, *Windows*): Tunnel SSH connections through an HTTP proxy.

CovertTCP (*Unix*): Utility who permits to encapsulate data by using directly TCP/IP headers.

HTTPort (*Windows*) [*Closed source*]: Tunnel TCP connections through the HTTP protocol, by simulating a SOCKS server, and by eventually using an intermediate server.

HTTPTunnel (*Unix*, *Windows*): Bidirectionnal tunnel through HTTP requests, eventually through an HTTP proxy.

ICMPTunnel (*Unix*): Utility who permits to encapsulate IP traffic in a ICMP tunnel by choosing the ICMP type.

ICMPXFer (*Unix*): Utility who permits to send files through ICMP.

ITunnel (*Unix*): ICMP tunnelling.

MailTunnel (*Unix*): Tunnel who permits to work by exchanging mails. Really slow, this system is a powerfull solution for users behind a heavy restrictive firewall.

STunnel (*Unix*, *Windows*): Permits to encapsulate arbitrary TCP connections in a SSL tunnel.

VPN:

FreeSWAN (*Linux*): IPSec implementation.

VTun (*Unix*): Permits to create virtual tunnels by using lots of protocols, like Ethernet, PPP, SLIP, IP, TCP, UDP, and Unix pipes.

Yavipin (*Linux*): Permits to create secure virtual tunnels.

Scanning:

NDif (*Unix*): Filtering tool for NMap logs, who permits principally to observe differences between 2 scans, and generate an HTML report.

RNMap (*Unix*): Python scripts who permits to realize distributed scans from hosts using NMap.

Fingerprinting:

FTPmap (*Unix*): FTP servers fingerprinting, by analyzing commands from the server.

LDistFP (*Unix*): Fingerprinting based on Ident requests, specific for each system.

MingSweeper (*Windows 2000 - XP*): Wonderfull scanner with a nice GUI interface, who offerts lots of possibilities: scanning, fingerprinting, banners grabbing, ...

NMap (*Unix*, *Windows NT - 2000 - XP*): Famous scanner who offerts lots of technics and options: domains, ports, fingerprinting, ...

Queso (*Unix*): Famous OS fingerprinting tool, fully parametrizable.

Retina (*Windows NT - 2000 - XP*) [*Closed source*, *Non-free*]: Powerfull vulnerability scanner, who permits fingerprinting, adding modules, ...

TelnetFP (*Unix*): Fingerprinting based on Telnet negociations, specific for each system.

WinFingerprint (*Windows*): Windows hosts scanning and fingerprinting, based on SMB requests.

XProbe (*Unix*): Utility implementing a full fingerprinting logic, based on ICMP, really usefull to differenciate Windows systems.

Passive:

Archaeopteryx (*Windows NT - 2000 - XP*) [*Closed source*]: Passive OS fingerprinting tool, based on Siphon. Has a great GUI.

P0F (*Unix*): Passive OS fingerprinting.

PSting (*Unix*): Passive OS fingerprinting, by analyzing ICMP Echo.

Syphon (*Unix*, *Windows*): Famous tool for passive OS fingerprinting.

Patchs:

Fingerprint Fucker (*Linux*): Kernel patch against NMap fingerprintings.

IPpersonality (*Linux*): Kernel patch for Linux 2.4, who permits to modify some TCP/IP stack parameters, to avoid fingerprinting technics.

Stealth (*Linux*): Kernel patch for Linux who permits to fake informations returned by the TCP/IP stack during OS fingerprintings.

Mapping:

Cheops (*Linux*): Graphical networking analysis tool, who permits to map networks, or to realize fingerprinting or scanning.

Ghost Port Scan (*Linux*, *BSD*): Scanner who permits distribution and spoofing by using ARP poisoning.

MingSweeper (*Windows 2000 - XP*): Wonderfull scanner with a nice GUI interface, who offerts lots of possibilities: scanning, fingerprinting, banners grabbing, ...

NMap (*Unix*, *Windows NT - 2000 - XP*): Famous scanner who offerts lots of technics and options: domains, ports, fingerprinting...

Nomad (*Unix*): SNMP scanner, who permits to map a local network.

ACL:

FilterRules (*Unix*): Server and client exchanging IP packets through a firewall, to detect the actual filtering rules.

Firewalk (*Unix*): Tool using Traceroute like technics, to detect ACL to access networks and map those networks.

FTester (*Unix*): Perl scripts (client and server) exchanging personalized packets through a firewall, to detect the actual filtering rules.

Ports:

LameScan (*Unix*): Multi-thread scanner who permits SYN scans, FIN scans, XMAS scans, ... and fragmentation.

MingSweeper (*Windows 2000 - XP*): Wonderfull scanner with a nice GUI interface, who offerts lots of possibilities: scanning, fingerprinting, banners grabbing, ...

NMap (*Unix*, *Windows NT - 2000 - XP*): Famous scanner who offerts lots of technics and options: domains, ports, fingerprinting...

WinFingerprint (*Windows*): Windows hosts scanning and fingerprinting, based on SMB requests.

Sockets:

SuperScan (*Windows*) [*Closed source*]: Powerfull domains scanner, multi-threads, and with a nice interface.

Banners:

ExScan (*Unix*): Domain scanner who permits to get banners.

FScan (*Windows*) [*Closed source*]: Command line TCP-UDP domain scanner, who permits to get banners.

Grabbb (*Unix*): Banners scanner.

MingSweeper (*Windows 2000 - XP*): Wonderfull scanner with a nice GUI interface, who offerts lots of possibilities: scanning, fingerprinting, banners grabbing, ...

Protocols:

Nat (*Unix*): Powerfull NetBios scanner, who permits to realizate brute forcing.

Nomad (*Unix*): SNMP scanner, who permits to map a local network.

RelayTest (*Unix*): Scanner who permits to detect if a SNMP server is an open relay.

ScanSSH (*Unix*): Permits to scan addresses lists, to find SSH servers, and known their version.

FTP:

Grim's Ping (*Windows*): Scanner who permits to find public FTP servers.

Proxies:

CUM Proxy Toolkit (*Unix*): Set of tools to check HTTP vulnerables proxies.

MultiProxy (*Windows*) [*Closed source*]: HTTP proxies tester. MultiProxy can be used as a proxy server who use a different proxy for each request.

Numby (*Unix*): Scanner for HTTP vulnerables proxies.

Proxomitron (*Windows*) [*Closed source*]: Scanner and redirector through HTTP proxies, who can also delete or modify informations contained in HTML transferred pages. For example, this permits to easily filter automatic popups, DHTML or JavaScript.

ProxyHunter (*Windows*) [*Closed source*]: HTTP and SOCKS proxies scanner. Permit to test them, and to be used itself as a proxy server who use a different proxy for each request.

ProxyTools (*Unix*, *Windows*): Set of Perl utilities, who permits to use, sort, test and search for HTTP proxies.

SocksChain (*Windows*) [*Closed source*, *Non-free*]: Utility who permits to easily chain a set of SOCKS or HTTP proxies.

Vulnerabilities:

Cerberus Internet Scanner (*Windows NT - 2000*) [*Closed source*]: Vulnerability scanner.

Messala (*Unix*): Little vulnerability scanner.

MNS (*Unix*): Domain scanner searching for vulnerabilities, like NMap and SScan.

MScan (*Unix*): Domain scanner searching for vulnerabilities.

Nessus (*Unix*, *Windows*): Famous vulnerability scanner, who offerts a server and different clients (GTK, Win32, Java, ...).

Retina (*Windows NT - 2000 - XP*) [*Closed source*, *Non-free*]: Powerfull vulnerability scanner, who permits fingerprinting, adding modules, ...

Saint (*Unix*): Vulnerability scanner based on Satan.

Sara (*Unix*): Vulnerability scanner based on Satan, but who can use others tools, like NMap.

Satan (*Unix*): Famous vulnerability scanner, but a little old.

Shadow Security Scanner (*Windows*) [*Closed source*, *Non-free*]: Very good vulnerability scanner.

SScan (*Unix*): Domain scanner who searchs for vulnerabilities, based on MScan. It supports worm integration, and a scripting langage.

SScan2K (*Unix*): Domain scanner searching for vulnerabilities, based on SScan, and who can use NMap and Wingates proxies.

HTTP:

BabelWeb (*Unix*): Scanner who permits to obtain lots of informations about a Web server: allowed pages, available CGIs, existing files types, ...

Stealth (*Windows*): Famous Web vulnerability scanner.

Whisker (*Unix*): Famous CGI vulnerability scanner who contains lots of technics against NIDS.

Sniffing:

Anti Anti Sniffer (*Linux*): Kernel patch to avoid detection of a sniffer by classical tools.

ButtSniff (*Windows*) [*Closed source*]: Little sniffer, who runs on a Telnet compatible interface, and who permits to reset connections and to hijack.

Ethereal (*Unix*, *Windows*): Graphical sniffer who knows a lot of protocols, and gives detailed informations on the different records of the headers. It also permits to rebuild TCP sessions. A text version is also furnished, who permits to filter using a incredible set of protocols records.

Hunt (*Unix*): Tool who permits to realize lots of operations on an Ethernet: sniffing, hijacking, desynchronisations, ARP relaying, ... Hunt can also work despite eventual switchs.

IPgrab (*Unix*): Sniffer who prints detailed informations about each used header.

Iris (*Windows*) [*Closed source*, *Non-free*]: Probably one of the best existing sniffers, who permits to fully rebuild sessions (HTTP, telnet, POP3, ...) and to print them in different formats, to rebuild packets, ...

LibPCap (*Unix*, *Windows 9X - NT - 2000*): Functions library who permits to capture paquets, by supporting Berkeley Packet filter. This library is used by lots of tools available on this page.

NetPacket (*Unix*): Perl library who offerts an interface to capture common TCP/IP protocols.

NetPCap (*Unix*): Perl library who permits to capture packets through LibPCap.

NetPCapUtil (*Unix*): Perl library who offerts a really simple interface to LibPCap.

NetRawIP (*Unix*): Perl library who permits to send and receive arbitrary packets.

NGrep (*Unix*, *Windows*): Packets analyzer, who uses regular expressions as the grep Unix tool.

Smit (*Unix*): ARP hijacking tool, who gives some technichs to sniff despite of eventual switchs.

SniffIt (*Unix*, *Windows NT - 2000*): Sniffer who permits to obtain detailed informations in lots of formats. Know ICMP, TCP and UDP protocols.

TCPdump (*Unix*, *Windows*): The most famous sniffer.

TCPflow (*Unix*): Permits to record TCP session with traffic recomposition.

TCPreplay (*Unix*): Tool who permits to regenerate network traffic artificially, by using logs generated by TCPDump.

TheWESP (*Unix*): Sniffer who captures packets directly by using a filter in a MySQL database.

Analyzers:

IPlayer (*Unix*): Permits to capture and regenerate packets captured using TCPDump's format, through NASL script or through SendIP.

NetDude (*Unix*): Graphical interface who permits to interprete and modify TCPDump logs, and to powerfully filter.

NStreams (*Unix*): Analyze TCP streams on a network, and determine if they are part of well-known protocols and ports.

Protocols:

DSniff (*Unix*, *Windows*): Set of tools who permits to sniff lots of protocols, like files and mail sending, SSH and HTTPS protocols in special conditions. DSniff also offer technics to capture network traffic if switchs are used.

Ettercap (*Unix*): Console sniffer, whith a really nice interface based on NCurses, who permits to write pluggins, and offer technics against switch, to realize passive fingerprinting, and some hijacking possibilities.

Phoss (*Unix*): Sniffer who permits to automatically capture passwords for those protocols: HTTP, FTP, LDAP, Telnet, IMAP4 and POP3.

SNMPsniff (*Unix*): Tool who permits to sniff SNMP packets.

SSLdump (*Unix*, *Windows*): Permits to observe SSL/TLS traffic. If keys are given, a full decryption is possible.

Files:

FTPXerox (*Windows NT - 2000 - XP*) [*Closed source*]: This tool permits to capture files from FTP transfers.

Owns (*Linux*, *Windows*): Sniffer who permits to capture files exchanged by HTTP, POP3 and NNTP protocols.

SMBsniff (*Unix*): Sniffer who permits to capture files exchanged using NetBios protocol.

Wireless:

AirSnort (*Linux*): Tool to sniff and break 802.11 WEP secret keys when enough packets have been captured.

WEPCrack (*Unix*): Tool to break 802.11 WEP secret keys.

Sockets:

Proxy Workbench (*Windows*) [*Closed source*, *Non-free*]: Debugging sockets tool, who works like a proxy.

TracePlus Winsock (*Windows*) [*Closed source*, *Non-free*]: Debugging sockets tool, who permits to observe all calls to Winsock APIs, and exchanged data.

ANEXO-III

Diccionario-Glosario-Léxico

- Si alguien no conocía el "Hacking Lexicon" a partir de ahora no será el mismo: http://www.robertgraham.com/pubs/hacking-dict.html
- y este otro del SANS: http://www.sans.org/resources/glossary.php