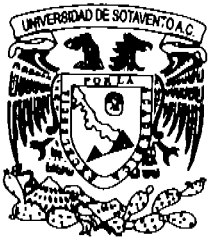


Universidad de Sotavento A.C.

ESTUDIOS INCORPORADOS A LA UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INFORMATICA

"LA IMPORTANCIA DE LA SEGURIDAD EN LOS MEDIOS DIGITALES"

TESIS PROFESIONAL

QUE PARA OBTENER EL TÍTULO DE

LICENCIADA EN INFORMÁTICA

PRESENTA:

Gabriela Felix Vazquez

ASESOR DE TESIS:

LIC. RAUL DE JESÚS OCAMPO COLÍN

Coatzacoalcos, Veracruz, Abril de 2008.

AGRADEZCO...

...A Dios
...A mis Amados Padres
...A mi Familia
...A mis Catedraticos
...A mi Asesor de Tesis
...A mis amigos

A todos ustedes.... Muchisisisismas gracias...

Índice de contenido

Introducción.

Existe una constante preocupación por parte de los investigadores que trabajan en desarrollos tecnológicos resultado de sus proyectos, en lo referente a la seguridad y confiabilidad de los mismos. Se considera necesario que cuenten con lineamientos y herramientas de apoyo para desminuir este problema y así poder liberar los resultados de sus trabajos sin ninguna preocupación.

El objetivo de este proyecto consiste en establecer un conjunto de lineamientos necesarios para implementar Seguridad en aplicaciones basadas en servicios digitales en internet para lograr su Confiabilidad y Calidad. Estos lineamientos estarán acompañados por las herramientas correspondientes para su implementación. Es de suma importancia el que los investigadores del Centro de Computo cuenten con lineamientos y herramientas de apoyo a la seguridad de los desarrollos tecnológicos resultado de sus proyectos de investigación para lograr un trabajo confiable y de calidad.

El objetivo general de este trabajo de tesis consiste en presentar los puntos más importantes de la seguridad informática aplicada a las necesidades de los investigadores del Centro de Computo que desarrollan aplicaciones WEB a intalarse en el servidor del Centro de Computo, ya que es algo ha considerar en cualquier red. Como sabemos por cualquier red circulan diariamente todo tipos de datos, entre ellos muchos se podrlan catalogar confidenciales (nóminas, expedientes, presupuestos..) o al menos como privados (correo electrónico, proyectos de investigación, articulos a punto de ser publicados..). Estos es independientemente de la etiqueta que cada usuario de la red quiera colgarle a sus datos, parece claro que el fallo de seguridad de la red no beneficia a nadie y muchos menos a la imagen de la organización

¿Qué es Seguridad?

La seguridad de software aplica los principios de la seguridad de información al desarrollo de software. Esto se refiere a la seguridad de información comúnmente conocida como la protección de sistemas de información contra el acceso desautorizado y la modificación de información, sí está en una fase de procesamiento, almacenamiento o tránsito.

La seguridad de software también protege contra la negación de servicios a usuarios desautorizados y la provisión de servicios a usuarios desautorizados, incluyendo las medidas necesarias para detectar, documentar y contraatacar tales amenazas.

Existen preguntas referentes a la seguridad, muchas de estas relacionadas con el ciclo de vida del software. Particularmente la seguridad del código y el proceso del software, deben ser considerados durante la fase del diseño y el desarrollo. Además, la seguridad debe ser preservada durante la operación y el mantenimiento para asegurar la integridad de una parte del software.

Un tema muy importante es que la falta de seguridad se origina fundamentalmente por dos problemas: Por un lado los sistemas que son teóricamente seguros pueden ser inseguros en la práctica y, por otro lado los sistemas son cada vez mas complejos; la complejidad de los sistemas proporcionan cada vez mas oportunidades para los ataques. Podemos llegar a decir que es mucho mas fácil demostrar que un sistema es inseguro que demostrar que uno es seguro.

¿Qué es Confiabilidad?

Pasando a la confiabilidad del software, podemos observar que la confiabilidad del software significa que un programa particular debe seguir funcionando bajo la

presencia de errores. Los errores pueden ser relacionados al diseño, a la implementación, a la programación, o el uso de errores. Así como cada vez los sistemas son mas complejos aumenta la probabilidad de errores. La confiabilidad de cualquier sistema depende de la corrección de su diseño, lo correcto de la correspondencia entre éste y su aplicación, y la confiabilidad de sus componentes. Una breve definición de lo que es la confiabilidad del software, es que la confiabilidad de un sistema de software es una medida de lo bien que proporciona los servicios que los usuarios esperan de él. Una aplicación confiable del diseño del software comprende que todas las partes del diseño deben ser realmente aplicadas y que tal aplicación tiene que ser una transformación correcta de la notación del diseño a un lenguaje de programación.

¿Qué es Calidad?

Si hablamos sobre la calidad en la ingeniería de software, podemos encontrar problemas importantes a considerar. Si ponemos la atención explícita a las características a la calidad de software puede conducir a ahorros significativos en gastos de ciclo de vida de software.

La calidad de software ha recibido una cantidad significativa de atención recientemente debido al énfasis aumentado por la industria americana sobre la calidad y debido a la tentativa de controlar gastos de mantenimiento crecientes. Hasta hace poco la calidad era la responsabilidad del ingeniero de software o el programador; sin embargo, el énfasis aumentado sobre la calidad ha incitado un aumento del número de organizaciones de calidad separadas. El éxito de una organización de calidad independiente es dependiente de varios factores: el estado corriente de tecnología de calidad de software, compromiso de dirección de alto nivel, y recursos adecuados técnicos.

Para sostener las calidades de un sistema de software durante la evolución, y adaptar los atributos de calidad como las exigencias se desarrollan, es necesario tener una arquitectura de software clara que es entendida por todos los reveladores y al que todos los cambios al sistema se adhieren.

Objetivos del Proyecto.

1. Investigación en seguridad para aplicaciones basadas en Internet, para asegurar su confiabilidad y calidad.
2. Establecimiento de lineamientos de seguridad para lograr la confiabilidad y calidad del software basado en Internet.
3. Análisis detallado de herramientas de apoyo para asegurar las Aplicaciones Web del Centro de Cómputo.

Alcances y Limitaciones.

En este trabajo de tesis queremos llegar a que los lineamientos que utilizaremos para implementar Seguridad en aplicaciones basadas en servicios digitales en Internet, para que sean lo más confiables y proporcionen un servicio de calidad. Aunque sabemos que actualmente no hay ninguna metodología que asegure totalmente la Seguridad y Confiabilidad en los sistemas de software, es nuestra meta que los servicios digitales en Internet sean en toda su plenitud seguros y confiables para los investigadores que trabajan en desarrollos tecnológicos y que sea mínima su constante preocupación referente a la seguridad y confiabilidad.

El éxito de esta tesis depende de algunos puntos importantes. Uno de ellos es que necesitamos saber cuales son las necesidades de los investigadores del Centro de Computo que trabajan en desarrollos tecnológicos, ya que sin esta información no podremos implementar todos los lineamientos a seguir para lograr la Seguridad

y Confiabilidad en los servicio digitales en Internet. Otro punto importante es que si no se ha hecho una adecuada investigación sobre los temas más importantes a trabajar en esta tesis, tendremos serios problemas a la hora de establecer la metodología de seguridad.

Capitulo 1.

Seguridad Informática: Conceptos básicos.

1.1. ¿Qué es "network security" o la seguridad en la red (servicios digitales en Internet)?

Definimos la seguridad de información como la protección de ventajas de información de la revelación no autorizada, de la modificación, o de la destrucción, o accidental o intencional, o la incapacidad para procesar esa información. La seguridad de la red, se compone de esas medidas tomadas para proteger una red del acceso no autorizado, interferencia accidental o intencionada con operaciones normales, o con la destrucción, inclusive la protección de facilidades físicas, del software, y de la seguridad del personal.

La seguridad en el Web es un conjunto de procedimientos, prácticas y tecnologías para proteger a los servidores y usuarios del Web y las organizaciones que los rodean. La Seguridad es una protección contra el comportamiento inesperado **[Garfinkel, 1999]**.

1.1.1. ¿Por qué requiere atención especial la seguridad en el Web?

- Internet es una red de dos sentidos. Así como hace posible que los servidores Web divulguen información a millones de usuarios, permite a los hackers, crackers, criminales y otros "chicos malos" irrumpir en las mismas computadoras donde se ejecutan los servidores Web.
- Las empresas, instituciones y los gobiernos utilizan cada vez más el Word Wide Web para distribuir información importante y realizar transacciones comerciales. Al violar servidores Web se pueden dañar reputaciones y perder dinero.

• Aunque el Web es fácil de utilizar, los servidores son piezas de software extremadamente complicadas y tienen diversas fallas de seguridad potenciales.
• Es mucho más onerosa y tardada la recuperación de un incidente de seguridad que implementar medidas preventivas.

1.1.2. ¿Por qué preocuparse sobre la seguridad en el Web?

Los servidores son un blanco atractivo para los trangesores por varias razones **(Garfinkel, 1999)**:

Publicidad. Los servidores web son la cara que las organizaciones presentan al público y al mundo electrónico. Un ataque exitoso a alguno de ellos es acto público que puede ser visto en unas horas por cientos de miles de personas. Los ataques pueden lanzarse por razones ideológicas o financieras, o ser simples actos vandálicos cometidos al azar.
Comercio. Muchos servidores web estan ralacionados con el comercio y el dinero. De hecho los protocolos criptográficos integrados a Navigator de Netscape y otros navegadores fueron originalmete incluidos para permitir a los usuarios enviar números de tarjetas de crédito por Internet sin preocuparse de que fueran interceptados. De esta forma, los servidores web se han convertido en repositorios de información financiera confidencial, lo cual los convierte en un blanco atractivo para los atacantes.
Información confidencial. Para las organizaciones, la tecnología del Web se ha convertido en una forma de distribuir información con gran sencillez, tanto internamente, a sus propios miembros, como de manera externa, a sus socios en todo el mundo. Esta información confidencial es un blanco atractivo para sus competidores y enemigos.
Acceso a las redes. Al ser utilizados por personas tanto dentro como fuera de las organizaciones, los servidores web sirven efectivamente como puente entre la red interna de la organización y las redes externas. Su posición privilegiada en cuanto

a las conexiones de red los convierte en un blanco ideal para ser atacados, ya que un servidor web violado puede emplearse como base para atacar desde ahí a las computadoras de una organización.

Extensibilidad de los servidores. Debido a su naturaleza, los servidores estan diseñados para ser extensibles, lo cual hace posible conectarlos con bases de datos, sistemas heredados y otros programas que se ejecutan en la red de una organización. Si no se implementan de modo adecuado, los módulos que se agregan a un servidor pueden comprometer la seguridad de todo el sistema.

Interrupción del servicio. Como la tecnología del Web se basa en la familia de protocolos TCP/IP, está sujeta a interrupciones del servicio: ya sea accidental o intencionamente por medio de ataques de negación del servicio. Las personas que utilizan dicha tecnologías den estar enteradas de sus fallas y prepararse para interrupciones importantes del servicio.

Soporte complicado. Los navegadores necesitan servicios internos, como DNS (Servicio de nombres de Dominio, Domain Name Service) y el enrutamiento del protocolo IP (Protocolo Interne, Internet Protocol) para funcionar bien. La robustez y confiabilidad de tales servicios pueden ser desconocidas y vulnerables a errores de programación, accidentes y subversión. la subversión de un servicio de mas bajo nivel puede causar problemas también a los navegadores.

1.2. Objetivos de la Seguridad.

Seguridad informática es el conjunto de procedimientos, estrategias y herramientas que permitan garantizar la integridad, la disponibilidad y la confidencialidad de la información de una entidad.

Integridad. Es necesario asegurar que los datos no sufran cambios no autorizados, la pérdida de integridad puede acabar en fraudes, decisiones erróneas o como paso a otros ataques. El sistema contiene información que debe

se protegida de modificaciones imprevistas, no autorizas o accidentales, como información de censo o sistemas de transacciones financieras.

Disponibilidad. Se refiere a la continuidad operativa de la entidad, la pérdidad de disponibilidad puede implicar, la pérdida de productividad o de cridibilidad de la entidad. El sistema contiene información o proporciona servicios que deben estar disponibles a tiempo para satisfacer requisitos o evitar pérdidas importantes, como sistemas esenciales de seguridad y protección de la vida.

Confidencialidad. Se refiere a la protección de datos frente a la difusión no autorizada, la pérdida de confidencialidad puede rsultar en problemas legales, pérdida del negocio o de credibilidad. El sistema contiene información que necesita protección contra la divulgación no autorizada, como información parcial de informes, información personal o información comercial patentada. Estos aspectos además de lidiar con el riesgo que representan los atacantes remotos, se ven amenazados también por los riesgos por desastres naturales, empleados desleales, virus y sabotaje, entre otros.

1.3. Estrategias de Seguridad.

La metodología de seguridad está diseñada para ayudar a los profesionales de la seguridad a desarrollar una estrategia para proteger la *disponibilidad, integridad y confidencialidad* de los datos de los sistemas informáticos (IT) de las organizaciones. Es de interés para los administradores de recursos de información, los directores de seguridad informática y los administradores, y tiene un valor especial para todos aquellos que intentan establecer directivas de seguridad.

La metodología ofrece un acercamiento sistemático a esta importante tarea y, como precaución final, también implica el establecimiento de planes de contingencia en caso de desastre.

Los administradores de seguridad tienen que decidir el tiempo, dinero y esfuerzo que hay que invertir para desarrollar las directivas y controles de seguridad apropiados.

Cada organización debe analizar sus necesidades específicas y determinar sus requisitos y limitaciones en cuanto a recursos y programación. Cada sistema informático, entorno y directiva organizativa es distinta, lo que hace que cada servicio y cada estrategia de seguridad sean únicos. Sin embargo, los fundamentos de una buena seguridad siguen siendo los mismos y este proyecto se centra en dichos principios **(Benson, 2001)**.

a) Identificar métodos, herramientas y técnicas de ataques probables.

Las listas de amenazas, de las que disponen la mayor de las organizaciones, ayudan a los administradores de seguridad a identificar los distintos métodos, herramientas y técnicas de ataque que se pueden utilizar en los ataques. Los métodos pueden abarcar desde virus y gusanos a la adivinación de contraseñas y la interceptación del correo electrónico. Es importante que los administradores actualicen constantemente sus conocimientos en esta área, ya que los nuevos métodos, herramientas y técnicas para sortear las medidas de seguridad evolucionan de forma continua.

b) **Establecer estrategias proactivas y reactivas.**

En cada método, el plan de seguridad debe incluir una estrategia *proactiva* y otra *reactiva*. La estrategia *proactiva* o de previsión de ataques es un conjunto de pasos que ayuda a reducir al mínimo la cantidad de puntos vulnerables existentes en las directivas de seguridad y a desarrollar planes de contingencia. La determinación del daño que un ataque va a provocar en un sistema y las debilidades y puntos vulnerables explotados durante este ataque ayudará a desarrollar la estrategia proactiva.

La estrategia *reactiva* o estrategia posterior al ataque ayuda al personal de seguridad a evaluar el daño que ha causado el ataque, a repararlo o a implementar el plan de contingencia desarrollado en la estrategia proactiva, a documentar y aprender de la experiencia, y a conseguir que las funciones comerciales se normalicen lo antes posible.

c) Pruebas.

El último elemento de las estrategias de seguridad, las pruebas y el estudio de sus resultados, se lleva a cabo después de que se han puesto en marcha las estrategias reactiva y proactiva. La realización de ataques simulados en sistemas de pruebas o en laboratorios permite evaluar los lugares en los que hay puntos vulnerables y ajustar las directivas y los controles de seguridad en consecuencia. Estas pruebas no se deben llevar a cabo en los sistemas de producción real, ya que el resultado puede ser desastroso. La carencia de laboratorios y equipos de pruebas a causa de restricciones presupuestarias puede imposibilitar la realización de ataques simulados. Para asegurar los fondos necesarios para las pruebas, es importante que los directivos sean conscientes de los riesgos y consecuencias de los ataques, así como de las medidas de seguridad que se pueden adoptar para proteger al sistema, incluidos los procedimientos de las pruebas. Si es posible, se

deben probar físicamente y documentar todos los casos de ataque para determinar las mejores directivas y controles de seguridad posibles que se van a implementar.

d) Equipos de respuestas a incidentes.

Es aconsejable formar un equipo de respuesta a incidentes. Este equipo debe estar implicado en los trabajos proactivos del profesional de la seguridad. Entre éstos se incluyen:

- El desarrollo de instrucciones para controlar incidentes.
- La identificación de las herramientas de software para responder a incidentes y eventos.
- La investigación y desarrollo de otras herramientas de seguridad informática.
- La realización de actividades formativas y de motivación.
- La realización de investigaciones acerca de virus.
- La ejecución de estudios relativos a ataques al sistema.

Estos trabajos proporcionarán los conocimientos que la organización puede utilizar y la información que hay que distribuir antes y durante los incidentes. Una vez que el administrador de seguridad y el equipo de respuesta a incidentes han realizado estas funciones proactivas, el administrador debe delegar la responsabilidad del control de incidentes al equipo de respuesta a incidentes.

Esto no significa que el administrador no deba seguir implicado o formar parte del equipo, sino que no tenga que estar siempre disponible, necesariamente, y que el equipo debe ser capaz de controlar los incidentes por sí mismo. El equipo será el responsable de responder a incidentes como virus, gusanos o cualquier otro código dañino, invasión, engaños, desastres naturales y ataques del personal

interno. El equipo también debe participar en el análisis de cualquier evento inusual que pueda estar implicado en la seguridad de los equipos o de la red.

1.3.1. Metodología para la definición de una estrategia de seguridad.

La figura 1.1 explica una metodología para definir una estrategia de seguridad informática que se puede utilizar para implementar directivas y controles de seguridad con el objeto de aminorar los posibles ataques y amenazas. Los métodos se pueden utilizar en todos los tipos de ataques a sistemas, independientemente de que sean intencionados, no intencionados o desastres naturales, y, por consiguiente, se puedan volver a utilizar en distintos casos de ataque.

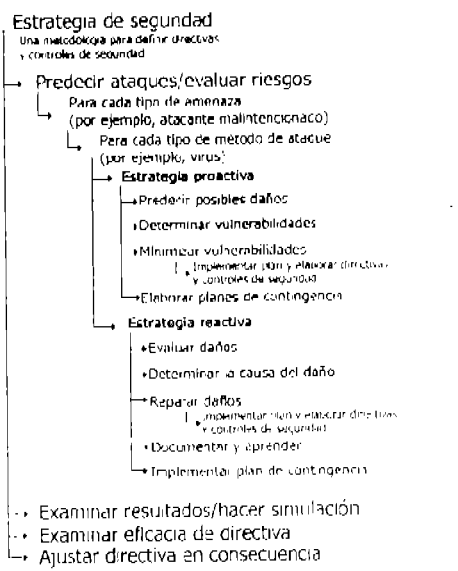


Figura 1.1 Metodología de estrategias de seguridad.

a) Predecir posibles ataques y analizar riesgos.

La primera fase de la metodología esquematizada en la figura 1.1, es determinar los ataques que se pueden esperar y las formas de defenderse contra ellos. Es imposible estar preparado contra todos los ataques; por lo tanto, hay que prepararse para los que tiene más probabilidad de sufrir la organización. Siempre es mejor prevenir o aminorar los ataques que reparar el daño que han causado.

Para mitigar los ataques es necesario conocer las distintas amenazas que ponen en peligro los sistemas, las técnicas correspondientes que se pueden utilizar para comprometer los controles de seguridad y los puntos vulnerables que existen en las directivas de seguridad. El conocimiento de estos tres elementos de los ataques ayuda a predecir su aparición e, incluso, su duración o ubicación. La predicción de los ataques trata de pronosticar su probabilidad, lo que depende del conocimiento de sus distintos aspectos. Los diferentes aspectos de un ataque se pueden mostrar en la siguiente ecuación:

Amenazas + Motivos + Herramientas y técnicas + Puntos vulnerables = Ataque

b) Para cada tipo de amenaza.

Considere todas las amenazas posibles que causan ataques en los sistemas. Entre éstas se incluyen los agresores malintencionados, las amenazas no intencionadas y los desastres naturales. La siguiente figura clasifica las distintas amenazas a los sistemas:

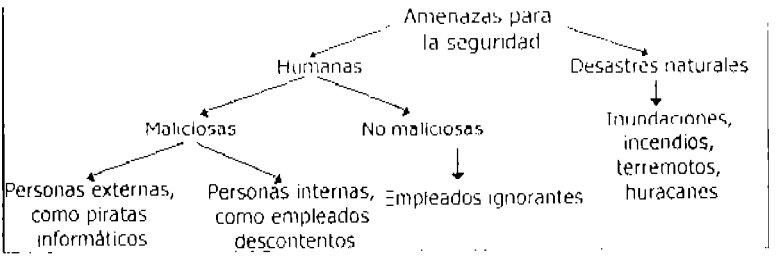


Figura 1.2. Amenazas para la Seguridad.

Amenazas como empleados ignorantes o descuidados, y los desastres naturales no implican motivos u objetivos; por lo tanto, no se utilizan métodos, herramientas o técnicas predeterminadas para iniciar los ataques. Casi todos estos ataques o infiltraciones en la seguridad se generan internamente; raras veces los va a iniciar alguien ajeno a la organización.

Para estos tipos de amenazas, el personal de seguridad necesita implementar estrategias proactivas o reactivas siguiendo las instrucciones de la figura 1.1.

c) Para cada tipo de método de ataque.

Para iniciar un ataque, se necesita un método, una herramienta o una técnica para explotar los distintos puntos vulnerables de los sistemas, de las directivas de seguridad y de los controles. Los agresores pueden utilizar varios métodos para iniciar el mismo ataque. Por lo tanto, la estrategia defensiva debe personalizarse para cada tipo de método utilizado en cada tipo de amenaza.

De nuevo, es importante que los profesionales de la seguridad estén al día en los diferentes métodos, herramientas y técnicas que utilizan los agresores. La siguiente es una lista breve de esta técnica:

- Ataques de denegación de servicio.
- Ataques de invasión.
- Ingeniería social.
- Virus.
- Gusanos.
- Caballos de Troya.
- Modificación de paquetes.
- Adivinación de contraseñas.
- Interceptación de correo electrónico.

d) Estrategia proactiva.

La estrategia proactiva es un conjunto de pasos predefinidos que deben seguirse para evitar ataques antes de que ocurran. Entre estos pasos se incluye observar cómo podría afectar o dañar el sistema, y los puntos vulnerables. Los conocimientos adquiridos en estas evaluaciones pueden ayudar a implementar las directivas de seguridad que controlarán o aminorarán los ataques.

Éstos son los tres pasos de la estrategia proactiva:

- Determinar el daño que causará el ataque.
- Establecer los puntos vulnerables y las debilidades que explotará el ataque.
- Reducir los puntos vulnerables y las debilidades que se ha determinado en el sistema para ese tipo de ataque específico.

El seguimiento de estos pasos para analizar los distintos tipos de ataques tiene una ventaja adicional: comenzará a emerger un modelo, ya que en los diferentes factores se superponen para diferentes ataques. Este modelo puede ser útil al determinar las áreas de vulnerabilidad que plantean el mayor riesgo para la

empresa. También es necesario tomar nota del costo que supone la pérdida de los datos frente al de la implementación de controles de seguridad. La ponderación de los riesgos y los costos forma parte de un análisis de riesgos del sistema que se explica en el documento técnico acerca del diseño de la seguridad.

Las directivas y controles de seguridad no serán, en ningún caso, totalmente eficaces al eliminar los ataques. Éste es el motivo por el que es necesario desarrollar planes de recuperación y de contingencia en caso de que se quebranten los controles de seguridad.

e) Determinar el daño posible que puede causar un ataque.

Los daños posibles pueden oscilar entre pequeños fallos del equipo y la pérdida, catastrófica, de los datos. El daño causado al sistema dependerá del tipo de ataque. Si es posible, utilice un entorno de prueba o de laboratorio para clarificar los daños que provocan los diferentes tipos de ataques. Ello permitirá al personal de seguridad ver el daño físico que causan los ataques experimentales. No todos los ataques causan el mismo daño.

f) Determinar los puntos vulnerables o las debilidades que pueden explotar los ataques.

Si se pueden descubrir los puntos vulnerables que explota un ataque específico, se pueden modificar las directivas y los controles de seguridad actuales o implementar otras nuevas para reducir estos puntos vulnerables. La determinación del tipo de ataque, amenaza y método facilita el descubrimiento de los puntos vulnerables existentes. Esto se puede reconocer por medio de una prueba real. Se deben determinar los puntos vulnerables o debilidades en las áreas de seguridad física, de datos y de red.

g) Reducir los puntos vulnerables y debilidades que puede explotar un posible ataque.

La reducción de los puntos vulnerables y las debilidades del sistema de seguridad que se determinaron en la evaluación anterior es el primer paso para desarrollar directivas y controles de seguridad eficaces. Ésta es la compensación de la estrategia proactiva.

Mediante la reducción de los puntos vulnerables, el personal de seguridad puede hacer disminuir tanto la probabilidad de un ataque como su eficacia, si se produce alguno.

Tenga cuidado de no implementar controles demasiado estrictos, ya que la disponibilidad de la información se convertirla en un problema. Debe haber un cuidado equilibrio entre los controles de seguridad y el acceso a la información. Los usuarios deben tener la mayor libertad posible para tener acceso a la información.

h) Elaborar planes de contingencia.

Un plan de contingencia es un plan alternativo que debe desarrollarse en caso de que algun ataque penetre en el sistema y dañe los datos o cualquier otro activo, detenga las operaciones comerciales habituales y reste productividad. El plan se sigue si el sistema no se puede restaurar a tiempo. Su objetivo final es mantener la disponibilidad, integridad y confidencialidad de los datos (es el típico "Plan B").

Debe haber un plan para cada tipo de ataque y tipo de amenaza. Cada plan consta de un conjunto de pasos que se han de emprender en el caso de que un ataque logre pasar las directivas de seguridad.

l) Estrategia reactiva.

La estrategia reactiva se implementa cuando ha fallado la estrategia proactiva y define los pasos que deben adoptarse después o durante un ataque. Ayuda a identificar el daño causado y los puntos vulnerables que se explotaron en el ataque, a determinar por qué tuvo lugar, a reparar el daño que causó y a implementar un plan de contingencia, si existe. Tanto la estrategia reactiva como la proactiva funcionan conjuntamente para desarrollar directivas y controles de seguridad con el fin de reducir los ataques y el daño que causan. El equipo de respuesta a incidentes debe incluirse en los pasos adoptados durante o después del ataque para ayudar a evaluarlo, a documentar el evento y a aprender de él.

j) Evaluar el daño.

Determine el daño causado durante el ataque. Esto debe hacerse lo antes posible para que puedan comenzar las operaciones de restauración. Si no se puede evaluar el daño a tiempo, debe implementarse un plan de contingencia para que puedan proseguir las operaciones comerciales y la productividad normales.

k) Determinar la causa del daño.

Para determinar la causa del daño, es necesario saber a qué recursos iba dirigido el ataque y qué puntos vulnerables se explotaron para obtener acceso o perturbar los servicios. Revise los registros del sistema, los registros de auditoría y las pistas de auditoría. Estas revisiones suelen ayudar a descubrir el lugar del sistema en el que se originó el ataque y qué otros recursos resultaron afectados.

m) Reparar el daño.

Es muy importante que el daño se repare lo antes posible para restaurar las operaciones comerciales normales y todos los datos perdidos durante el ataque.

Los planes y procedimientos para la recuperación de desastres de la organización (que se tratan en el documento acerca del diseño de la seguridad) deben cubrir la estrategia de restauración. El equipo de respuesta a incidentes también debe poder controlar el proceso de restauración y recuperación, y ayudar en este último.

n) Documentar y aprender.

Es importante documentar el ataque una vez que se ha producido. La documentación debe abarcar todos los aspectos que se conozcan del mismo, entre los que se incluyen el daño que ha causado (en hardware y software, pérdida de datos o pérdida de productividad), los puntos vulnerables y las debilidades que se explotaron durante el ataque, la cantidad de tiempo de producción perdido y los procedimientos tomados para reparar el daño. La documentación ayudará a modificar las estrategias proactivas para evitar ataques futuros o mermar los daños.

ñ) Implementar un plan de contingencia.

Si ya existe algún plan de contingencia, se puede implementar para ahorrar tiempo y mantener el buen funcionamiento de las operaciones comerciales. Si no hay ningún plan de contingencia, desarrolle un plan apropiado basado de la documentación del paso anterior.

l) Revisar el resultado y hacer simulaciones.

Tras el ataque o tras defenderse de él, revise su resultado con respecto al sistema.

La revisión debe incluir la pérdida de productividad, la pérdida de datos o de hardware, y el tiempo que se tarda en recuperarlos.

Documente también el ataque y, si es posible, haga un seguimiento del lugar en el que se originó, qué métodos se utilizaron para iniciarlo y qué puntos vulnerables se explotaron. Para obtener los mejores resultados posibles, realice simulaciones en un entorno de prueba.

o) Revisar la eficacia de las directivas.

Si hay directivas para defenderse de un ataque que se ha producido, hay que revisar y comprobar su eficacia. Si no hay directivas, se deben redactar para aminorar o impedir ataques futuros.

p) Adjustar las directivas en consecuencia.

Si la eficacia de la directiva no llega al estándar, hay que ajustarla en consecuencia. Las actualizaciones de las directivas debe realizarlas el personal directivo relevante, los responsables de seguridad, los administradores y el equipo de respuesta a incidentes. Todas las directivas deben seguir las reglas e instrucciones generales de la organización.

1.4. Amenazas a la seguridad de la información.

Los tres elementos principales a proteger en cualquier sistema informático son el *software*, el *hardware* y los datos. Contra cualquiera de los tres elementos dichos anteriormente (pero principalmente sobre los datos) se pueden realizar multitud de ataques o, dicho de otra forma, están expuestos a diferentes amenazas. Generalmente, la taxonomía más elemental de estas amenazas las divide en cuatro grandes grupos: interrupción, interceptación, modificación y fabricación. Un ataque se clasifica como **interrupción** si hace que un objeto del sistema se pierda, quede inutilizable o no disponible. Se tratará de una **interceptación** si un elemento no autorizado consigue un acceso a un determinado objeto del sistema, y de una **modificación** si además de conseguir el acceso consigue modificar el objeto; algunos autores **(Olovsson, 1992)** consideran un caso especial de la modificación: la **destrucción**, entendiéndola como una modificación que inutiliza al objeto afectado. Por último, se dice que un ataque es una **fabricación** si se trata de una modificación destinada a conseguir un objeto similar al atacado de forma que sea difícil distinguir entre el objeto original y el "fabricado". En la figura 1.3 se muestran estos tipos de ataque de una forma gráfica.

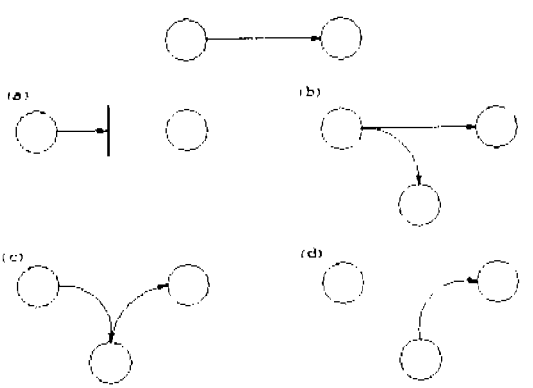


Figura 1.3: Flujo normal de información entre emisor y receptor y posibles amenazas: (a) interrupción, (b) interceptación, (c) modificación y (d) fabricación **(Gallo, 2001).**

En la gran mayoria de publicaciones relativas a la seguridad informática en general, tarde o temprano se intenta clasificar en grupos a los posibles elementos que pueden atacar nuestro sistema. Con frecuencia, especialmente en las obras menos técnicas y más orientadas a otros aspectos de la seguridad **(Icove, 1995) (Meyer, 1989)**, se suele identificar a los atacantes únicamente como personas; esto tiene sentido si hablamos por ejemplo de responsabilidades por un delito informático. Pero en este trabajo es preferible hablar de "elementos" y no de personas: aunque a veces lo olvidemos, nuestro sistema puede verse perjudicado por múltiples entidades aparte de humanos. A continuación se presenta una relación de los elementos que potencialmente pueden amenazar a nuestro sistema. No pretende ser exhaustiva, ni por supuesto una taxonomía formal (para este tipo de estudios, se recomienda consultar **(Landwher,1994)(Aslam, 1996)**; simplemente trata de proporcionar una idea acerca de qué o quién amenaza un sistema.

a) Personas.

No podernos engañarnos, la mayoría de ataques a nuestro sistema van a provenir en última instancia de personas que, intencionada o inintencionadamente, pueden causarnos enormes pérdidas.

Aquí se listan los diferentes tipos de personas que de una u otra forma pueden constituir un riesgo para nuestros sistemas; generalmente se dividen en dos grandes grupos: los atacantes **pasivos**, aquellos que fisgonean por el sistema pero no lo modifican o destruyen, y los **activos**, aquellos que dañan el objetivo atacado, o lo modifican en su favor. Generalmente los curiosos y los *crackers* realizan ataques pasivos (que se pueden convertir en activos), mientras que los terroristas y ex-empleados realizan ataques activos puros; los intrusos remunerados suelen ser atacantes pasivos si nuestra red o equipo no es su

objetivo, y activos en caso contrario, y el personal realiza ambos tipos indistintamente, dependiendo de la situación concreta.

- Personal.
- Ex-empleados.
- Curiosos.
- Crackers.
- Terroristas.
- Intrusos (remunerados).

b) Amenazas lógicas.

Bajo la etiqueta de "amenazas lógicas" encontramos todo tipo de programas que de una forma u otra pueden dañar a nuestro sistema, creados de forma intencionada para ello (*software* malicioso, también conocido como *malware*) o simplemente por error (*bugs* o agujeros). Una excelente lectura que estudia las definiciones de algunas de estas amenazas y su implicación se presenta en **(Garfinkel, 1996)**; otra buena descripción, pero a un nivel más general, se puede encontrar en:

- **Software incorrecto.** Las amenazas más habituales a un sistema provienen de errores cometidos de forma involuntaria por los programadores de sistemas o de aplicaciones.
- **Herramientas de Seguridad.** Cualquier herramienta de seguridad representa un arma de doble filo: de la misma forma que un administrador las utiliza para detectar y solucionar fallos en sus sistemas o en la subred completa, un potencial intruso las puede utilizar para detectar esos mismos fallos y aprovecharlos para atacar los equipos.
- **Puertas traseras.** Durante el desarrollo de aplicaciones grandes o de sistemas operativos es habitual entre los programadores insertar "atajos". A estos atajos se

les denomina puertas traseras. Algunos programadores pueden dejar estos atajos en las versiones definitivas de su *software*; la cuestión es que si un atacante descubre una de estas puertas traseras (no nos importa el método que utilice para hacerlo) va a tener un acceso global a datos que no debería poder leer.

• **Bombas lógicas.** Las bombas lógicas son partes de código de ciertos programas que permanecen sin realizar ninguna función hasta que son activadas. Los activadores más comunes de estas bombas lógicas pueden ser la ausencia o presencia de ciertos ficheros, la llegada de una fecha concreta; cuando la bomba se activa va a poder realizar cualquier tarea que pueda realizar la persona, los efectos pueden ser fatales.

• **Virus.** Un virus es una secuencia de código que se inserta en un fichero ejecutable (denominado *huésped*), de forma que cuando el archivo se ejecuta, el virus también lo hace, insertándose a sí mismo en otros programas.

• **Gusanos.** Un gusano es un programa capaz de ejecutarse y propagarse por sí mismo a través de redes, en ocasiones portando virus o aprovechando *bugs* de los sistemas a los que conecta para dañarlos.

• **Caballos de Troya.** Los troyanos o caballos de Troya son instrucciones escondidas en un programa de forma que éste parezca realizar las tareas que un usuario espera de él, pero que realmente ejecute funciones ocultas sin el conocimiento del usuario.

c) Catástrofes.

Las catástrofes (naturales o artificiales) son la amenaza menos probable contra los entornos habituales: simplemente por su ubicación geográfica, a nadie se le escapa que la probabilidad de sufrir un terremoto o una inundación que afecte a los sistemas informáticos en una gran ciudad, es relativamente baja, al menos en comparación con el riesgo de sufrir un intento de acceso por parte de un pirata o una infección por virus. Sin embargo, el hecho de que las catástrofes sean amenazas poco probables no implica que contra ellas no se tomen unas medidas básicas, ya que si se produjeran generarían los mayores daños.

En este capitulo he tocado a profundidad los temas más importantes de acuerdo a la seguridad informática. Desde los conceptos más básicos hasta las estrategias y métodos que se deben seguir para hacer que un sistema o una aplicación sean seguros, confiables y ofrezca calidad.

Capítulo 2.

Seguridad en UNIX y Herramientas de Seguridad.

2.1. La Seguridad en UNIX.

En la década de los ochenta para mucha gente el concepto de *seguridad* era algo inimaginable en el entorno Unix: la facilidad con que un experto podía acceder a un sistema, burlar todos sus mecanismos de protección y conseguir el máximo nivel de privilegio era algo de sobra conocido por todos, por lo que nadie podía pensar en un sistema Unix *seguro*.

Afortunadamente, los tiempos han cambiado mucho desde entonces. Aunque en un principio y según uno de sus creadores, Unix no se diseñó para ser seguro **(Ritchie,1986)**, a finales de los 80 se convirtió en el primer sistema operativo en alcanzar niveles de seguridad casi militares. En la actualidad se puede considerar el sistema operativo de propósito general más fiable del mercado; cualquier entorno Unix puede ofrecer los mecanismos de seguridad suficientes para satisfacer las necesidades de la mayoría de instituciones. El problema es que en muchas ocasiones se pone a trabajar a Unix tal y como se instala por defecto, lo que convierte a cualquier sistema operativo, Unix o no, en un auténtico agujero en cuanto a seguridad se refiere: cuentas sin *passwords* o con *passwords* por defecto, servicios abiertos, sistemas de ficheros susceptibles de ser compartidos.

A la vista de lo comentado en este punto, parece claro que Unix ha dejado de ser ese sistema arcaico e inseguro de sus primeros tiempos para convertirse en el entorno de trabajo más fiable dentro de la gama de sistemas operativos de propósito general; sin embargo, por alguna extraña razón, mucha gente tiende a considerar todavía a los equipos Unix como amenazas en la red, especialmente a

los clones gratuitos como Linux o FreeBSD que habitualmente se ejecutan en PCs, el hecho de que sean gratuitos no implica en ningún momento que sean inestables, y mucho menos, inseguros: empresas tan importantes como *Yahoo!* (www.yahoo.com) o *Citroën* (www.citroen.com), o el propio servicio postal de Estados Unidos utilizan estos entornos como servidores *Web* o como *cortafuegos* en sus redes.

2.2. Seguridad: Sistema de Archivos.

Dentro del sistema Unix todo son archivos: desde la memoria física del equipo hasta el ratón, pasando por módems, teclado, impresoras o terminales. Esta filosofía de diseño es uno de los factores que más éxito y potencia proporciona a Unix **(Kernighan,1984)**, pero también uno de los que más peligros entraña: un simple error en un permiso puede permitir a un usuario modificar todo un disco duro o leer los datos tecleados desde una terminal. Por esto, una correcta utilización de los permisos, atributos y otros controles sobre los ficheros es vital.

En un sistema Unix típico existen tres tipos básicos de archivos: archivos planos, directorios, y archivos especiales (dispositivos); generalmente, al hablar de *archivos* nos solemos referir a todos ellos si no se especifica lo contrario. Los **archivos planos** son secuencias de *bytes* que *a priori* no poseen ni estructura interna ni contenido significante para el sistema: su significado depende de las aplicaciones que interpretan su contenido. Los **directorios** son archivos cuyo contenido son otros archivos de cualquier tipo (planos, más directorios, o archivos especiales), y los **archivos especiales** son archivos que representan dispositivos del sistema; este último tipo se divide en dos grupos: los dispositivos orientados a carácter y los orientados a bloque. La principal diferencia entre ambos es la forma de realizar operaciones de entrada/salida: mientras que los dispositivos orientados a carácter las realizan *byte* a *byte* (esto es, carácter a carácter), los orientados a bloque las realizan en bloques de caracteres.

El **sistema de archivos** es la parte del nucleo más visible por los usuarios; se encarga de abstraer propiedades fisicas de diferentes dispositivos para proporcionar una interfaz única de almacenamiento: el archivo. Cada sistema Unix tiene su sistema de archivos nativo (por ejemplo, *ext2* en Linux, UFS en Solaris o EFS en IRIX), por lo que para acceder a todos ellos de la misma forma el núcleo de Unix incorpora una capa superior denominada VFS (*Virtual File System*) encargada de proporcionar un acceso uniforme a diferentes tipos de sistema de archivos.

2.2.1. Permisos de un archivo.

Los permisos de cada archivo son la protección más básica de estos objetos del sistema operativo; definen quién puede acceder a cada uno de ellos, y de qué forma puede hacerlo. Cuando hacemos un listado largo de ciertos archivos podemos ver sus permisos junto al tipo de fichero correspondiente, en la primera columna de cada línea:

```
anita: ~# ls -l /sbin/rc0
-rwxr--r-- 3 root sys          2689 Dec 1 1998 /sbin/rc0
anita:~#
```

En este caso vemos que el archivo listado es un archivos plano (el primer carácter es un `-') y sus permisos son `rwxr-r-'. ¿Cómo interpretar estos caracteres?. Los permisos se dividen en tres ternas en función de a qué usuarios afectan; cada una de ellas indica la existencia o la ausencia de permiso para leer, escribir o ejecutar el archivo: una r indica un permiso de lectura, una w de escritura, una x de ejecución y un `-' indica que el permiso correspondiente no está activado. Así, si en una de las ternas tenemos los caracteres rwx, el usuario o usuarios afectados por esa terna tiene o tienen permisos para realizar cualquier operación sobre el

archivo. ¿De qué usuarios se trata en cada caso? La primera terna afecta al propietario del archivo, la segunda al grupo del propietario cuando lo creó (recordemos un mismo usuario puede pertenecer a varios grupos) y la tercera al resto de usuarios.

Una forma sencilla para controlar esto es utilizar el comando umask, por ejemplo:

Con umask 022 los archivos tendrán los permisos: - r w – r - - r - -
Con umask 077 crea los archivos con los permisos: - r w - - - - - - -

Cuando un usuario intenta acceder en algún modo a un archivo, el sistema comprueba qué terna de permisos es la aplicable y se basa únicamente en ella para conceder o denegar el acceso; así, si un usuario es el propietario del archivo sólo se comprueban permisos de la primera terna; si no, se pasa a la segunda y se aplica en caso de que los grupos coincidan, y de no ser así se aplican los permisos de la última terna.

De esta forma es posible tener situaciones tan curiosas como la de un usuario que no tenga ningún permiso sobre uno de sus archivos, y en cambio que el resto de usuarios del sistema pueda leerlo, ejecutarlo o incluso borrarlo; obviamente, esto no es lo habitual, y de suceder el propietario siempre podrá restaurar los permisos a un valor adecuado.

Después de ver el procedimiento de modificación de los permisos de un archivo, este puede parecer demasiado complicado y arcaico para un sistema operativo moderno; a fin de cuentas, mucha gente prefiere gestores gráficos de permisos, igual que prefiere gestores gráficos para otras tareas de administración , programas que dan todo hecho y no obligan al administrador a "complicarse" llenos de menús desplegables y diálogos que una y otra vez preguntan si realmente deseamos modificar cierto permiso (*>Está usted seguro? >Realmente seguro? >Es mayor de edad? >Me lo jura?*). Incluso esas personas aseguran que

el procedimiento gráfico es mucho más claro y más potente que el que Unix ofrece en modo texto. Nada más lejos de la realidad, hemos de pensar que este modelo de protección está vigente desde hace casi treinta años y no ha cambiado **absolutamente nada**. Si en todo este tiempo no se ha modificado el mecanismo, obviamente es porque siempre ha funcionado y lo sigue haciendo bien.

2.2.2. Lista de control de acceso: ACLs.

Las Listas de Control de Accesos (ACL por Access Control List) son un mecanismo proporcionado por el UNIX de Hewlett-Packard (HP-UX) para poner en práctica un sistema de control de accesos discrecional. Su utilización permite aumentar enormemente la flexibilidad con que se manejan los permisos de acceso de los archivos y directorios en UNIX.

El mecanismo clásico de permisos de acceso proporcionado por el UNIX tan solo define tres clases de usuarios a los que se puede asociar dichos permisos: el propietario (u), el grupo (g) y otros (o). Sin embargo utilizando listas de control de accesos es posible manipular los permisos de acceso para cada uno de los usuarios y los grupos del sistema, lo que da lugar a un nivel de selectividad mucho más elevado.

Supongamos por ejemplo que queremos permitir que el usuario *pablo* de nuestro grupo pueda escribir en un archivo dado, pero no queremos conceder este permiso a ninguno de los otros usuarios de nuestro grupo. Este tipo de control de permisos no es posible utilizando el mecanismo tradicional de bits de permiso (rwxrwxrwx).

Supongamos por otro lado que pertenecemos al grupo *inf* y queremos conceder el permiso de lectura de un archivo dado a todos los usuarios del grupo *temp*. Utilizando el método tradicional la única forma de hacerlo es conceder este mismo

permiso a todos los usuarios de grupos distintos de *inf*. Si embargo , utilizando las listas de control de accesos podemos lograr los dos objetivos anteriores, dado que con las ACLs podemos manejar los permisos a nivel de usuarios individuales, a nivel de grupos individuales, e incluso a nivel de usuarios concretos en grupos concretos.

Cada archivo y directorio del sistema tiene asociada una lista de control de accesos (ACL), siendo cada elemento o entrada de la lista de la forma (usuario.grupo,modo). Cada una de las entradas de la lista de un archivo especifica los permisos de acceso a ese archivo para un usuario en un grupo.

2.3. Auditoría del Sistema.

Casi todas las actividades realizadas en un sistema Unix son susceptibles de ser, en mayor o menor medida, monitorizadas: desde las horas de acceso de cada usuario al sistema hasta las páginas *Web* más frecuentemente visitadas, pasando por los intentos fallidos de conexión, los programas ejecutados o incluso el tiempo de CPU que cada usuario consume. Obviamente esta facilidad de Unix para recoger información tiene unas ventajas inmediatas para la seguridad: es posible detectar un intento de ataque nada más producirse el mismo, así como también detectar usos indebidos de los recursos o actividades "sospechosas"; sin embargo, existen también desventajas, ya que la gran cantidad de información que potencialmente se registra puede ser aprovechada para crear negaciones de servicio o, más habitualmente, esa cantidad de información puede hacer difícil detectar problemas por el volumen de datos a analizar **(Schneier, 1998)**, por esta razón hablamos muy poco este tema.

2.4. Sistemas de detección de intrusos.

A pesar de que un enfoque clásico de la seguridad de un sistema informático siempre define como principal defensa del mismo sus controles de acceso (desde una política implantada en un cortafuegos hasta unas listas de control de acceso en un *router* o en el propio sistema de ficheros de una máquina), esta visión es extremadamente simplista si no tenemos en cuenta que en muchos casos esos controles no pueden protegernos ante un ataque **(Lunt, 1990)**.

Por poner un ejemplo sencillo, pensemos en un *cortafuegos* donde hemos implantado una política que deje acceder al puerto 80 de nuestros servidores *Web* desde cualquier máquina de Internet; ese cortafuegos sólo comprobará si el puerto destino de una trama es el que hemos decidido para el servicio HTTP, pero seguramente no tendrá en cuenta si ese tráfico representa o no un ataque o una violación de nuestra política de seguridad: por ejemplo, no detendrá a un pirata que trate de acceder al archivo de contraseñas de una máquina aprovechando un *bug* del servidor *Web*. Desde un pirata informático externo a nuestra organización a un usuario autorizado que intenta obtener privilegios que no le corresponden en un sistema, nuestro entorno de trabajo no va a estar nunca a salvo de intrusiones. Llamaremos **intrusión** a un conjunto de acciones que intentan comprometer la integridad, confidencialidad o disponibilidad de un recurso **(Heady, 1990)**; analizando esta definición, podemos darnos cuenta de que una intrusión no tiene por qué consistir en un acceso no autorizado a una máquina: también puede ser una negación de servicio.

A los sistemas utilizados para detectar las intrusiones o los intentos de intrusión se les denomina **sistemas de detección de intrusiones** (*Intrusion Detection Systems*, IDS) o, más habitualmente y aunque no sea la traducción literal **sistemas de detección de intrusos**; cualquier mecanismo de seguridad con este propósito puede ser considerado un IDS, pero generalmente sólo se aplica esta denominación a los sistemas automáticos (*software* o *hardware*): es decir, aunque

un guardia de seguridad que vigila en la puerta de la sala de operaciones pueda considerarse en principio como un sistema de detección de intrusos, como veremos a continuación lo habitual (y lógico) es que a la hora de hablar de IDSs no se contemplen estos casos.

a) Clasificación de los IDSs.

Generalmente existen dos grandes enfoques a la hora de clasificar a los sistemas de detección de intrusos: o bien en función de **qué** sistemas vigilan, o bien en función de **cómo** lo hacen. Si elegimos la primera de estas aproximaciones tenemos dos grupos de sistemas de detección de intrusos: los que analizan actividades de una única máquina en busca de posibles ataques, y los que lo hacen de una subred (generalmente, de un mismo dominio de colisión) aunque se emplacen en uno sólo de los *hosts* de la misma.

Esta última puntualización es importante: un IDS que detecta actividades sospechosas en una red no tiene porqué (y de hecho en la mayor parte de casos no suele ser así) ubicarse en todas las máquinas de esa red.

• **IDSs basados en red**. Un IDS basado en red monitoriza los paquetes que circulan por nuestra red en busca de elementos que denoten un ataque contra alguno de los sistemas ubicados en ella; el IDS puede situarse en cualquiera de los *hosts* o en un elemento que analice todo el tráfico (como un HUB o un enrutador). Esté donde esté, monitorizará diversas máquinas y no una sola: esta es la principal diferencia con los sistemas de detección de intrusos basados en *host*.

• **IDSs basados en máquina**. Mientras que los sistemas de detección de intrusos basados en red operan bajo todo un dominio de colisión, los basados en máquina realizan su función protegiendo un único sistema; de una forma similar guardando las distancias, por supuesto a como actúa un escudo antivirus residente en MS-DOS, el IDS es un proceso que trabaja en *background* (o que despierta

periódicamente) buscando patrones que puedan denotar un intento de intrusión y alertando o tomando las medidas oportunas en caso de que uno de estos intentos sea detectado.

La segunda gran clasificación de los IDSs se realiza en función de cómo actúan estos sistemas; actualmente existen dos grandes técnicas de detección de intrusos **(Serlin, 1991)**: las basadas en la detección de anomalías (*anomaly detection*) y las basadas en la detección de usos indebidos del sistema (*misuse detection*). Aunque más tarde hablaremos con mayor profundidad de cada uno de estos modelos, la idea básica de los mismos es la siguiente:

• **Detección de anomalías**. La base del funcionamiento de estos sistemas es suponer que una intrusión se puede ver como una anomalía de nuestro sistema, por lo que si fuéramos capaces de establecer un perfil del comportamiento habitual de los sistemas seríamos capaces de detectar las intrusiones por pura estadística: probablemente una intrusión sería una desviación excesiva de la media de nuestro perfil de comportamiento.

• **Detección de usos indebidos**. El funcionamiento de los IDSs basados en la detección de usos indebidos presupone que podemos establecer patrones para los diferentes ataques conocidos y algunas de sus variaciones; mientras que la detección de anomalías conoce lo normal (en ocasiones se dice que tienen un "conocimiento positivo", *positive knowledge*) y detecta lo que no lo es, este esquema se limita a conocer lo anormal para poderlo detectar (conocimiento negativo, *negative knowledge*).

2.5. Firewall.

Un *firewall* es un sistema o grupo de sistemas que hace cumplir una política de control de acceso entre dos redes. De una forma más clara, podemos definir que un firewall como cualquier sistema (desde un simple *router* hasta varias redes en serie) utilizado para separar, en cuanto a seguridad se refiere, una máquina o

subred del resto, protegiéndola así de servicios y protocolos que desde el exterior puedan suponer una amenaza a la seguridad. El espacio protegido, denominado **perímetro de seguridad**, suele ser propiedad de la misma organización, y la protección se realiza contra una red externa, no confiable, llamada **zona de riesgo (Ranum, 1995).** Evidentemente la forma de aislamiento más efectiva para cualquier política de seguridad consiste en el aislamiento físico, es decir, no tener conectada la máquina o la subred a otros equipos o a Internet (figura 1.4 (A)). Sin embargo, en la mayoría de organizaciones, los usuarios necesitan compartir información con otras personas situadas en muchas ocasiones a miles de kilómetros de distancia, con lo que no es posible un aislamiento total. El punto opuesto consistiría en una conectividad completa con la red (figura 1.5 (B)), lo que desde el punto de vista de la seguridad es muy problemático: cualquiera, desde cualquier parte del mundo, puede potencialmente tener acceso a nuestros recursos.

Un término medio entre ambas aproximaciones consiste en implementar cierta separación lógica mediante un cortafuegos (figura 2.1 (C)).

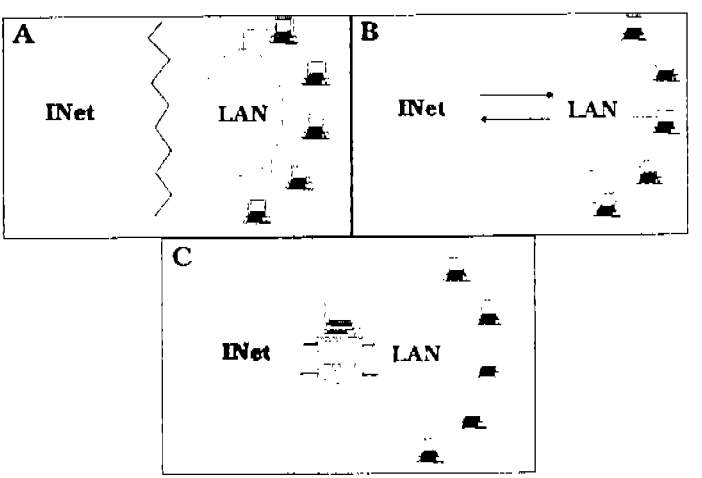


Figura 2.1: (A) Aislamiento, (B) Conexión total, (C) Cortafuegos entre la zona de riesgo y el perímetro de seguridad **[Lizárraga, 2001]**.

Antes de hablar de los firewall es casi obligatorio dar una serie de definiciones de partes o características de funcionamiento de un ***firewall***; por máquina o *host* **bastión** (también se denominan **gates**) se conoce a un sistema especialmente asegurado, pero en principio vulnerable a todo tipo de ataques por estar abierto a Internet, que tiene como función ser el punto de contacto de los usuarios de la red interna de una organización con otro tipo de redes. El *host* bastión filtra tráfico de entrada y salida, y también esconde la configuración de la red hacia fuera. Por **filtrado de paquetes** entendemos la acción de denegar o permitir el flujo de tramas entre dos redes (por ejemplo la interna, protegida con el *cortafuegos*, y el resto de Internet) de acuerdo a unas normas predefinidas; aunque el filtro más elemental puede ser un simple *router*, trabajando en el nivel de red del protocolo OSI, esta actividad puede realizarse además en un puente o en una máquina individual.

El filtrado también se conoce como *screening*, y a los dispositivos que lo implementan se les denomina **chokes**; el *choke* puede ser la máquina bastión o un elemento diferente. Un **proxy** es un programa (trabajando en el nivel de aplicación de OSI) que permite o niega el acceso a una aplicación determinada entre dos redes.

Los *firewall* son cada vez más necesarios en nuestras redes, pero todos los expertos recomiendan que no se usen **en lugar de** otras herramientas, sino **junto a** ellas; cualquier firewall, desde el más simple al más avanzado, presenta dos gravísimos problemas de seguridad: por un lado, centralizan todas las medidas en un único sistema, de forma que si éste se ve comprometido y el resto de nuestra red no está lo suficientemente protegido el atacante consigue amenazar a toda la subred simplemente poniendo en jaque a una máquina. El segundo problema, relacionado con éste, es la falsa sensación de seguridad que un firewall proporciona: generalmente un administrador que no disponga de un firewall va a preocuparse de la integridad de todas y cada una de sus máquinas, pero en el

momento en que instala el firewall y lo configura asume que toda su red es segura, por lo que se suele descuidar enormemente la seguridad de los equipos de la red interna. Esto, como acabamos de comentar, es un grave error, ya que en el momento que un pirata acceda a nuestro firewall, recordemos que es un sistema muy expuesto a ataques externos, automáticamente va a tener la posibilidad de controlar toda nuestra red. Además, esto ya no es un problema de los *firewall* sino algo de sentido común, un firewall evidentemente no protege contra ataques que no pasan por él: esto incluye todo tipo de ataques internos dentro del perímetro de seguridad, pero también otros factores que *a priori* no deberían suponer un problema.

2.5.1. Componentes de un Firewall.

a). Proxy de aplicación. Es habitual que los firewall utilicen aplicaciones *software* para reenviar o bloquear conexiones a servicios como *finger*, *telnet* o FTP; a tales aplicaciones se les denomina servicios **proxy**, mientras que a la máquina donde se ejecutan se le llama **pasarela de aplicación**. Los servicios *proxy* poseen una serie de ventajas de cara a incrementar nuestra seguridad **(Wack, 1994)**; en primer lugar, permiten únicamente la utilización de servicios para los que existe un *proxy*, por lo que si en nuestra organización la pasarela de aplicación contiene únicamente *proxies* para *telnet*, HTTP y FTP, el resto de servicios no estarán disponibles para nadie.

Una segunda ventaja es que en la pasarela es posible filtrar protocolos basándose en algo más que la cabecera de las tramas, lo que hace posible por ejemplo tener habilitado un servicio como FTP pero con órdenes restringidas (podríamos bloquear todos los comandos put para que nadie pueda subir ficheros a un servidor

b) Monitorización de la actividad. Monitorizar la actividad de nuestro firewall es algo indispensable para la seguridad de todo el perímetro protegido; la monitorización nos facilitará información sobre los intentos de ataque que estemos sufriendo (origen, franjas horarias, tipos de acceso...), así como la existencia de tramas que aunque no supongan un ataque *a priori* sí que son al menos sospechosas **(Bellovin, 1993).**

c) Filtrado de paquetes. Un *firewall* sencillo puede consistir en un dispositivo capaz de filtrar paquetes, un *choke*: se trata del modelo de firewall más antiguo **(Schimmel, 1997)**, basado simplemente en aprovechar la capacidad de algunos *routers*, denominados *screening routers*, para hacer un enrutado selectivo, es decir, para bloquear o permitir el tránsito de paquetes. En un firewall de filtrado de paquetes los accesos desde la red interna al exterior que no están bloqueados son directos (no hay necesidad de utilizar *proxies*, como sucede en los firewall basados en una máquina con dos tarjetas de red), por lo que esta arquitectura es la más simple de implementar (en muchos casos sobre *hardware* ya ubicado en la red) y la más utilizada en organizaciones que no precisan grandes niveles de seguridad, como las que vemos aquí. No obstante, elegir un firewall tan sencillo puede no ser recomendable en ciertas situaciones, o para organizaciones que requieren una mayor seguridad para su subred, ya que los simples *chokes* presentan más desventajas que beneficios para la red protegida. El principal problema es que no disponen de un sistema de monitorización sofisticado, por lo que muchas veces el administrador no puede determinar si el *router* está siendo atacado o si su seguridad ha sido comprometida. Además las reglas de filtrado pueden llegar a ser complejas de establecer, y por tanto es difícil comprobar su corrección: habitualmente sólo se comprueba a través de pruebas directas, con los problemas de seguridad que esto puede implicar.

2.6. Criptografía.

En el *mundo real*, si una universidad quiere proteger los expedientes de sus alumnos los guardará en un armario ignífugo, bajo llave y vigilado por guardias, para que sólo las personas autorizadas puedan acceder a ellos para leerlos o modificarlos; si queremos proteger nuestra correspondencia de curiosos, simplemente usamos un sobre; si no queremos que nos roben dinero, lo guardamos en una caja fuerte.

Lamentablemente, en una red no disponemos de todas estas medidas que nos parecen habituales: la principal (podríamos decir **única**) forma de protección va a venir de la mano de la criptografía. El cifrado de los datos nos va a permitir desde proteger nuestro correo personal para que ningún curioso lo pueda leer, hasta controlar el acceso a nuestros archivos de forma que sólo personas autorizadas puedan examinar (o lo que quizás es más importante, modificar) su contenido, pasando por proteger nuestras claves cuando conectamos a un sistema remoto o nuestros datos bancarios cuando realizamos una compra a través de Internet. Hemos presentado con anterioridad algunas aplicaciones que utilizan de una u otra forma la criptografía para proteger nuestra información **(Schneier, 1998)**. La **criptología** (del griego *krypto* y *logos*, estudio de lo oculto, lo escondido) es la ciencia que trata los problemas teóricos relacionados con la seguridad en el intercambio de mensajes en clave entre un emisor y un receptor a través de un canal de comunicaciones (en términos informáticos, ese canal suele ser una red de computadoras). Esta ciencia está dividida en dos grandes ramas: la **criptografía**, ocupada del cifrado de mensajes en clave y del diseño de criptosistemas (hablaremos de éstos más adelante), y el **criptoanálisis**, que trata de descifrar los mensajes en clave, rompiendo así el criptosistema. En lo sucesivo nos centraremos más en la criptografía y los criptosistemas que en el criptoanálisis, ya que nos interesa, más que romper sistemas de cifrado (lo cual es bastante complicado cuando trabajamos con criptosistemas serios), el saber cómo funcionan éstos y conocer el diseño elemental de algunos sistemas seguros. La

criptografía es una de las ciencias consideradas como más antiguas, ya que sus orígenes se remontan al nacimiento de nuestra civilización.

Su uso original era el proteger la confidencialidad de informaciones militares y políticas, pero en la actualidad es una ciencia interesante no sólo en esos círculos cerrados, sino para cualquiera que esté interesado en la confidencialidad de unos determinados datos: actualmente existe multitud de software y hardware destinado a analizar y monitorizar el tráfico de datos en redes de computadoras; si bien estas herramientas constituyen un avance en técnicas de seguridad y protección, su uso indebido es al mismo tiempo un grave problema y una enorme fuente de ataques a la intimidad de los usuarios y a la integridad de los propios sistemas. Aunque el objetivo original de la criptografía era mantener en secreto un mensaje, en la actualidad no se persigue únicamente la privacidad o **confidencialidad** de los datos, sino que se busca además garantizar la **autenticidad** de los mismos (el emisor del mensaje es quien dice ser, y no otro), su **integridad** (el mensaje que leemos es el mismo que nos enviaron) y su **no repudio** (el emisor no puede negar el haber enviado el mensaje).

2.6.1. Criptosistemas.

a) Criptosistemas de clave secreta. Denominamos criptosistema de clave secreta (de clave privada, de clave única o simétrico) a aquel criptosistema en el que la clave de cifrado, puede ser calculada a partir de la de descifrado y viceversa. En la mayoría de estos sistemas, ambas claves coinciden, y por supuesto han de mantenerse como un secreto entre emisor y receptor: si un atacante descubre la clave utilizada en la comunicación, ha roto el criptosistema. Hasta la década de los setenta, la invulnerabilidad de todos los sistemas dependía de este mantenimiento en secreto de la clave de cifrado. Este hecho presentaba una gran desventaja: había que enviar, aparte del criptograma, la clave de cifrado del emisor al receptor, para que éste fuera capaz de descifrar el mensaje. Por

tanto, se incurría en los mismos peligros al enviar la clave, por un sistema que había de ser supuestamente seguro, que al enviar el texto plano.

b) Criptosistemas de clave pública. En 1976, Whitfield Diffie y Martin Hellman, de la Universidad de Stanford, demostraron la posibilidad de construir criptosistemas que no precisaran de la transferencia de una clave secreta en su trabajo **(Diffie, 1976)**. Esto motivó multitud de investigaciones y discusiones sobre la criptografía de clave pública y su impacto, hasta el punto que la NSA (*National Security Agency*) estadounidense trató de controlar el desarrollo de la criptografía, ya que la consideraban una amenaza peligrosa para la seguridad nacional. Veamos ahora en que se basan los criptosistemas de clave pública. En éstos, la clave de cifrado se hace de conocimiento general (se le llama **clave pública**). Sin embargo, no ocurre lo mismo con la clave de descifrado (**clave privada**), que se ha de mantener en secreto. Ambas claves no son independientes, pero del conocimiento de la pública no es posible deducir la privada sin ningún otro dato (recordemos que en los sistemas de clave privada sucedía lo contrario).

Tenemos pues un par clave pública-clave privada; la existencia de ambas claves diferentes, para cifrar o descifrar, hace que también se conozca a estos criptosistemas como **asimétricos**. Cuando un receptor desea recibir una información cifrada, ha de hacer llegar a todos los potenciales emisores su clave pública, para que estos cifren los mensajes con dicha clave. De este modo, el único que podrá descifrar el mensaje será el legítimo receptor, mediante su clave privada. Matemáticamente, si es el algoritmo cifrador y el descifrador, se ha de cumplir que representando un mensaje, y siendo y las claves de descifrado y cifrado, respectivamente.

2.6.2. Criptoanálisis.

El criptoanálisis es la ciencia opuesta a la criptografía (quizás no es muy afortunado hablar de ciencias *opuestas*, sino más bien de ciencias *complementarias*), ya que si ésta trata principalmente de crear y analizar criptosistemas seguros, la primera intenta romper esos sistemas, demostrando su vulnerabilidad: dicho de otra forma, trata de descifrar los criptogramas.

El término *descifrar* siempre va acompañado de discusiones de carácter técnico, aunque asumiremos que descifrar es conseguir el texto en claro a partir de un criptograma, sin entrar en polémicas de reversibilidad y solidez de criptosistemas. En el análisis para establecer las posibles debilidades de un sistema de cifrado, se han de asumir las denominadas condiciones del peor caso: (1) el criptoanálista tiene acceso completo al algoritmo de encriptación, (2) el criptoanálista tiene una cantidad considerable de texto cifrado, y (3) el criptoanálista conoce el texto en claro de parte de ese texto cifrado.

2.7. Herramientas de Seguridad.

¿Por qué utilizar herramientas de seguridad en los sistemas Unix? Ningún sistema operativo se puede considerar "seguro" tal y como se instala por defecto; normalmente, cualquier distribución de un sistema se instala pensando en proporcionar los mínimos problemas a un administrador que desee poner la máquina a trabajar inmediatamente, sin tener que preocuparse de la seguridad. Es una cuestión de puro *marketing*: imaginemos un sistema Unix que por defecto se instalara en su modo más restrictivo en cuanto a seguridad; cuando el administrador desee ponerlo en funcionamiento conectándolo a una red, ofreciendo ciertos servicios, gestionando usuarios y periféricos, deberá conocer muy bien al sistema, ya que ha de dar explícitamente los permisos necesarios para realizar cada tarea, con la consiguiente pérdida de tiempo. Es mucho más

productivo para cualquier empresa desarrolladora de sistemas proporcionarlos completamente abiertos, de forma que el administrador no tenga que preocuparse mucho de cómo funciona cada parte del sistema que acaba de instalar: simplemente inserta el CDROM original, el *software* se instala, y todo funciona a la primera, aparentemente sin problemas. Esta política, que lamentablemente siguen casi todas las empresas desarrolladoras, convierte a un sistema Unix que no se haya configurado mínimamente en un fácil objetivo para cualquier atacante.

Es más, la complejidad de Unix hace que un administrador que para aumentar la seguridad de su sistema se limite a cerrar ciertos servicios de red o detener algunos demonios obtenga una sensación de falsa seguridad, esta persona va a pensar que su sistema es seguro simplemente por realizar un par de modificaciones en él, cosa que es completamente falsa.

a) Nessus. Es la herramienta de evaluación de seguridad "Open Source" de mayor renombre. Nessus es un escáner de seguridad remoto para Linux, BSD, Solaris y Otros Unix. Está basado en plug-in(s), tiene una interfaz basada en GTK, y realiza más de 1200 pruebas de seguridad remotas. Permite generar reportes en HTML, XML, LaTeX, y texto ASCII; también sugiere soluciones para los problemas de seguridad.

b) Ethereal. Oliendo el pegamento que mantiene a Internet unida. Ethereal es un analizador de protocolos de red para Unix y Windows, y es libre {free}. Nos permite examinar datos de una red viva o de un archivo de captura en algún disco. Se puede examinar interactivamente la información capturada, viendo información de detalles y sumarios por cada paquete. Ethereal tiene varias caracteristicas poderosas, incluyendo un completo lenguaje para filtrar lo que querremos ver y la habilidad de mostrar el flujo reconstruido de una sesión de TCP. Incluye una versión basada en texto llamada tethereal.

c) **Tripwire.** El abuelo de las herramientas de comprobación de integridad de archivos.

Un comprobador de integridad de archivos y directorios. Tripwire es una herramienta que ayuda a administradores y usuarios de sistemas monitoreando alguna posible modificación en algún set de archivos. Si se usa regularmente en los archivos de sistema (por Ej. diariamente), Tripwire puede notificar a los administradores del sistema, si algún archivo fue modificado o reemplazado, para que se puedan tomar medidas de control de daños a tiempo.

d) Titan. Este software de auditoria informática, detecta problemas de seguridad en la máquina local. El mismo se limita a informarnos de los posibles problemas de seguridad que podemos tener.

e) TCP Wrappers. TCP Wrappers se encarga de definir una serie de redes o máquinas autorizadas a conectar con nosotros. Cualquier administrador que desee un mínimo de seguridad ha de instalar TCP Wrappers en sus equipos; incluso algunos Unixes como Linux o BSDI lo ofrecen por defecto al instalar el operativo.

f) SSH (Secure Shell). Es un software cuya principal función es permitir la conexión remota segura a sistemas a través de canales inseguros, aunque también se utiliza para la ejecución de órdenes en ese sistema remoto o transferir archivos desde o hacia él de manera fiable.

g) Crack. Crack, desarrollado por el experto en seguridad Alec Muffet, es el "adivinador" de contraseñas más utilizado en entornos Unix; actualmente se encuentra en su versión 516.5, que funciona correctamente en la mayoría de clones del sistema operativo (Linux, Solaris, OSF...). Ejecutar periódicamente Crack sobre el archivo de contraseñas de sus sistemas es algo muy recomendable para cualquier administrador minimamente preocupado por la seguridad, sin

importar que se utilicen mecanismos para obligar a los usuarios a elegir passwords aceptables.

En este capítulo hemos tocado lo temas más importante de la seguridad en UNIX, de acuerdo a la investigación que se realizo en este trabajo de tesis los temas que exponemos son importantes para alcanzar un buen nivel de seguridad.

Capítulo 3.

Análisis de una Herramienta de Apoyo para Asegurar las Aplicaciones Web del Centro de Cómputo.

3.1. ¿Por qué es tan importante contar con una herramienta de apoyo para asegurar las aplicaciones Web?

Hoy en día los piratas informáticos o los llamados "hackers" llegan a ser cada vez más sostificados, haciendo cada vez más difícil proteger la integridad de las aplicaciones y la información que estas guardan. Proteger estas aplicaciones poniendo parches manualmente es una estrategia que tarde o temprano fallara. La seguridad Web, en la actualidad, debe ser construida de abajo hacia arriba, desde el desarrollo de la aplicación, pruebas de calidad, el despliegue y el mantenimiento.

Por estas razones es muy importante que las Aplicaciones Web del Centro cuenten con una Herramienta de Apoyo, para mantener su seguridad, confiabilidad y calidad que estas aplicaciones requieren; de esta manera, obtener y mantener una buena imagen de los servicios que brinda el Centro de Computo.

Una Herramienta de Apoyo permite construir la seguridad de la aplicación, esto es muy importante, porque el costo relativo de arreglar los errores y defectos que tiene una aplicación después del despliegue, es casi 15 veces más grande que hacerlo en el desarrollo.

3.2. AppScan DE: Una Herramienta de Apoyo como solución para asegurar las Aplicaciones Web del Centro de Computo.

3.2.1. ¿Qué es AppScan DE?

AppScan DE es una poderosa herramienta de pruebas que permite el rápido desarrollo de la seguridad. Esta herramienta ayuda a hacer que la lógica de la aplicación sea resistente a ataques sin tocar su presentación o eficacia. **AppScan DE** detecta los defectos de la seguridad automáticamente; como un componente integrado al desarrollo de la empresa, esta herramienta, automatiza las pruebas de creación de escritura, modificación y proceso de mantenimiento, asegurando confiabilidad y pruebas que son repetibles.

AppScan DE es una herramienta que ayuda a las empresas a reducir costos y a crear aplicaciones confiables y resistentes contra hackers, en el ambiente de desarrollo.

Esta herramienta escánea las aplicaciones Web desde el punto de vista del usuario.

AppScan DE explora la aplicación y aprende la lógica del negocio.

AppScan DE crea "Vulnerabilidades Potenciales" que son los defectos potenciales de la seguridad en el código y entonces los prueba para verificar que ellos existen. Las "Vulnerabilidades Potenciales" son las Aplicaciones Específicas relacionadas directamente a nuestra aplicación.

AppScan DE reporta los errores o defectos de la aplicación, luego se los proporciona al usuario para empezar a arreglar estos errores.

3.2.2. Proceso del AppScan DE.

AppScan DE encuentra y prueba las vulnerabilidades desconocidas (ASVs):

- Vulnerabilidades Desconocidas: también llamadas Vulnerabilidades Específicas de la Aplicación (ASVs). Estas vulnerabilidades son el producto de las aplicaciones que se configuran impropiamente o cuyas reglas para el uso válido no son impuestas apropiadamente por la lógica de las aplicaciones.
- Una vez que **AppScan DE** ha identificado las vulnerabilidades potenciales dentro de una aplicación Web, prueba cada vulnerabilidad potencial para determinar su severidad y la mejor manera de arreglarlo.

3.2.3. AppScan DE asegura las Aplicaciones Web en contra de las Perversiones del Web.

Una Perversión del Web es una manera en el que un pirata informático o un hacker modifica o destruye una aplicación Web, sin el debido permiso del dueño de la aplicación y todo lo que necesita es un pequeño agujero en el código, un examinador del Web y una pequeña determinación.

A continuación la figura 3.1 muestra cuales son las perversiones del Web que asegura **AppScan DE**:

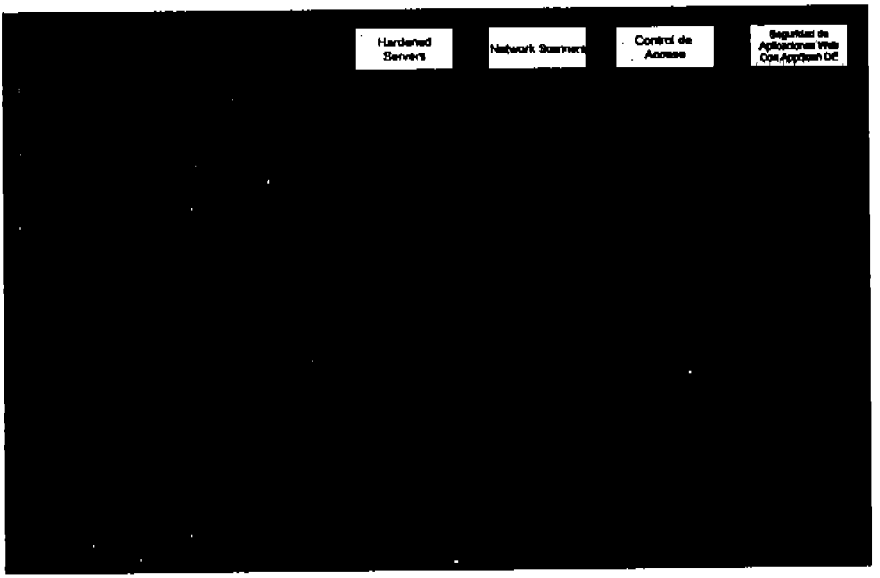


Figura 3.1. Perversiones del Web

Ahora mostraremos ejemplos y casos de algunas de estas perversiones del Web, de como estos tipos de perversiones del Web pueden ser fácilmente ejecutados para robar dinero, transferencia ilegales de dinero, obtener información de clientes o consumidores y destruir el sitio Web.

a) Manipulación Escondida.

Situación:

Hay una aplicación Web en una tienda en línea, el objeto más vendido de la semana es una cámara para PC, el precio es de $129 US, ver figura 3.2.

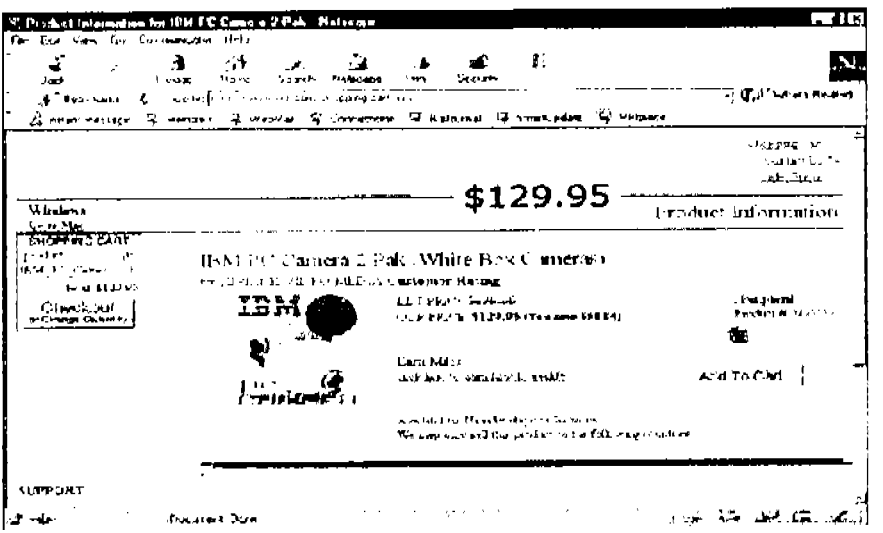


Figura 3.2. Manipulación Escondida **[Sanctum, 2004]**.

Punto de Ataque:

Para facilitar un desarrollo rápido de esta aplicación de comercio electrónico fue diseñada con campos escondidos (hidden fields). La información del precio se puso en un campo escondido, con la asignación de $129 US, ver figura 3.3.

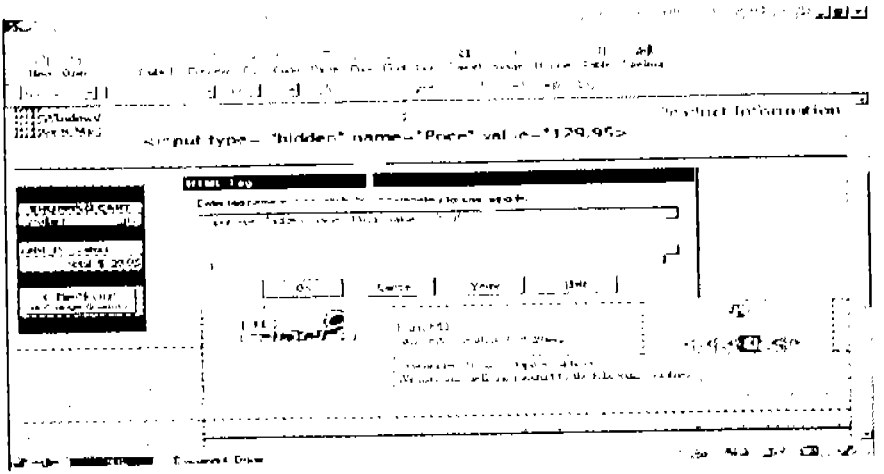


Figura 3.3. Manipulación Escondida, punto de ataque **(Sanctum, 2004)**.

El desarrollador de la aplicación asume que la información del precio seguirá intacta. Pero si se usa un estándar *Netscape HTML Editor* un hacker puede cambiar el valor del campo escondido y cambiarlo a $1.95 US, ver figura 3.4.

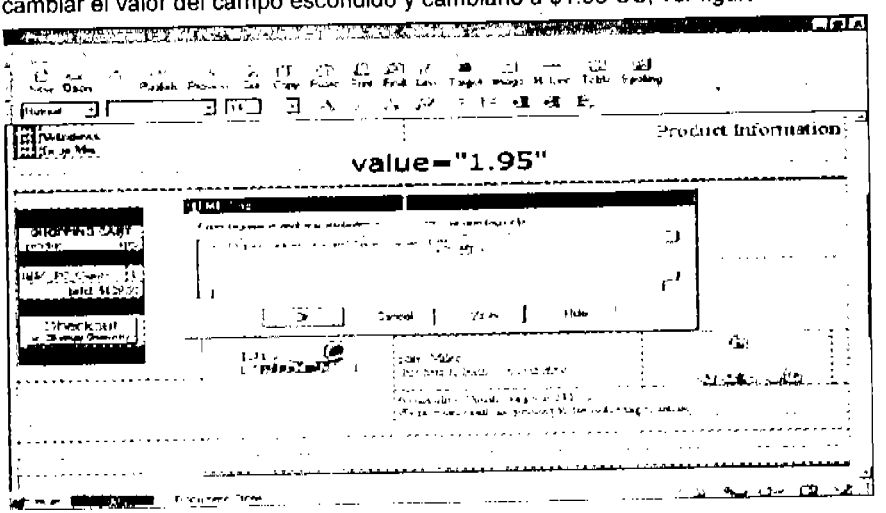


Figura 3.4. Manipulación Escondida, punto de ataque (cont) **(Sanctum, 2004)**.

Daño:

El hacker somete el pequeño cambio en la página HTML, entonces el puede comprar la misma cámara por el precio de $1.95 US, ver figura 3.5.

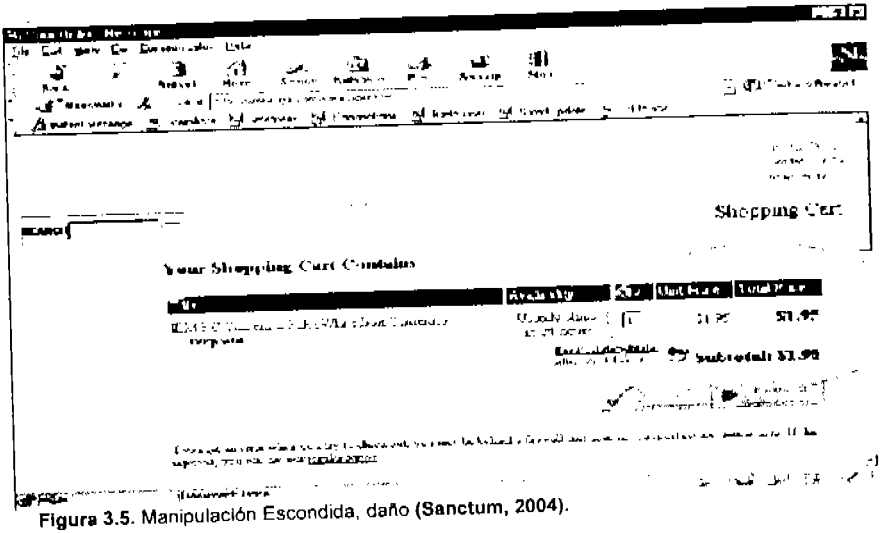


Figura 3.5. Manipulación Escondida, daño **(Sanctum, 2004).**

b) Envenenamiento de *Cookies*.

Situación:

Supongamos que hay una típica página Web de pagos de cuentas en línea, ya que este tipo de servicios se ha vuelto muy popular. En este ejemplo un cliente o un hacker llamado "Abacarius" ingresa al sitio y descubre las cantidades que debe pagar, ver figura 3.6.

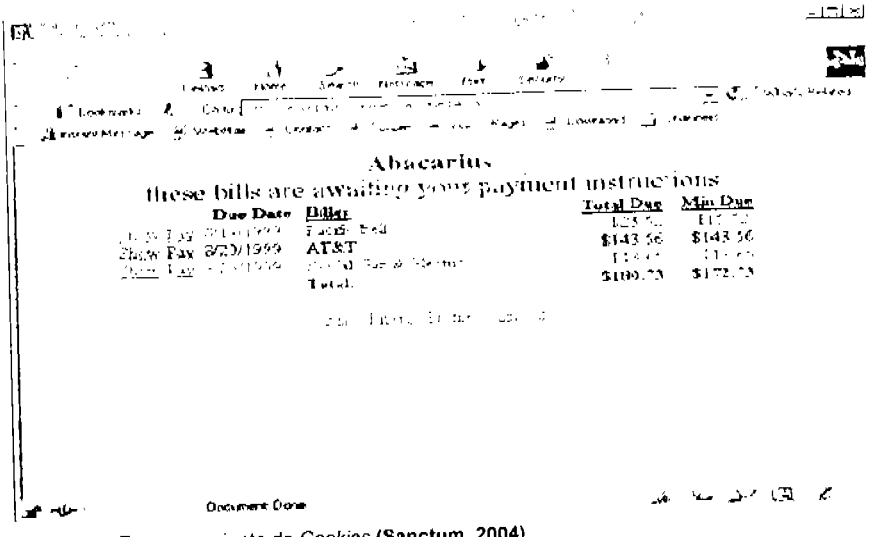


Figura 3.6. Envenenamiento de *Cookies* **(Sanctum, 2004)**

Punto de Ataque:

El área vulnerable en este ejemplo es el "*cookie*", es una pequeña pieza de información que el sitio pone en la computadora de cualquier cliente que ingresa. El "*cookie*" pone información que identifica al cliente para el sitio y es incluido con cualquier petición enviada al sitio, ver figura 3.7.

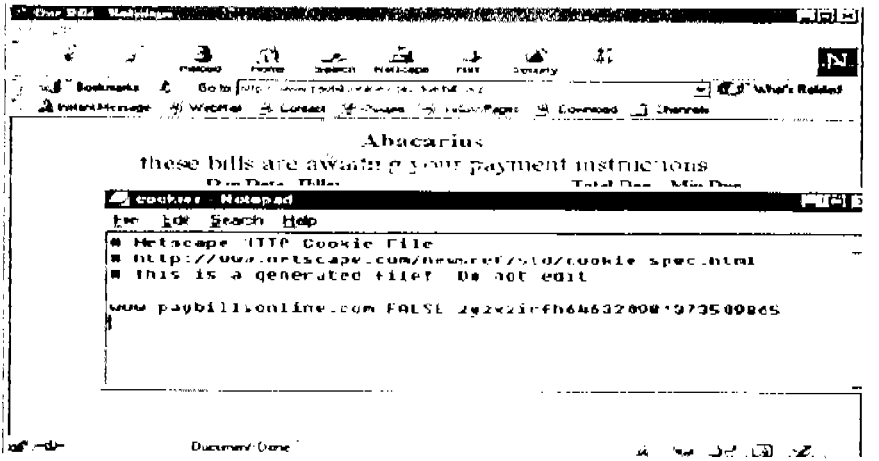


Figura 3.7. Envenenamiento de *Cookies*, punto de ataque **(Sanctum, 2004)**

Si nos fijamos en el archivo de cookies, nos encontraremos con una extraña combinación de letras y números. Nos estaremos enterando de la conexión entre el nombre del consumidor, Abacarius, y el *cookie* encriptado, zyzxrir. En esta encriptación, la "a" se vuelve "z" y "b" se vuelve "y", ver figura 3.8.

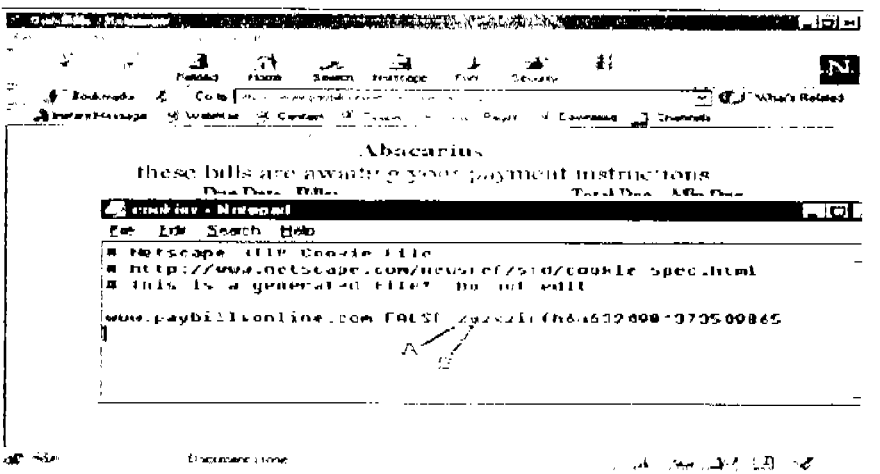


Figura 3.8. Envenenamiento de *Cookies*, punto de ataque (cont) **(Sanctum, 2004)**.

Cookies es un alimento para los hackers listos. En este caso el hacker toma el *cookie* original y la convierte, para hacer que este sitio lo reconozca como "Johnson". "J" se vuelve "q" y "o" se vuelve "i". Una vez que el *cookie* es re-encriptada, el hacker esta listo para hacer uso de "Jonson", ver figura 3.9.

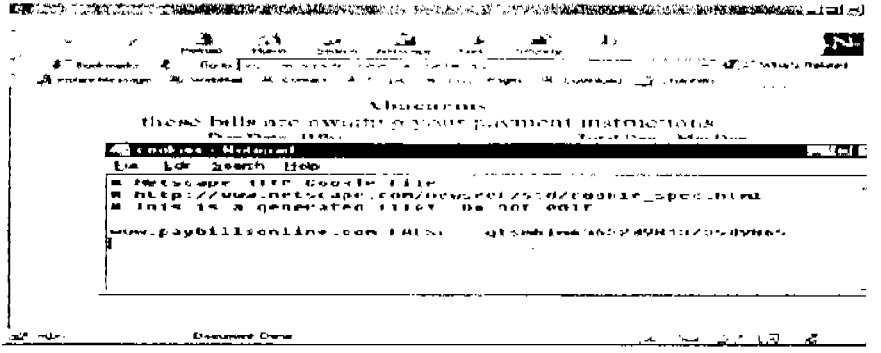


Figura 3.9. Envenenamiento de *Cookies*, punto de ataque (cont) **(Sanctum, 2004)**

Daño:

Ahora este sitio reconoce Abacarius como Johnson, y deja que el vea la cuenta de Johnson, puede borrarlo, o ser especialmente generoso y actualizar el pago de algunas cuentas solo con la manipulación de números, ver figura 3.10.

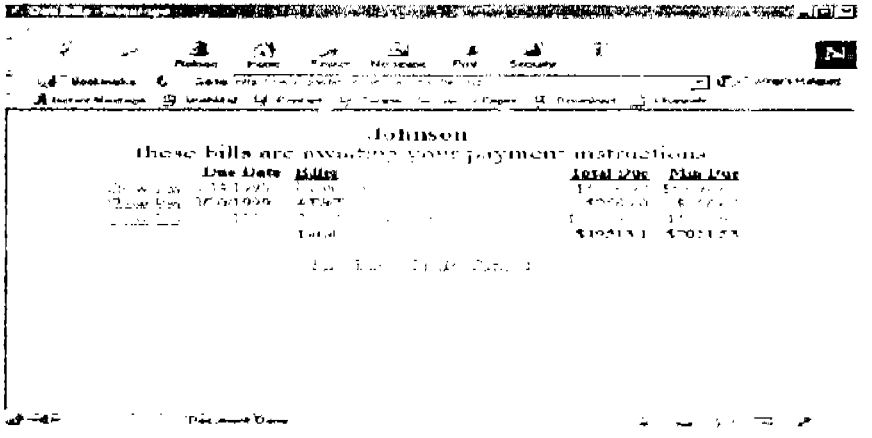


Figura 3.10. Envenenamiento de *Cookies*, daño **(Sanctum, 2004)**.

c) Puertas Traseras.

Situación:

El chequeo de cuentas en el sitio de un banco es muy popular por los clientes. Con a nueva gama de servicios financieros en línea, los clientes del banco pueden fácilmente cceder a su record financiero y hacer transacciones financieras sofistificadas, incluyendo transferencias de dinero, ver figura 3.11.

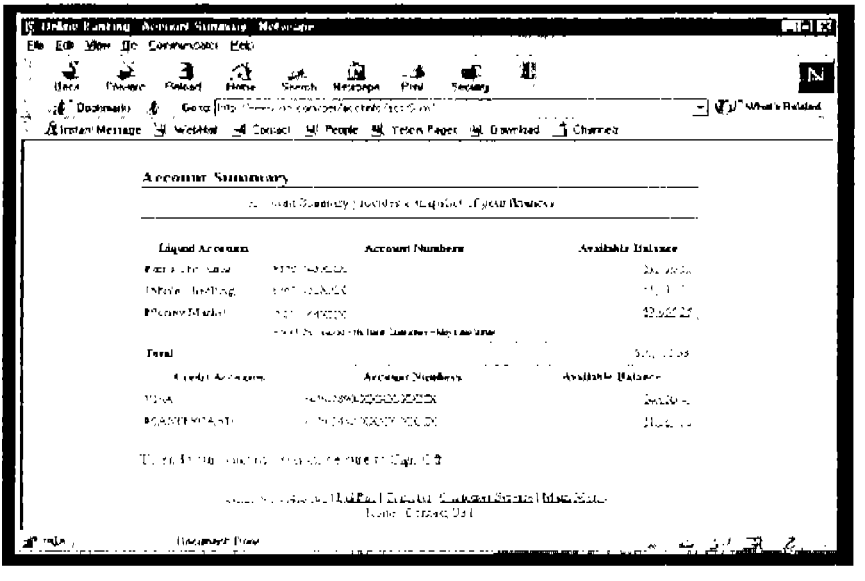


Figura 3.11. Puertas Traseras **(Sanctum, 2004)**

Punto de Ataque:

La administración del banco, cree tener sus aplicaciones de transferencias de dinero en un lugar tranquilo. En la prisa de tener estas aplicaciones en línea, la opción de depuración utilizada para probar durante el desarrollo se dejo con

errores. ¿Puedes imaginar lo que un hacker puede hacer en esta situación?, ver figura 3.12.

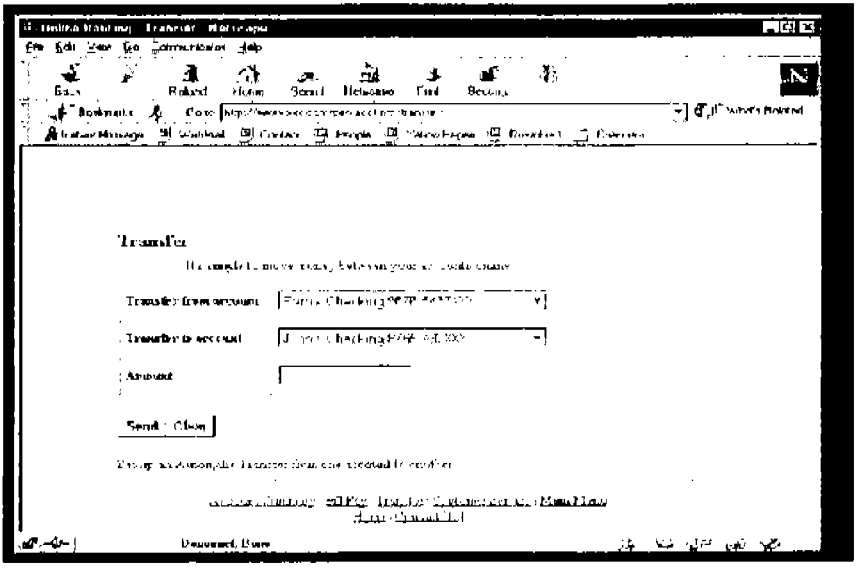


Figura 3.12. Puertas Traseras, punto de ataque **(Sanctum, 2004)**.

Daño:

Transmitiendo una petición a la transferencia CGI con un parámetro debug "=on", el hacker activa el *debug o atajo* y comienza a manipular una característica que fue erróneamente dejada en la versión final de la aplicación. Usando una opción común del *debug*, el hacker descubre que puede fácilmente transferir cualquier cantidad de dinero de cualquier cuenta a cualquier otra cuenta, ver figura 3.13.

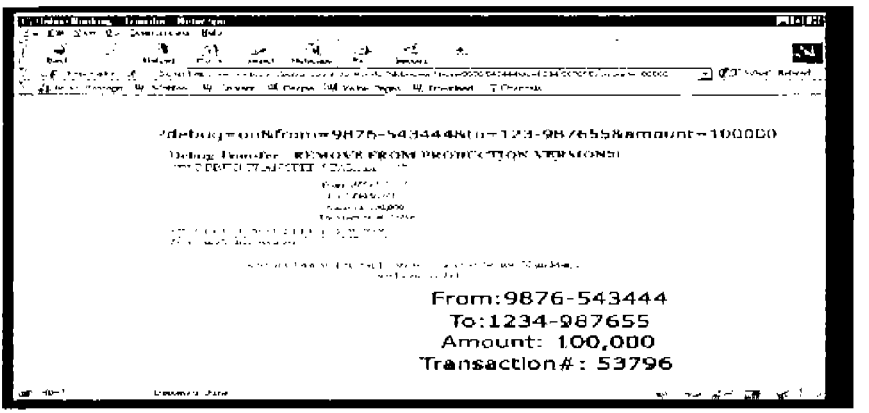


Figura 3.13. Puertas Traseras, daño **(Sanctum, 2004)**.

d) Vulnerabilidades Conocidas.

Situación:

En este sitio de comercio electrónico, los administradores manejan varias tareas usando una interfaz Web. Un procedimiento de la entrada protege esta área de la administración de usuarios no invitados, ver figura 3.14.

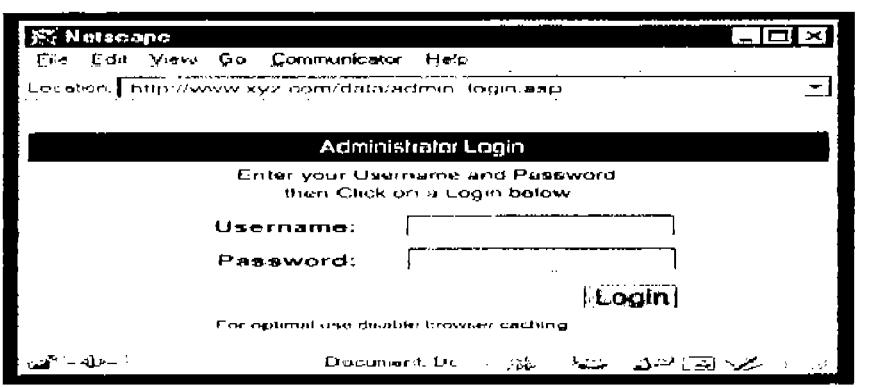


Figura 3.14. Vulnerabilidades Conocidas **(Sanctum, 2004)**

Punto de Ataque:

Te has dado cuenta en el URL, que este sitio se relaciona con la tecnología de *Microsoft Active Server Page* (ASP). Usado en millones de sitios, la tecnología ASP infortunadamente ha obtenido debilidades que hackers listos y persistentes pueden explotar. Una rápida búsqueda de seguridad relacionada con un sitio Web, www.securityfocus.com, revela un número de de potenciales y peligrosos problemas con ASPs. Una debilidad, NT *ASP Alternate Data Streams Vulnerability*, deja a cualquiera ver el código fuente de cualquier ASP, ver figura 3.15.

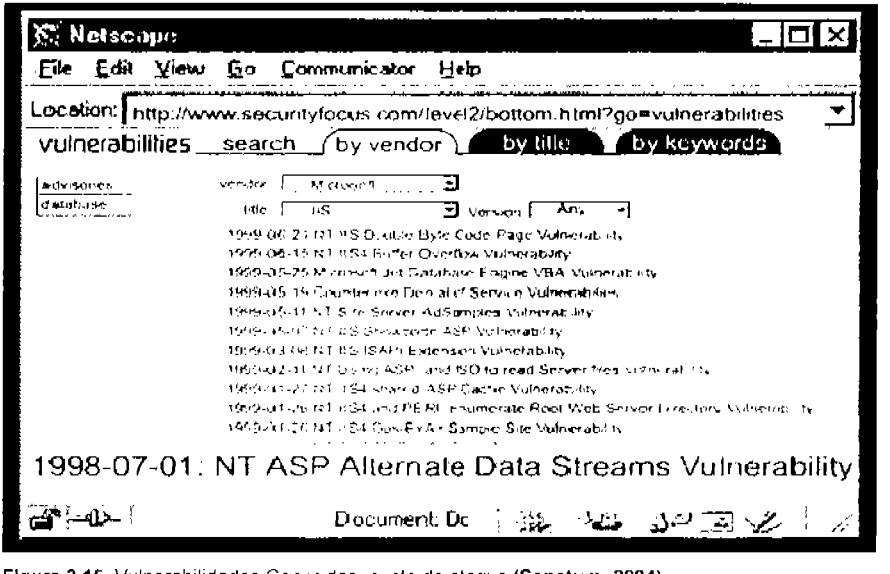


Figura 3.15. Vulnerabilidades Conocidas, punto de ataque **(Sanctum, 2004)**.

El problema es uno de muchos encontrados en software de terceros. Con demasiados "*bugs*" por arreglar y parches para aplicar, los programadores simplemente no pueden mantener el ritmo. La mayoría de los sitios Web son vulnerables de una o otra manera, incluso teniendo parches recientemente aplicados.

Daño:

En este ejemplo, el último parche no fue aplicado, es una puerta abierta para un hacker. Añadiendo al URL "*::$DATA*" al nombre del ASP y refrescando la página, un hacker puede instruir al servidor Web que le mande el código fuente completo para el "admin._login" ASP, ver figura 3.16.

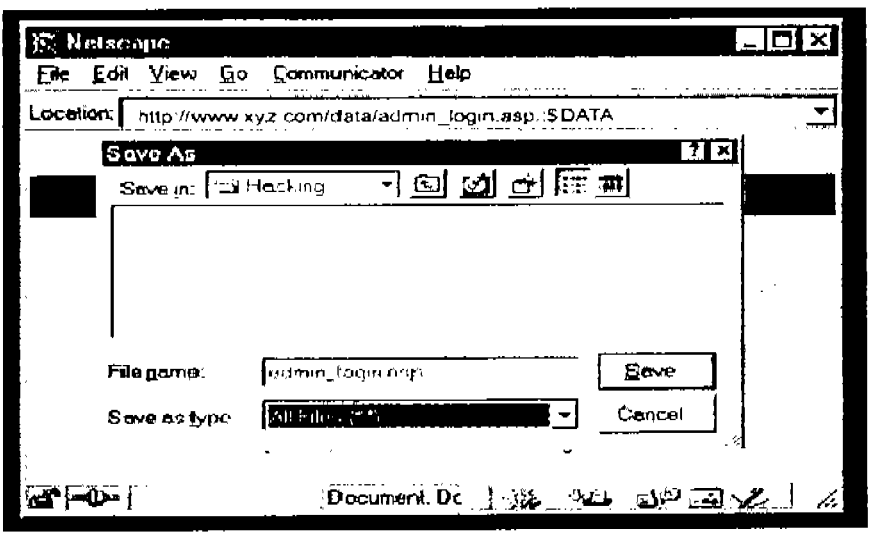


Figura 3.16. Vulnerabilidades Conocidas, daño **(Sanctum, 2004)**.

Si miramos el código fuente ASP, observamos que tenemos el nombre del administrador (XYZ_Root) y también la contraseña (SeCuRiTy@1). Ahora el hacker tiene el completo control del sitio Web, ver figura 3.17.

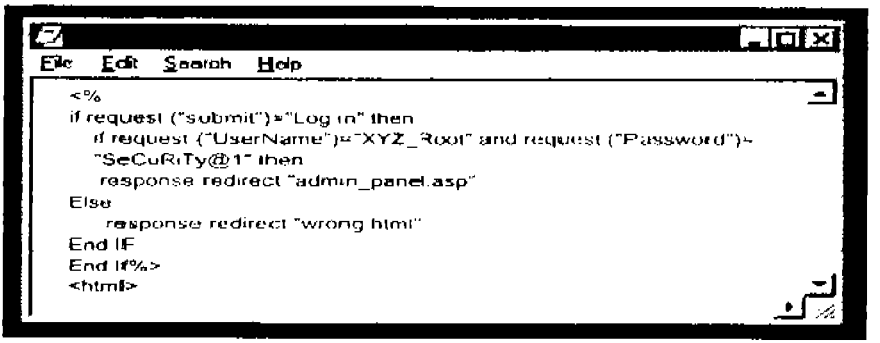


Figura 3.17. Vulnerabilidades Conocidas, daño (cont) **[Sanctum, 2004]**.

e) Manipulando Parámetros.

Situación:

Para un hacker, información personal es como dinero en el banco. Una vez que mete sus manos en información confidencial, tales como, record médicos, el hacker puede rápidamente volver esa información en dinero. En este ejemplo, los clientes de una farmacia en línea, tienen sus perfiles de salud en un sitio Web. Asumiendo que esta información será bien protegida. El cliente Jenny Smith incluyó información de alta confidencialidad en su perfil, ver figura 3.18.

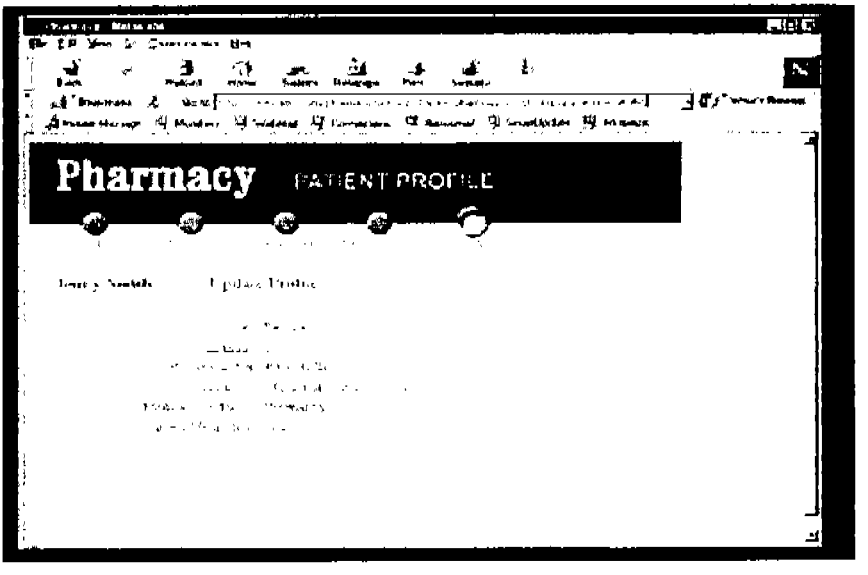


Figura 3.18. Manipulando Parámetros **(Sanctum, 2004)**.

Punto de Ataque:

Miren cerca del URL que da acceso al perfil de Jenny. Se darán cuenta que contiene el ID del paciente, un parámetro que identifica solamente a ella. Quedarán asombrados cuando vean que pasa cuando un hacker cambia el ID del paciente a "*" y refresca la página del perfil, ver figura 3.19.

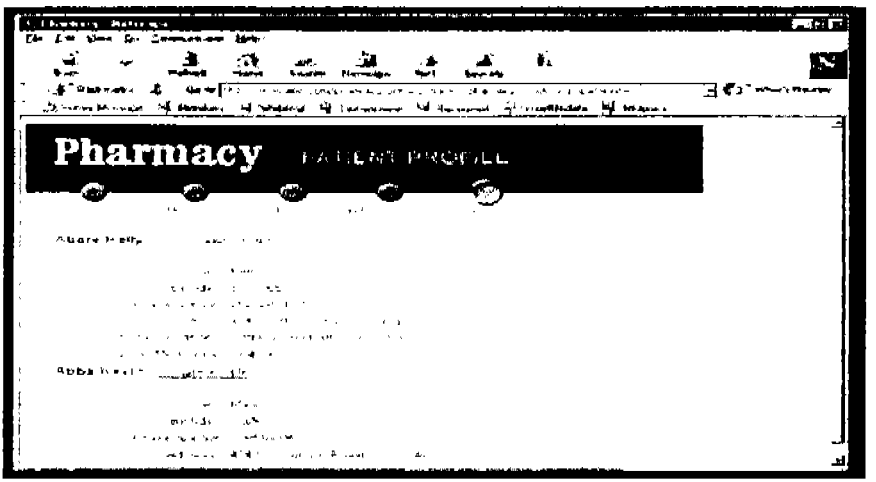


Figura 3.19. Manipulando Parámetros, punto de ataque **(Sanctum, 2004)**.

Daño:

Con este simple comando, el hacker ahora tiene acceso a la entera base de datos de los pacientes, incluyendo las condiciones médicas de todos los pacientes y medicamentos. La aplicación trata de usar el símbolo "*" para encontrar un paciente específico, este símbolo significa "*matchall*", la aplicación recupera toda la base de datos y la manda al hacker. El resultado no es solo una gran invasión de privacidad, sino también quebranta una serie de leyes en contra de la farmacia.

3.3. Situación en el Centro de Cómputo y sus Necesidades.

Existe una preocupación que es constante por parte de los investigadores que realizan sus proyectos desarrollando aplicaciones Web en el Centro de Computo. Esto es referente a la seguridad, confiabilidad y calidad de las Aplicaciones Web resultado de sus proyectos de investigación. Muchas veces estas Aplicaciones Web son propensas a ataques, de los cuales muchas veces, no nos damos cuentas, ni sabemos cuales son las vulnerabilidades que estas tienen.

La necesidad principal del Centro de Cómputo es que sus aplicaciones Web sean seguras, a que se refiere esto, tanto a la modificación, robo o destrucción total de información valiosa que contienen estas Aplicaciones. Otra necesidad que es muy importante en una organización es que los servicios que brinda sean confiables y de calidad, esto se logra a través de seguimientos de pasos o lineamientos.

De acuerdo con la investigación que se ha realizado en este trabajo de tesis, la mejor manera de solucionar este problema es utilizando una herramienta de apoyo que nos ayude a desminuir este problema. Con la investigación que se llevó a acabo la mejor manera de asegurar las Aplicaciones Web, es saber cuales son sus vulnerabilidades en la etapa de desarrollo. De esta manera, sabiendo cuales son sus vulnerabilidades se arreglan sus errores y se pueden publicar en la red, con menos posibilidad o menos propensas a un ataque informático.

3.4. Beneficios de la Herramienta para el Centro de Cómputo.

Como todos sabemos no se puede asegurar un sistema al 100 por ciento, siempre existen vulnerabilidades. En tiempo real, existen Aplicaciones Web del Centro de Cómputo que muchas de estas no son seguras, que tienen vulnerabilidades. Esta herramienta ayudará a encontrar cuales son las vulnerabilidades existentes que tienen estas Aplicaciones, hacer un reporte y proponer cual es la mejor manera de arreglar estas vulnerabilidades. De esta manera, ahorrar costos significativos en pérdidas si se llegará a realizar algún ataque a estas Aplicaciones.

Otro beneficio que traerá esta herramienta al Centro de Cómputo, es que a partir de que esta herramienta este trabajando, su objetivo principal es encontrar cuales son las vulnerabilidades de las Aplicaciones en la etapa de desarrollo. Esto es, con el fin de que una vez publicada en la red sean menos propensas a ataque de hackers. También brinda una solución para automatizar los análisis de

vulnerabilidades y pruebas de penetración de sus aplicaciones y plataformas Web. Elimina los exámenes manuales que eran necesarios antes de implementar una aplicación, genera reportes que determinan la mejor manera de cumplir con estas auditorías para asegurar sus aplicaciones, antes de su implementación. Como vemos, esta herramienta traerá muchos beneficios al Centro de Cómputo, se le dará un proceso a las aplicaciones descubriendo sus vulnerabilidades y llegando a ese 100 por ciento de seguridad que queremos alcanzar.

3.5. Arquitectura de la Herramienta AppScan DE.

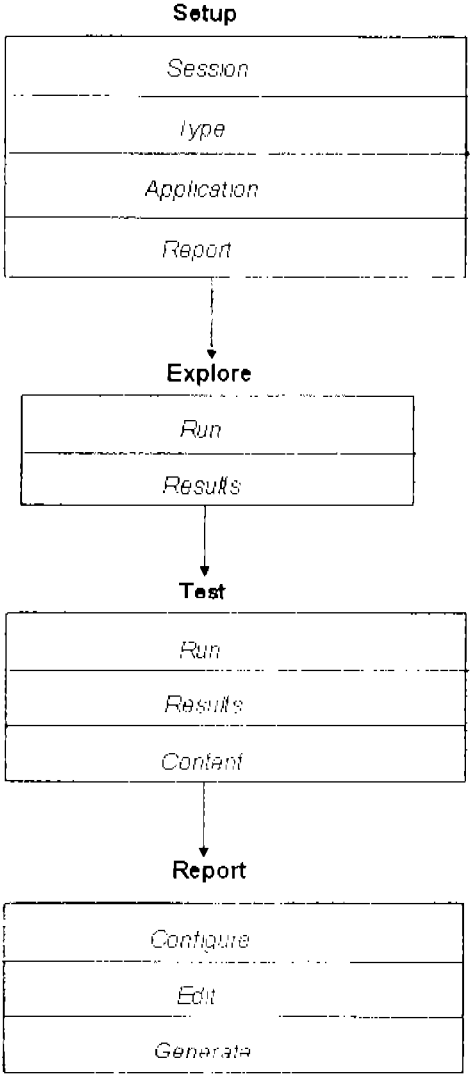


Figura 3.20. Arquitectura de AppScan DE **(Sanctum, 2004).**

Etapa *Setup*. Se selecciona que tipo de Scan quieres correr de la lista de tipos de Scan. Hay dos tipos de Scan: uno que es completamente automático y uno que es "Scan Interactivo". También se puede hacer cambios y salvar uno nuevo, personalizar los tipos de Scan, exactamente a la medida de las necesidades del usuario. Una vez que esta etapa está completa, comienza el proceso de evaluación que consiste en tres partes:

Etapa de Exploración. Durante la etapa de Exploración, se explora el sitio Web, visitando los links que tiene, como lo hace un usuario normal. Esta exploración del sitio puede ser manual, automática, o interactiva (una combinación de las dos), dependiendo del tipo de Scan que se escogió/definió durante la etapa de *Setup*. La exploración se puede realizar completa de la aplicación o se puede dividir en procesos pertinentes de la aplicación. Cuando se explora el sitio Web, se reúne información del sitio, tales como, los links y las respuestas a las peticiones. Esta información se almacena en la base de datos de la etapa de Exploración y se utiliza para crear una lista de "vulnerabilidades potenciales"; las peticiones del URL que se diseñan para probar la elasticidad del sitio Web y revelar sus debilidades.

Etapa de Prueba. Durante la etapa de Prueba, las vulnerabilidades potenciales que se encontraron en la etapa de Exploración son probadas. Estas pruebas utilizan técnicas avanzadas de hackers informáticos para escoger las vulnerabilidades verdaderas (de las muchas vulnerabilidades probadas) y valoran su severidad. Como en la etapa de Exploración, en esta etapa el proceso se puede realizar automáticamente o con muchas intervención del usuario. Una vez, que las pruebas están completas el sistema permite que el usuario verifique los resultados reexaminando manualmente alguna vulnerabilidad potencial. Si el usuario tiene sospecha de cualquiera de los resultados de la prueba, se puede modificar el reporte de la sesión e incluir cualquier paso siguiente para una investigación adicional. Los piratas informáticos pueden obtener información sensible de ciertos archivos de sitio Web viendo su código fuente. La exposición de este código fuente es una vulnerabilidad potencial, así que la revisión del

código fuente es altamente recomendada. Artículos de interés pueden ser añadidos al reporte de la Sesión.

Etapa del Reporte. Esta etapa final procesa la evaluación de las vulnerabilidades y permite personalizar el reporte. La información del reporte incluye las listas de vulnerabilidades potenciales, así como la severidad de vulnerabilidades y recomendaciones verdaderas para así arreglarlas. Además de poder personalizar los reportes, usted puede exportar también los datos crudos del reporte en el formato de CSV, para un análisis adicional.

3.6. Funcionamiento del Sistema.

Consola de Administración de AppScan DE.

La Consola de Administración de **AppScan DE**, se divide en cinco secciones principales, que se muestran en la figura 3.21:

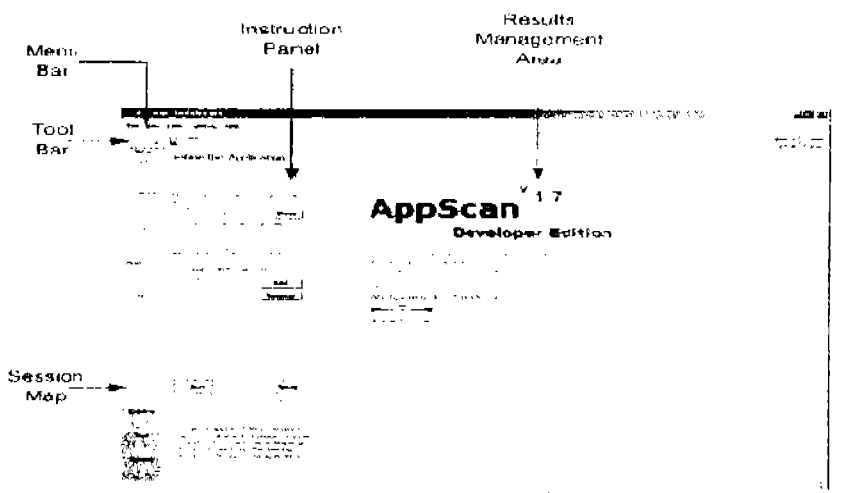


Figura 3.21. Consola de Administración **(Sanctum, 2004).**

Sección del Mapa. Se utiliza la sección de Mapa, para ver y navegar entre las etapas diferentes de AppScan (*Setup, Explore, Test, Results*), haciendo clic en el icono a la que pertenece cada etapa.

Barra de Herramientas (*Toolbar*). Una barra de herramientas uniforme para tareas comúnmente utilizadas, tales como Salva, Abre la sesión, Crea la sesión nueva, etc.

Menú Bar. La barra de Menú de **AppScan DE** tiene cinco menús, que consiste en los siguientes: *File, View, Tools, Settings and Help.*

Panel de Instrucciones (*Instruction Panel*). El Panel de Instrucciones se divide en tres áreas: El área ***Entrada***, aquí es donde se proporciona a AppScan la entrada de datos necesaria para completar el paso. Área de ***Navegación***, incluye los botones de *Next* y *Back* para navegar entre los pasos diferentes. Área de **Información**, proporciona una descripción corta acerca del paso y las acciones que se requirió para completarlo. En la figura 3.22 se muestra estas tres áreas:

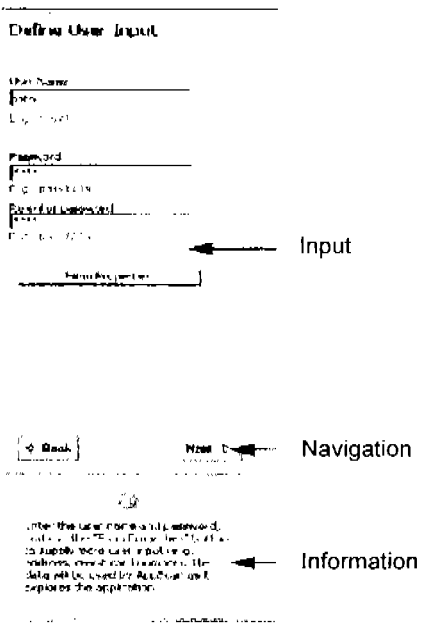


Figura 3.22. Panel de Instrucciones **(Sanctum, 2004).**

Área de Administración de Resultados (*Results Management Area*). El Área del Administración de Resultados se utiliza para ver, manejar y procesar la producción de cada una de las etapas y pasos de **AppScan DE**.

3.6.1. Etapa *Setup*.

Para realizar esta etapa, lo primero que se debe hacer es proporcionar a Pazcan con información preliminar acerca del sitio y escoger el tipo de Scan que se desea realizar. Durante el proceso de *Setup*, se escoge uno de los tipos de Scan y se suministra los escenarios que determinan cómo el específico tipo de Scan se realizará. Es importante notar que las definiciones en *Setup* se salvan y se pueden

volver a utilizar en sesiones futuras. El proceso de *Setup* tiene cuatro partes, para visualizar el contenido de cada parte se hace un clic a cualquiera de ellas:

a) Sesión.

Session: escoger entre crear una nueva sesión y la locación donde se desea salvarla, o seguir trabajando en una sesión ya existente.

b) Tipo.

Type: se escoge el tipo de Scan que se desea realizar, se puede escoger entre los dos tipos de Scan que están predefinidos o se puede crear un nuevo tipo de Scan.

AppScan DE provee los siguientes Scans predefinidos:

Scan Automático. Es el tipo de Scan más completo, no requiere la intervención del usuario durante (y entre) las etapas de Exploración y Prueba, pero permite definir y configurar los parámetros de la etapa de Exploración y Prueba.

Scan Interactivo. Permite al usuario intervenir en el proceso de la etapa de Prueba. También permite al usuario que manualmente se exploren y prueben páginas específicas utilizando el examinador de **AppScan DE**. En la etapa de Prueba se puede volver al modo automático haciendo clic en el botón que dice "*Run*".

El usuario puede modificar las propiedades de un Scan predefinido que convengan a las necesidades del usuario. Una vez que el usuario hizo esto, se puede escánear utilizando la nueva configuración (sin salvar).

Usuario Define Tipo. Permite seleccionar una propiedad del Scan que será salvado en el sistema. (Se activa solamente si al menos una propiedad ha sido salvada).

Propiedades del tipo de Scan, "*This Session scan type properties*". Corre un Scan con las propiedades actuales configuradas por el usuario. (Está opción solo se activa una vez que se hicieron cambios en una predefinida (o que el usuario defina) configuración.

Cada tipo de Scan tiene una única configuración. La configuración del tipo de Scan puede ser vista y modificada en la caja de diálogo de las Propiedades del Tipo del Scan (*Scan Type Properties*), que se mostrará en la figura 3.23. Para ver o editar las propiedades del Scan se hace clic en el botón de "*Scan Type Properties*".

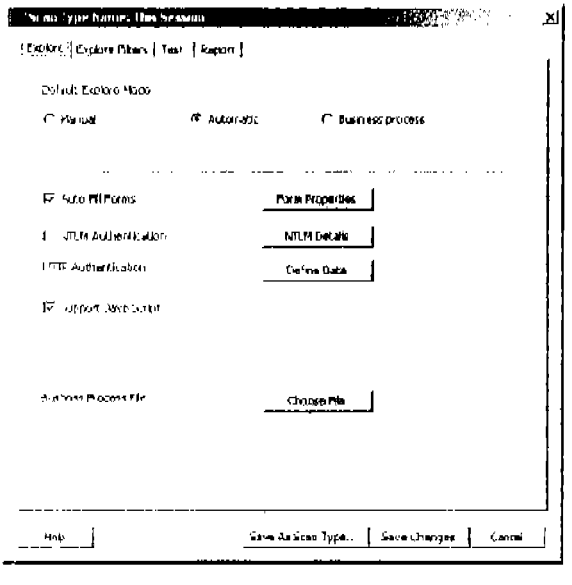


Figura 3.23. Propiedades del tipo de Scan **(Sanctum, 2004).**

La caja de diálogo de las Propiedades del Tipo de Scan, contiene cuatro etiquetas. Las dos primeras contienen las propiedades del Scan para la etapa de Exploración, la tercera para la etapa de Prueba y la cuarta para la etapa de Reporte. Las secciones siguientes describen los campos y propiedades de cada una de estas etiquetas.

Etiqueta *Explore*.

Hay dos etiquetas de *Explore*, la general, que aparece en la figura anterior, y que se explicara en esta parte. Y la etiqueta de "*Explore Filters*" que se explicará más adelante. Las etiquetas de *Explore* se usan para definir el Scan para la etapa de Exploración. La siguiente tabla explica cada una de las partes del contenido de la etiqueta de Exploración:

Tabla 1 Explore Tab Options

Artículo	Descripción
Default Explore Mode	Define el modo en que se va utilizar durante la etapa de Explore. ya sea (Manual. Automatico o Business Process)
Auto Fill Forms	Una vez seleccionado. automáticamente llenará formas necesarias para el sitio que se explora
Form Properties button	Abre un diálogo donde se definen los valores para utilizarlas cuando llene las formas automaticamente
NTML Authentication	(f T LAN Manager) se usa cuando se necesita una autenticacion NTML
NTML Details button.	Abre una dialogo donde se pide el nombre del dominio el usuario y password necesarios para la autenticación NTML
HTTP Authentication define Data button	Abre un diálogo donde se pide el usuario y password necesarios para http autenticación
Support Java Script	Cuando se selecciona se incluirá links creados de Java Script en la exploracion automatica
Businnes Procces File button	Abre un diálogo que permite escoger un proceso que ha sido salvado previamente

Forma de Propiedades (*Form Properties*). Cuando la caja de diálogo de "*Auto Fill Form*" es seleccionada, se llenan automáticamente formas encontradas en la etapa de Exploración. En orden para llenar las formas, se usan los valores de las formas que están definidas en la caja de diálogo de "*Form Properties*". Esta caja de diálogo se puede ver en vista simple o avanzada. En vista simple se despliegan el grupo del nombre y los valores. En vista avanzada se despliegan dos columnas extras, para cambiar de una vista a la otra, con el segundo botón se hace clic en la caja de diálogo y aparecen las dos opciones. La figura 3.24 muestra esta caja de diálogo en vista avanzada:

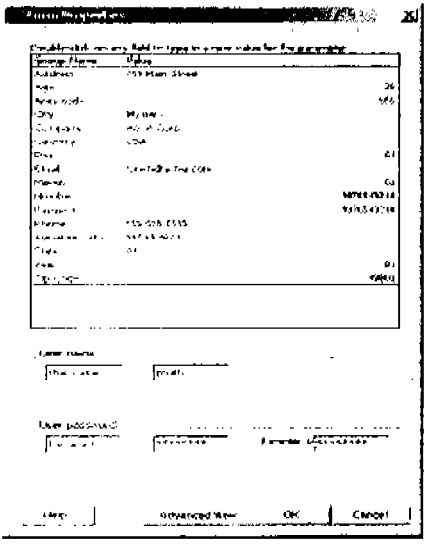


Figura 3.24. Forma de propiedades (**Sanctum, 2004**).

La Tabla siguiente explica los campos que aparecen en esta caja de diálogo:

Tabla 2. Form Properties Fields

Campo	Descripción
Group Name	El nombre que describe el contenido
Parameters	Los parametros de http que se buscaran para indentificar el grupo (Solo en Advance View)
Value	Los valores que se asignarán con este parametro
Match Type	Define que tipo de match quieres usar si el match completo o un match parcial Por ejemplo si seleccionas match completo para el parametro Address y el parametro definido es Add se someterá la dirección definida solo si se detecta el exacto "string" (Add) que aparece en el campo del parametro Country Si se escoge el match parcial se agregara el valor definido cuando encuentre cualquier "string" en la forma que incluya estos "string" (Add Addr y Address)

Cada celda en la tabla de la caja de diálogo de "*Form Properties*" puede ser modificada o borrada, incluso se puede añadir un nuevo campo; este nuevo campo se pondrá al final de la tabla.

Nombre del usuario y contraseña. Este es el campo más común e importante. Notar que el nombre del usuario y contraseña son parte del paso 4 en la etapa de *Setup*.

Definiendo los detalles de NTML. Para definir los detalles de NTML se deben de seguir los siguientes pasos:

1. En la etiqueta *Explore* (de la caja de diálogo de "*Scan Type Properties*), hacer clic en "*NTML Details*". La caja de diálogo de la autenticación de NTML aparece como se muestra en la figura 3.25:

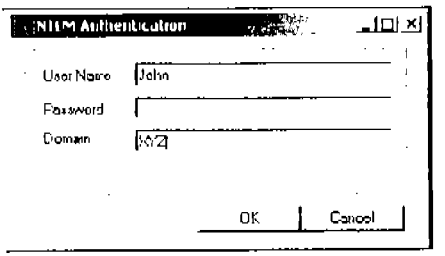


Figura 3.25. Autenticación de NTML **(Sanctum, 2004).**

2. Teclear un nombre de usuario, una contraseña y el dominio.
3. Hacer clic en OK.
4. Seleccionar la caja de diálogo de la autenticación de NTML.

Etiqueta *Explore Filters*.

En esta etiqueta se definen los filtros para una mayor rapidez y eficiencia. La siguiente Tabla explica los campos y propiedades que se presenta en esta etiqueta:

Tabla 3 The Explore - Filter Parameter Tab Properties

Artículo	Descripción
Limit Path	Al seleccionarlo no se tendra acceso al path más que el número de ves especificados
Limit Depth	Al seleccionarlo no se explorara links que se anidan "profundos" que el limte espicificado
Limit Number of Links	Al seleccionarlo la etapa de Site-Exploring esta siendo limitada por el número de links que se especifica
Restrict Exploring to these paths	Se restringe para explorar los paths indicados eso es útil para el desarrollador del "scan" para definir que parte de la aplicación será escaneada
Exclude these paths (Regular Expressions)	No se escaneará paths que emparejan las expresiones regulares entradas aquí

Tabla de archivos de extensión (*The File Extensión Table*). Esta tabla contiene una lista de los tipos de archivos que no serán tomados en cuenta durante la etapa de Explore, no serán incluidos los tipos de archivos seleccionados en el Scan. La lista contiene un número de tipos de archivos defectuosos y archivos de extensiones. También se puede escoger de remover un tipo de archivo y añadir uno nuevo.

Etiqueta *Test*.

Esta etiqueta es usada para configuraciones asociadas con la etapa de Prueba, como se muestra en la figura 3.26:

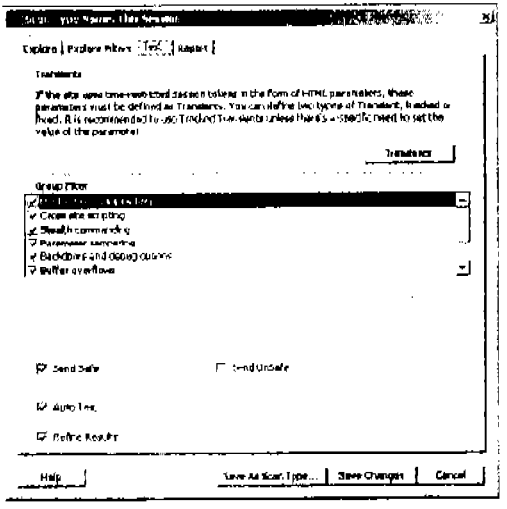


Figura 3.26. Etiqueta Test **(Sanctum, 2004).**

Transeúntes (*Transients*). Después que se explora un sitio, se puede escoger realizar las pruebas en un tiempo posterior. Consecuentemente, una cantidad significativa de tiempo puede pasar entre la etapa de Exploración y la etapa de

Pruebas, y entre diferente sesiones de escaneo. **AppScan DE** utiliza información almacenada en su base de datos durante la etapa de Prueba. Si el sitio utiliza muestras de identificación de sesiones con tiempo restringido (en la forma de parámetros de *cookies o* HTML), el sitio rechazará los pedidos que contienen muestras expiradas, así causará que las pruebas del sitio fallen. Por lo tanto todas muestras de sesiones con tiempo restringido en la forma de parámetros de HTML o *cookies*, se deben definir como Transeúntes.

Cuándo una muestra de sesión se define como un **AppScan DE** transitorio siempre lo asignará el valor más reciente disponible. Esto prevendrá la sesión "time out".

Durante la Exploración **AppScan DE** automáticamente detecta los parámetros de *cookies* y HTML que son probables de ser muestras de sesión, y las agregan a la lista de transeúntes. Transeúntes innecesarios se pueden borrar de esta lista antes de la etapa de la Prueba. Se pueden definir dos tipos de transeúntes, rastreado o fijo. Transeúntes fijos retienen un valor fijo. Para transeúntes rastreados, **AppScan DE** utiliza el valor más reciente encontrado dentro de la base de datos (al realizar una prueba). Se recomienda utilizar transeúntes rastreados a menos que hay una necesidad especifica de poner el valor del parámetro o un *cookie*. Notar que si se utiliza transeúntes rastreados, (cuando se sospecha que la muestra de la sesión del sitio se contuvo en la base de datos, puede haber expirado,) actualice la base de datos con un valor más reciente antes de probar el sitio.

Para actualizar la base de datos, visita simplemente la página en el sitio Web donde una muestra de sesión se manda (por ejemplo, una página de la entrada). Un clic en el botón de los Transeúntes abrirá la tabla Transitoria de administración, como se muestra en la figura 3.27:

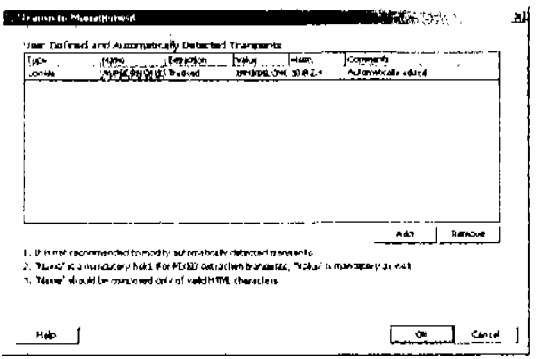


Figura 3.27. Tabla Transitoria **(Sanctum, 2004).**

La siguiente Tabla describe las propiedades de la tabla Transitoria de Administración:

Tabla 4: Transient Management

Campo	Descripción
Type.	Los tipos de Transient son. Cookie y HTML parametros.
Parameter Name.	Nombre del parametro o cookie.
Extraction.	Los tipos de extracciones de la Transient son: Traked. el valor del traked es actualizado por el más reciente valor que se encontró para este parametro o el cookie.
Value	Valor del parametro o del cookie (para extracciones tipo fixed)
Host	Se puede dejar vacío, si se deja vacío se utilizará los transients para todos los hots relevantes. Cuando un host específico es definido se utilizará los transients solo para el host definido.
Comments	Para comentarios del usuario.

Para añadir un Transeúnte se siguen los pasos siguientes:

1. En la pantalla de Propiedades de Tipo de Scan, hacer clic en la etiqueta de Prueba después hacer clic en Transeúntes. Se abre la tabla Transitoria de Administración como se ve en la figura anterior.
2. Hacer clic en *Add*. Una fila nueva se añade a los Transeúntes.
3. Llenar los campos para el Transeúnte nuevo, como se refiere en la tabla anterior.

4. Hacer clic en *OK*. El Transeúnte nuevo se agrega.

Para modificar un Transeúnte se siguen los pasos siguientes:

1. Editar los campos en la tabla de Transeúntes.

Para borrar un Transeúnte se siguen los pasos siguientes:

1. De la tabla Transitoria de Administración, hacer clic en el record o registro transitorio que se quiere remover.
2. Hacer clic en *Remove*. El Transeúnte se borra de la lista.

Lista del Grupo de Filtros (*Group Filter List*). Durante la etapa de Prueba se corren una serie de pruebas para checar y analizar la seguridad del sitio. En el grupo hay nueve categorías, cuando se selecciona una de estas categorías está será incluida en la etapa de Prueba. El Área del grupo de filtros se muestra en la figura anterior y los filtros son explicados en la siguiente tabla, al igual que estas categorías son explicadas y mejor detalladas en el capítulo 4.

Tabla 5 Group Filter Checkboxes

Nombre del Checkbox	Descripción
Hidden field manipulation	Modificación de los campos de formas ocultos permiten el daño de datos que alcanza la aplicación web
Cross site scripting	Inserta lenguajes encriptados en campos de textos para así mostrar información a otros usuarios
Stealth commanding	Plantear Caballos de Troyan en un campo de texto haciendo que la aplicación web realice órdenes que no debo hacer
Parameter tampering	Modificación de parámetros que son parte del URL
Backdoor and debug options	Explotación de vulnerabilidades que se dejaron abiertas durante la etapa de desarrollo del código del sitio web
Buffer overflows	Sobrecargar el sitio web con una simple petición
Cookie Poisoning	Modificación de archivos de cookies para tener acceso a información sensible o realizar actividades a favor un usuario diferente
Suspicious Content	Durante la etapa de Test se puede también buscar contenido sospechoso

Filtros Adicionales. Cada una de estas categorías de pruebas incluye un rango de prueba que puede ser mostrado en el sitio. En el área de filtros adicionales se define que tipo de prueba en cada categoría será mandado. El área de filtros adicionales se muestra en la figura anterior y la siguiente tabla explica sus funciones:

Tabla 6 Additional Filter Checkboxes

Campo	Descripción
Send Safe Test requests	Al seleccionarlo se mandará solo las peticiones seguras de la prueba las que no pueden causar daño al sitio web
Send Unsafe Test requests	Al seleccionarlo mandará las peticiones inseguras las que pueden causar daño al sitio web
Auto Test	Al seleccionarlo no avanzara automaticamnete a la siguiente etapa
Refine Results	Al seleccionarlo se realizará procesos internos del refinamiento de resultados para aumentar la certeza del resultado

Etiqueta *Report*.

Esta etiqueta es usada para filtrar el contenido que aparecerá en el Reporte del Resultado del Scan.

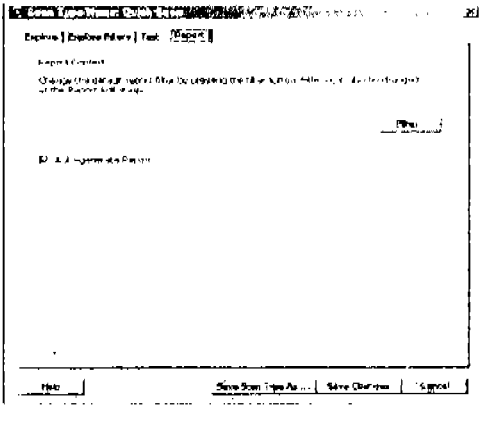


Figura 3.28. Etiqueta Report **(Sanctum, 2004).**

Cuando se selecciona la opción de generar el reporte automáticamente, **AppScan DE** genera de manera automática un reporte del Scan después que la etapa de Prueba esta completa.

Para filtra el contenido del Reporte:

1. En la etiqueta Reporte, hacer clic en "*Filter*". Aparece una caja de diálogo con la configuración del contenido del Reporte:

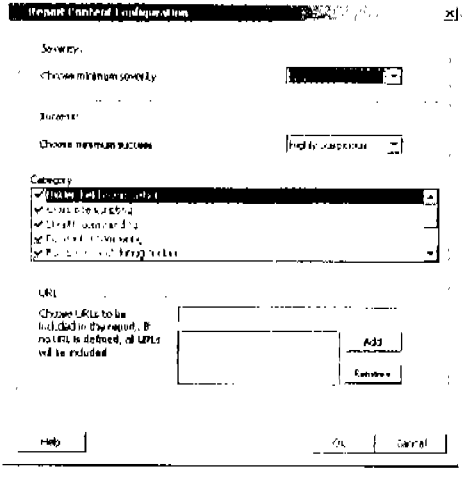


Figura 3.29. Configuración del contenido del reporte **(Sanctum, 2004).**

2. Del Área de "*Severity*", seleccionar el nivel de severidad que se quiere obtener (*High/ Medium/Low*).

3. Del Área de "*Success*", seleccionar el nivel de los sucesos que se quiere obtener

(*Vulnerable/ Highly Suspicious/ Not Vulnerable*).

4. Del Área de las categorías, limpiar alguna categoría que se quiere excluir del reporte.

5. Si se quiere que en el reporte se incluya solo los resultados de un URL específico, entonces especificar el URL en la caja de texto y hacer clic en "*Add*".

Salvar los cambios en el tipo de Scan. Después de definir la configuración en el tipo de Scan, se puede salvar los cambios de la sesión en uso; esto se hace, haciendo clic en el botón de "*Save Changes*". También se puede salvar la configuración permanentemente bajo un nuevo nombre haciendo clic en el botón de "*Save Scan Type As*".

c) Aplicación.

Application: se define la aplicación que se quiere scanear y el punto de partida que el proceso de Scan realizará.

Definir el servidor o servidores que serán escáneados. Sobre el título de "*type the server to be scanner*". Por ejemplo: **http://www.sanctuminc.com/**.

Para añadir servidores a la lista se siguen los pasos siguientes:

1. Seleccionar "*Additional Server*".
2. En la caja de texto de "*Additional Server*", poner el servidor adicional que se quiere agregar.
3. Hacer clic en "*Add*".

Para remover servidores de la lista se hace lo siguiente:

1. De la lista de servidores, seleccionar el servidor que se quiere remover.
2. Hacer clic en "*Remove*".

Definir el punto de partida del Scan. Hay dos maneras para definir el punto partida:

- La primera es muy simple, teclear la dirección en la caja de texto de "*server to be scanned*".
- La segunda opción es manual, esto se hace de la siguiente manera:

1. En el panel de instrucciones hacer clic en "*Show*".

La página de punto de partida aparece en el área de administración de **AppScan DE**, como se muestra en la figura 3.30:

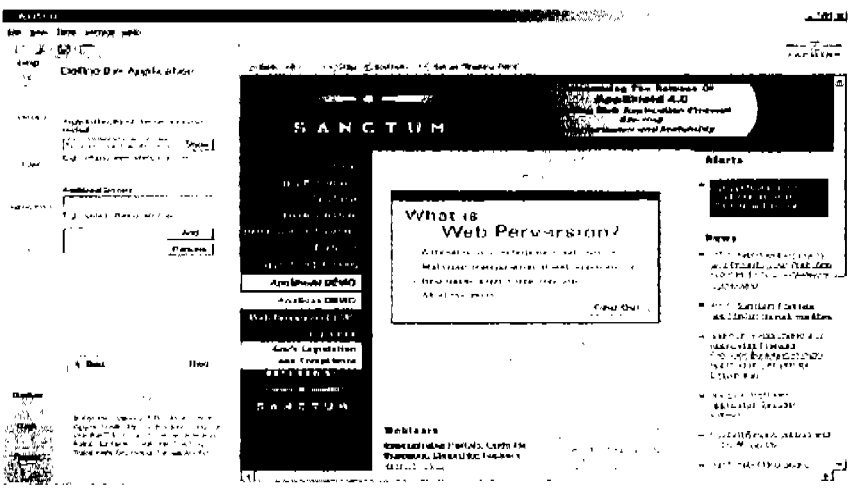


Figura 3.30. Página del punto de partida **(Sanctum, 2004).**

2. Usar la barra de navegación del área de administración para examinar el sitio y localizar el punto de partida del Scan.

3. Hacer clic en Set As Starting Point para mandar la página en uso como el punto de partida.

d) Entrada.

Input: provee a **AppScan DE** con la información del usuario, tales como, nombre del usuario y contraseña o acceder a la caja de diálogo de "*Form Properties*", antes de proceder a la etapa de Pruebas. Esto puede ser requerido para ciertas áreas en la etapa de pruebas.

Hay una opción avanzada que permite seleccionar una sesión de ingreso manualmente para casos donde la sesión de ingreso automática puede no ser suficiente, esto lo veremos más adelante. En la figura 3.31 se muestra la sesión de ingreso:

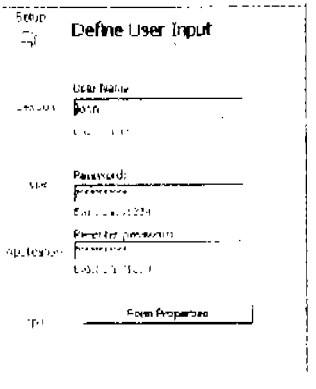


Figura 3.31. Datos de ingreso **(Sanctum, 2004).**

Opciones de la Sesión de Ingreso. Durante la etapa de Exploración **Pazcan DE** identifica el proceso de ingreso. El botón de "*Session Login button*", aparece en la barra de herramientas del examinador durante la sesión de ingreso o "*Input*".

Esta opción permite cambiar el ingreso de datos de Automático a Manual o No ingresar datos. La figura 3.32 muestra estas opciones, al cual se accede haciendo clic en el botón de "*Session Log*":

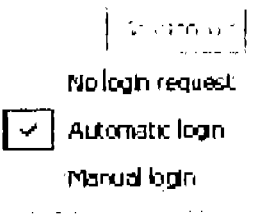
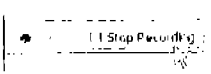


Figura 3.32. Opciones de la sesión de ingreso (**Sanctum, 2004**).

Ingresar Manualmente. Se selecciona ingresar manualmente si se quiere hacer peticiones de ingreso para grabaciones manuales, que **AppScan DE** usará para ingresar al sitio instantáneamente a la petición de la grabación automáticamente durante la etapa de Exploración. Para grabar una petición de ingreso manual se siguen los siguientes pasos:

1. Seleccionar "*Manual Login*" de la lista del botón de "*Session Log*", el examinador se abre y va al punto de partida.
2. Examinar manualmente hasta llegar a la página de ingreso.
3. Apretar
4. Llenar los campos de ingreso como sea necesario. Si el ingreso envuelve mas de una forma, ir a cada forma (usando el examinador) y llenar los datos (manualmente) antes de para la grabación
5. Una vez que se han llenado todas las formas requeridas, apretar 

Los datos de ingresos grabados serán usados (automáticamente) para ingresar dentro del sitio para la etapa de Prueba.

3.6.2. Etapa de Exploración.

La primera parte en el proceso de **AppScan DE** es la Exploración. **AppScan DE** explora el sitio y construye un modelo de las aplicaciones que se corrieron en él. También crea una lista de las vulnerabilidades potenciales, basándose en las vulnerabilidades potenciales identificadas, **AppScan DE** crea Pruebas para verificar las vulnerabilidades actuales del sitio. La etapa de Exploración esta subdividida en dos partes:

a) Correr.

Run: manda peticiones a la aplicación (*"Explore the Site"*) y colecciona las respuestas de esas peticiones. Esto se puede hacer automáticamente, o manualmente, incluso se puede escoger una de las aplicaciones del proceso. Basado en las respuestas que recibe de la aplicación, se prepara una lista de peticiones de Pruebas, las cuales serán mandadas durante la etapa de Prueba. Cuando se "corre" el proceso de Exploración, **AppScan DE** inicia el Scan de la "Página inicial" que se escogió durante la etapa de *Setup*. De ahí, se explora el sitio metódicamente visitando cada liga o *"link"* con la aplicación hasta que haya visitado cada liga con la aplicación. La exploración automática en mucho más rápida que la exploración manual, ya que así, se pueden visitar miles de ligas en segundos. Es importante notar que el criterio que se dio en la etapa de *Setup* define los límites del proceso de Exploración. Por ejemplo, si el límite que se dio fue de 5 ligas, no se visitarán más que 5 ligas iniciando del punto de partida.

Peticiones de Exploración que abren una pantalla. El panel de instrucciones de la petición inicial abre una pantalla con botones de inicio, parar y repetir el proceso de Exploración. Incluso muestra la configuración del punto de partida de la Exploración. La información desplegada en el área de administración de resultados depende del tipo de Scan que se selecciono en la etapa de *Setup*.

La Exploración puede darse en Automático, Manual o en el modo de proceso. Una vez que la Exploración dio inicio, para cambiar a un modo diferente de exploración, primero se necesita para la exploración en curso, y luego seleccionar el nuevo modo de exploración.

Exploración Automática. Una vez que el Scan fue propiamente configurado e iniciado a correr en el modo Automático, todo lo que se tiene que hacer es iniciar el Scan y dejar que **AppScan DE** haga el resto.

Como la etapa de Exploración progresa, el contador en la tabla del progreso de la Exploración indica que tanto ha progresado. Como se muestra en la figura 3.33:

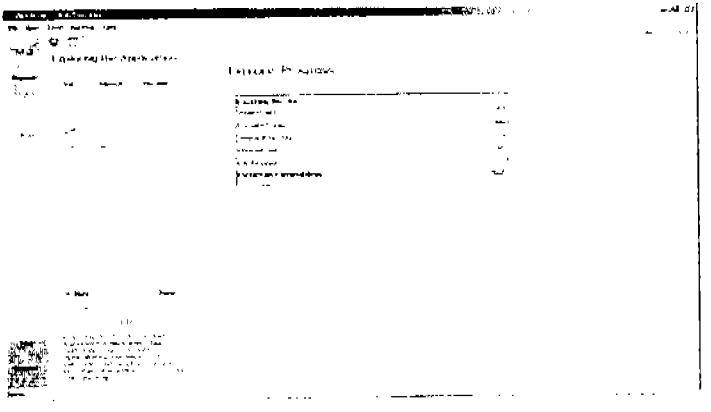


Figura 3.33. Exploración automática **(Sanctum, 2004).**

Para iniciar la Exploración Automática se deben seguir los siguientes pasos:

1. Para proceder en la etapa de Exploración (habiendo completado la etapa de *Setup*) presionar ***Next***.
2. Presionar ***Run***

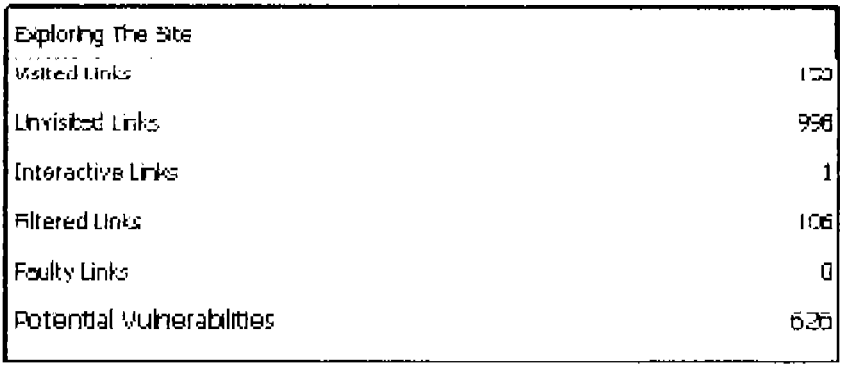


Figura 3.34. Tabla de progreso **(Sanctum, 2004).**

El contador en la tabla del progreso de la Exploración crece cuando las peticiones iniciales son mandadas.

3. Espera hasta que el contador deja de crecer y el botón de ***Next*** se vuelve activo, esto indica que el proceso de Exploración esta completo, o para parar la Exploración antes de que este completo presionar ***Stop.***

4. Para proceder con los resultados iniciales, presionar ***Next*** en el panel de instrucciones. Una vez que Exploración esta completa se inicia automáticamente con la etapa de Pruebas.

La tabla del progreso de la Exploración. Cuando la Exploración esta completa, la tabla de progreso de la Exploración resume las exploraciones que se hicieron. La figura anterior muestra esta tabla, y la siguiente tabla muestra como se explica cada campo de la tabla de progreso de la Exploración.

Tabla 7 Explore Progress table summary

Artículo	Descripción
Visited links	Número de links que se han visitado hasta ahora
Unvisited links	Número de links que se tienen que visitar durante el proceso de Exploring
Interactive links	Numero de links que requieren alguna entrada que no se puede llenar automaticamente
Filtered links	Numero de links que no se exploraron porque ellos fueron filtrados fuera de la linea de la etapa de Explore o por un defecto del filtro
Faulty links	Número de links totales que no respondieron a la petición inicial
Potential vulnerabilities	Peticiones que fueron indentificadas como vulnerabilidades potencia es durante el proceso de Explore y que serán probadas durante la etapa de Test

Exploración Manual. Si se desea escánear el sitio manualmente, hacer clic en *"Manual"* para abrir el examinador de **AppScan DE** en la ventana de administración. Como se muestra en la figura 3.35:

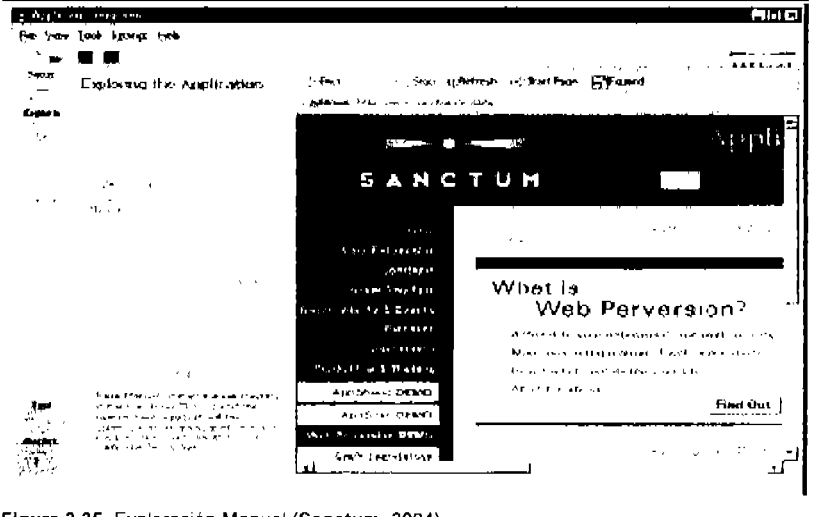


Figura 3.35. Exploración Manual **(Sanctum, 2004).**

La siguiente Tabla explica los botones del examinador:

Tabla 8 AppScan Browser buttons

Botón.	Descripción
Back	Pagina anterior
Next	Pagina Siguiente
Expand / Collapse (toggle)	Muestra el Browser como una ventana separada (muestra el progreso de la tabla de la etapa de Explore en el area de la Administración: Muestra el Browser en el area de la Administración en vez del progreso de la tabla de la etapa de Explore).
Stop	Para de mandar la peticion actua
Refresh	Refresca la pagina
Start Page	Regresa al punto de inicio de la página
Auto Scan	Cierra el Browser e inicia automaticamente la etapa de Explore
Finish	Cierra el Browser

Para la Exploración Manual, se deben de seguir los pasos siguientes:

1. Usar el examinador de **AppScan DE** para hacer clic en las ligas que se quieren visitar. Por cada petición mandada, automáticamente se clasifican las respuestas y se generan peticiones de Prueba. Notar que una vez que el la exploración ha iniciado, para cambiar a un modo diferente primero se debe parar la exploración, y luego seleccionar un nuevo modo de exploración.
2. Para cerrar el examinador, hacer clic en ***Finas*** en la barra de herramientas del examinador.
3. Para proceder con los resultados iniciales, hacer clic en ***Next*** en el panel de instrucciones.

Proceso del Negocio (*Business Proccess).* Una tercera opción para escánear tu sitio, es escánear una específica transacción o proceso dentro de la aplicación. Para crear un archivo *".bps"* para una porción deseada de la aplicación, se siguen el paso siguiente:

1. En la etapa de Exploración, seleccionar la etiqueta de ***Proccess*** (en el panel de instrucciones).

2. Si se desea conjuntar un punto de partida diferente, hacer clic en *"browser"* en la etiqueta de proceso. El examinador aparece mostrando la aplicación seleccionada. (Opcional).

3. Manualmente explora el sitio para el punto de partida deseada. (Opcional).

4. En la etiqueta de proceso hacer clic en .

5. Manualmente explora todas las secciones relevantes del sitio.

6. En la etiqueta de proceso hacer clic en para parar la fase de grabación.

Una lista de las peticiones grabadas es presentada, como se muestra en la figura 3.36:

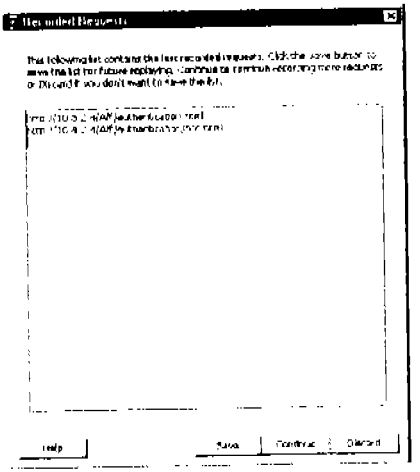


Figura 3.36. Lista de peticiones grabadas **(Sanctum, 2004).**

7. Hacer clic en ***Save*** para salvar la exploración grabada. Clic en ***Continue*** para continuar grabando, o ***Discard*** para cancelar cualquier cambio o reiniciar la sesión de los resultados de la exploración.

8. En la caja de diálogo de *"Save recorded data"*, teclear el nombre del archivo para el proceso grabado (.bps) y dar salvar.
La exploración grabada esta ahora salvada como archivo de proceso en el fólder de ***Business Proccess.*** Notar que las ligas visitadas durante este proceso de exploración son añadidos como ligas visitadas manualmente de la sesión en uso.

Para Explorar un proceso se deben seguir los pasos siguientes:

1. Cuando se esta en la etapa de Exploración, seleccionar la etiqueta de ***Proccess*** (localizada en el panel de instrucciones).
2. En la etiqueta de ***Proccess***, hacer clic en [...]. Cuando se abre la caja de diálogo del Proceso, aparece en la figura 3.37:

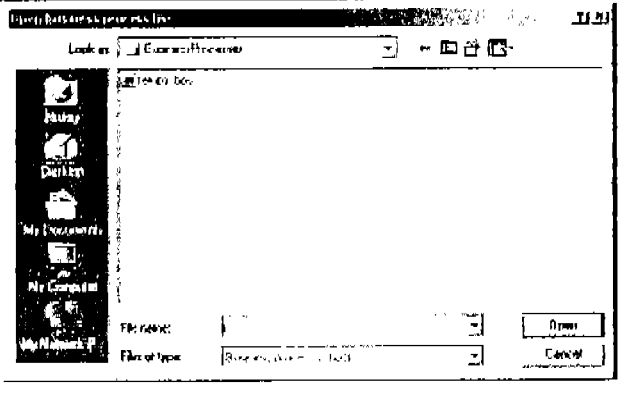


Figura 3.37 Abrir un proceso [Sanctum, 2004]

3. Seleccionar el proceso que se desea bajar.
4. Hacer clic en ***Open***, el proceso seleccionado ha sido bajado.

5. En la etiqueta de ***Proccess***, clic en [...]. **AppScan DE** explora el proceso seleccionado.

Conflicto con el *Host*. Si se encuentra en una situación de conflicto con el *Host*, se escoge bajar un proceso salvado en la etapa de Exploración, y el *Host* configurado no está definido en el proceso que se selecciono, se puede hacer:

1. Ignorar el *Host* salvado.
2. Añadir el *Host* grabado a la lista de servidores de la exploración.
3. Durante la exploración, reemplazar el *Host* salvado con el *Host* que está en uso.

b) Resultados.

Results*:** los resultados de la etapa de Exploración son una manera de evaluar para determinar si el escáneo fue preciso y comprehensivo como se quiso. La página de los resultados de la Exploración es una versión interactiva de la tabla final del Progreso de la Exploración (con la excepción de las ligas no visitadas, lo cual no es relevante en los resultados). Da la oportunidad de examinar las peticiones que fueron mandadas, las peticiones que no fueron mandadas, y cualquier parámetro que fue mandado con la petición (u otras ligas interactivas). Si es necesario se puede mandar o volver a mandar peticiones antes de la etapa de Prueba. Una reexaminación de las vulnerabilidades potenciales encontradas durante los resultados en la etapa de Exploración es recomendada, porque estas serán probadas durante la etapa de Pruebas. Cuando se examina los resultados de la Exploración, se puede hacer clic en ***Analysis Category para más información sobre los artículos actuales en esa categoría. Para más información sobre cualquier artículo hacer clic en la lista, hacer clic en ese artículo. En la figura 3.38 se muestra la página de los resultados de la Exploración:

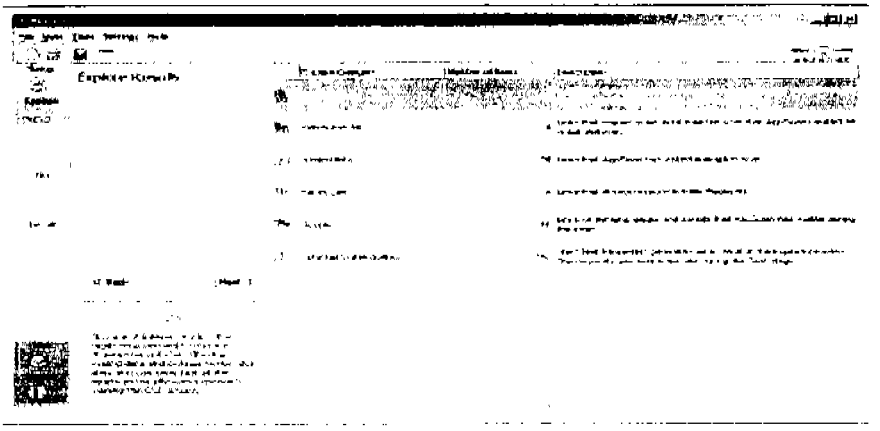


Figura 3.38. Resultados de la exploración **(Sanctum, 2004).**

Ligas Filtradas. La lista de ligas filtradas muestra las ligas que no fueron visitadas porque fueron filtradas fuera. Se puede examinar las ligas de esta lista y si se desea se puede hacer manualmente. Los valores de la tabla de los resultados en la Exploración serán actualizados acordemente. Para explorar las ligas filtradas se siguen los siguientes pasos:

1. En la lista de los resultados de la Exploración, hacer clic en ***Filtered Links.*** La lista completa de las ligas filtradas aparece así:

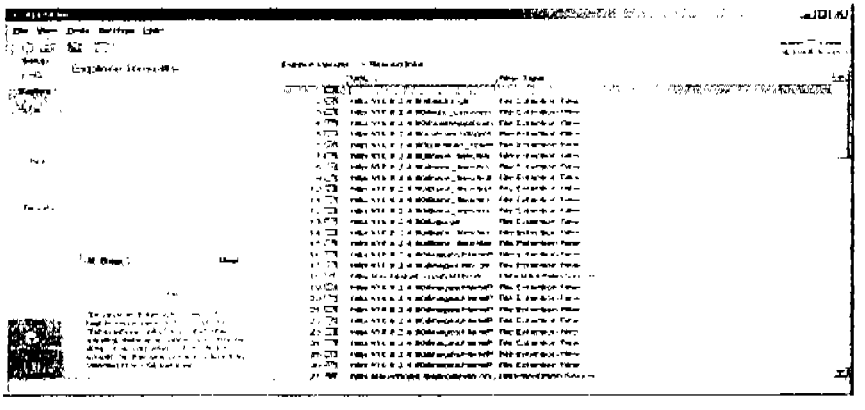


Figura 3.39. Ligas filtradas **(Sanctum, 2004).**

Hacer clic en la liga que se quiere visitar. El examinador se abre, y se manda la petición.
2. Para regresar a los resultados de la Exploración, clic en ***Explore Results*** en la barra de navegación. Los resultados de la Exploración están actualizados para ser incluidas las peticiones mandadas.

Ligas Interactivas. La lista de las ligas interactivas muestra las peticiones que no fueron mandadas, porque ellas requieren entradas del usuario que no se hicieron durante la etapa de *Setup*. Exactamente cuales ligas se clasifican como "Interactivas" depende del tipo de Scan que se selecciono durante la etapa de *Setup*. Se puede examinar las ligas interactivas, y si se quiere se puede requerir la información del usuario y mandarla manualmente. Los números de la tabla de resultados de la Exploración serán actualizados acordemente. Se recomienda que se examinen muy bien la lista de las ligas interactivas, llenar los datos requeridos y mandar esas peticiones. Entonces se incluirá esas ligas durante la fase de prueba. Después de visitar las ligas interactivas se puede continuar con el proceso de Exploración. Incluso si previamente se completo el proceso de Exploración, nuevas ligas serán añadidas y el botón de Exploración se vuelve activo otra vez. Para explorar una liga interactiva se siguen los pasos siguientes:

1. En la lista de Resultados de la Exploración, hacer clic en ***Interactive Links.*** La página de la lista completa con las ligas interactivas aparece.
2. Hacer clic en el URL requerido. El examinador abre la página seleccionada, como se muestra en la figura 3.40:

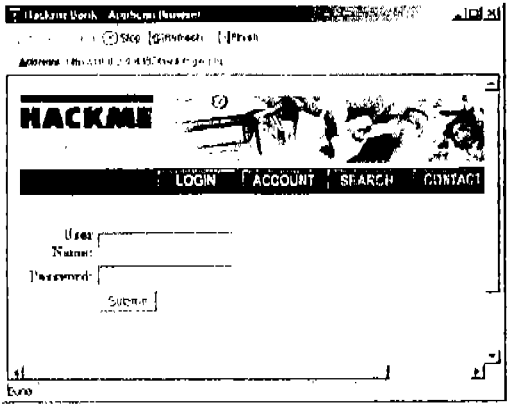


Figura 3.40. Página de la lista completa con las ligas interactivas **[Sanctum, 2004].**

3. Para cerrar el examinador, hacer clic en ***Finish.***

4. Para regresar a los resultados de la Exploración, hacer clic en ***Explore Results*** en la barra de navegación, en la parte de arriba en el área de administración. Los resultados de la Exploración se actualizaron para incluir las formas mandadas.

Notar que desde que la liga interactiva se mandó, puede haber abierto una nueva parte del sitio, es muy recomendable que se resuma y complete el proceso de la Exploración después de haber mandado una liga interactiva.

Ligas Visitadas. Las ligas visitadas son peticiones por las cuales **AppScan DE** recibe una repuesta inicial válida. Basada en esas respuestas se generan las peticiones de Prueba, peticiones diseñadas para revelar las debilidades en el sitio, que serán mandadas durante la etapa de Pruebas. Si un número pequeño de ligas o páginas fueron cambiadas en el sitio después del Scan, tal vez se quiera volver a mandar algunas peticiones manualmente, en vez de volver a correr todo el proceso de la Exploración de nuevo. Para explorar las ligas visitadas se debe seguir los pasos siguientes:

1. En la lista de resultados de la Exploración, clic en ***Visited Links.*** La lista de ligas visitadas aparece, y se puede volver a mandar cualquier petición de nuevo haciendo clic en ella.
2. Para regresar a los resultados de la Exploración, clic en ***Explore Results*** en la barra de navegación en la parte de arriba en el área de administración.

Ligas No Válidas. Las ligas no válidas o ***Faulty Links*** son esas peticiones donde la petición fue mandada pero no hay una respuesta válida. Usualmente es un problema de comunicación, ya sea que el sitio este sin red, o cuando la liga se ha roto (ligas con una página no existente). Si una liga importante es listada como no válida, se puede volver a mandar la petición manualmente, en vez de repetir todo el proceso. Para volver a mandar una liga no válida se siguen estos pasos:

1. En la lista de resultados de la Exploración, clic en ***Faulty Links.***

La lista de ligas no válidas aparece. Luego para cada URL, **AppScan DE** los lista como no válidos, como se muestra en la figura 3.41:

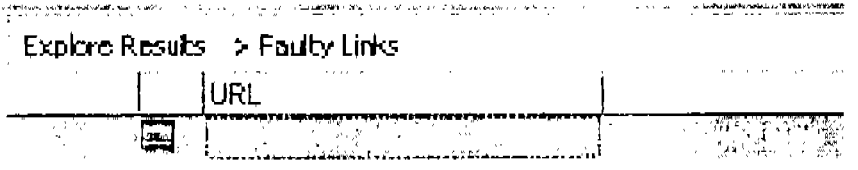


Figura 3.41. Lista de ligas no válidas **(Sanctum, 2004).**

2. Para volver a intentar con una liga, hacer clic en ella. El examinador de **AppScan DE** se abre y manda la petición seleccionada. Se puede continuar con la Exploración de esta manera como se desee, todas la peticiones mandadas son añadidas a los resultados de la Exploración.
3. Para regresar a los resultados de la Exploración, clic en ***Explore Results***. Los resultados de la Exploración se actualizaron para incluir las peticiones mandadas.

Scripts. Se lista un *Scripts* todas las peticiones iniciales que incluye uno o más parámetros. Estos son las ligas más vulnerables para tener un ataque informático. Para cada petición se puede examinar los parámetros de nombre, valor (es) y tipo.

Para examinar un *Script* se siguen estos pasos:

1. En la lista de resultados de la Exploración, clic en ***Script***. La lista de ligas *Script* aparece, y con cada URL están los parámetros de nombre, valor (es) y tipo, como se muestra en la figura 3.42:

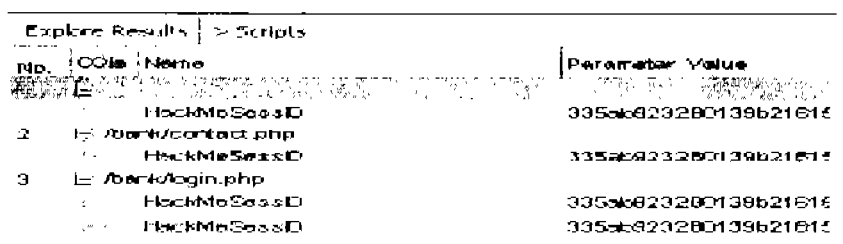


Figura 3.42. Lista de resultados de la exploración **(Sanctum, 2004).**

Notar que hay más de un parámetro para cada URL, los parámetros son listados uno seguido del otro. Donde hay más de un valor para un particular parámetro de nombre, los valores son separados por comas.

2. Para regresar a los resultados de la Exploración, clic en ***Explore Results***. Los resultados de la Exploración se actualizaron para incluir las peticiones mandadas

Vulnerabilidades Potenciales. El artículo final en la tabla de resultados de la Exploración es más que un conjunto de resultados, contiene las peticiones propuestas para la Prueba, que **AppScan DE** generó basado en los resultados de la Exploración. Se puede observar el número de las Vulnerabilidades Potenciales generadas en cada categoría, como mirar cada petición de Prueba como URL. Para examinar las vulnerabilidades potenciales se siguen estos pasos:

1. En la lista de resultados de la Exploración, clic en ***Potential Vulnerabilities.*** La lista de vulnerabilidades potenciales provee un resumen con las ligas para detallar sobre las vulnerabilidades potenciales detectadas como se mostrará en la figura siguiente. Luego para cada categoría que esta sobre la columna de *Test Type* es el número total de vulnerabilidades potenciales en esa categoría. A la derecha de los totales están los números de peticiones las cuales están en un nivel de *High (H), Medium (M) and Low (L) Severity.* Como se ve en la figura 3.43:

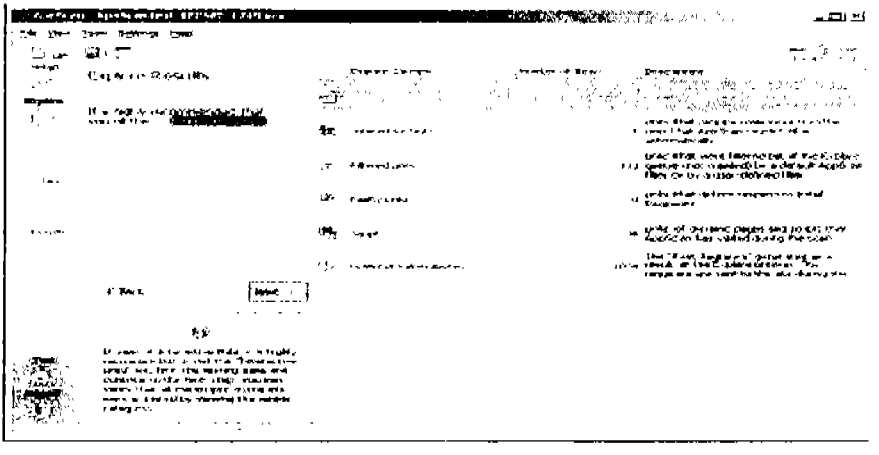


Figura 3.43. Vulnerabilidades potenciales **(Sanctum, 2004).**

2 Para examinar una vulnerabilidad potencial especifica en una categoría, clic en la Categoría, como se muestra en la figura 3.44:

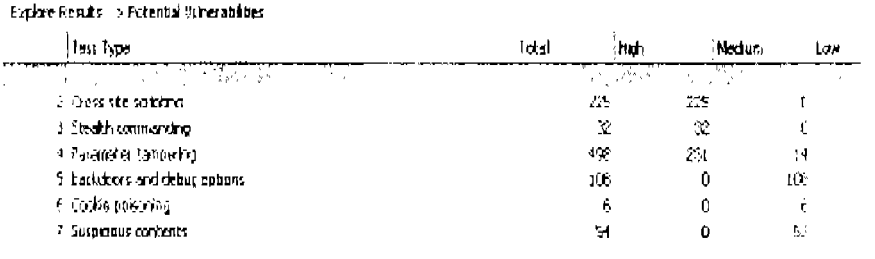


Figura 3.44. Examinar una vulnerabilidad potencial **(Sanctum, 2004).**

3. Para regresar a la lista de categorías, clic en ***Potential Vulnerabilities***.

4. Para regresar a la tabla principal de resultados de la Exploración, clic ***Explore Results.***

3.6.3. Etapa de Prueba.

La segunda parte en el proceso de **AppScan DE** es la etapa de Prueba, en la cual se usa la información generada durante la etapa de Exploración para probar el sitio. La etapa de Prueba esta subdivida en tres partes:

a) Prueba.

Test: primero **AppScan DE** manda varias (Pre-Pruebas) peticiones al sitio (tales como, peticiones de ingreso), para mejorar la eficiencia de la Prueba. Luego da inicio a mandar peticiones de las Pruebas, las cuales están diseñadas a revelar los riesgos de seguridad en el sitio.

Correr la Prueba. La pantalla inicial de las peticiones de Prueba muestra una tabla de categorías. Luego para cada categoría se puede ver el número de las Pruebas propuestas en esa categoría, como se muestra en la figura 3.45:

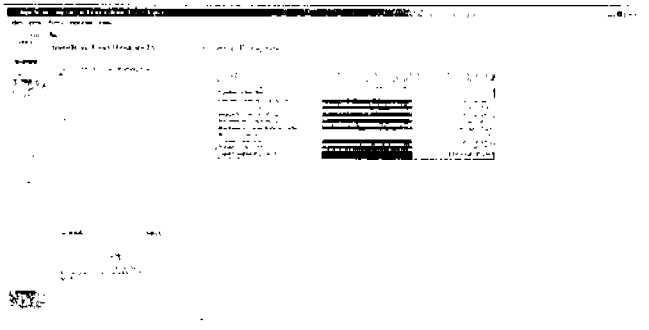


Figura 3.45. Correr la prueba **(Sanctum, 2004).**

Cuando se presiona el botón de ***Test,* AppScan DE** hace lo siguiente:

- Ingresa a donde sea en la aplicación que requiera autenticación.
- Realiza varias pruebas preliminares en los URL´s que ayuda a interpretar los resultados de la Prueba.
- Iniciar a probar URL´s en el sitio mandando las peticiones de la Prueba *"Test Requests"* (peticiones diseñadas a revelar las vulnerabilidades actuales) y grabando las respuestas del sitio para cada petición. (Las peticiones de la Prueba fueron creadas durante la etapa de Exploración, basándose en las vulnerabilidades potenciales descubiertas).

Para iniciar a probar el sitio, se deben seguir los pasos siguientes:

1. Hacer clic en el botón de ***Test*** .

2. Para examinar los resultados hacer clic en (Next).

La pantalla interactiva de resultados aparece, con un resumen de todas las pruebas, agrupadas por Resultados, como se muestra en la figura 3.46:

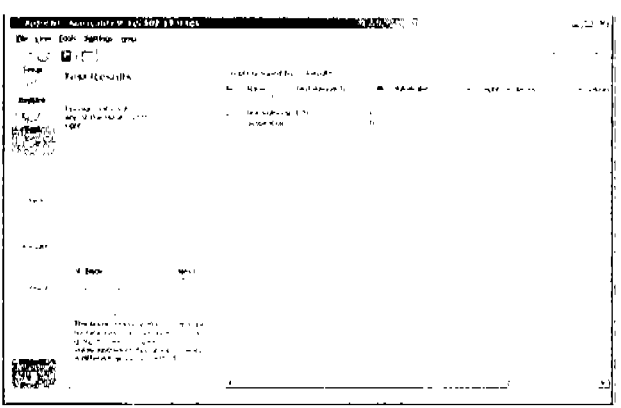


Figura 3.46. Resultados de la prueba **(Sanctum, 2004).**

Usar la lista de las Pruebas Agrupadas para seleccionar un método de un grupo diferente, como se ve en la figura 3.47:

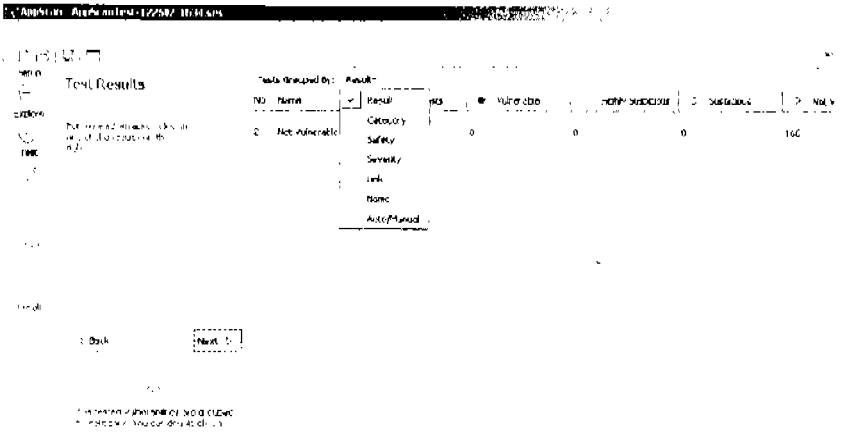


Figura 3.47. Lista de las pruebas agrupadas **(Sanctum, 2004).**

3. Hacer clic en un artículo de la tabla para ver todas las Pruebas que pertenecen a ese grupo.

b) Resultados.

Results: después que las pruebas están completas se puede ver y analizar los resultados en varios grupos. Hay numerosas opciones disponibles en esta fase, por ejemplo. se puede observar pruebas individuales y consejos que explican el problema descubierto, los cuales se pueden editar para imprimir un Reporte. Incluso se puede comparar los resultados de la Prueba del sitio con los resultados recibidos durante la etapa de Exploración. Los resultados se pueden observar en tres niveles.

• Nivel 1: Un resumen de todos los resultados, agrupados de acuerdo con el método de agrupación que se seleccionó, es desplegado. Solo el número de resultados por grupo es mostrado, como se muestra en la figura 3.48, no los resultados actuales.

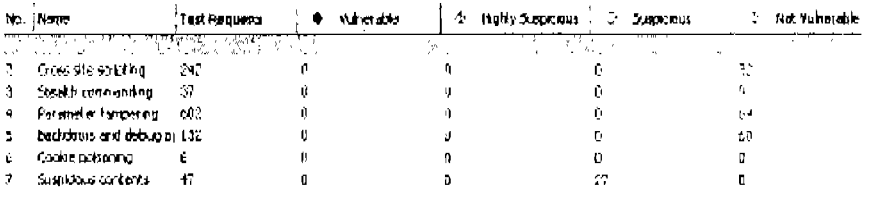


Figura 3.48. Resumen de resultados **(Sanctum, 2004).**

• Nivel 2: Todos los resultados con un grupo seleccionado es desplegado, como se muestra en la figura 3.49:

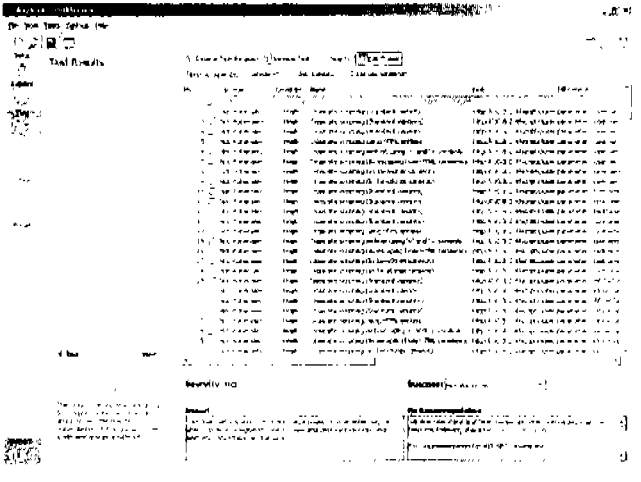


Figura 3.49. Resultados con un grupo seleccionado **[Sanctum, 2004].**

• Nivel 3: Resultados Detallados. La "tarjeta índice" para un resultado individual es desplegado, como se muestra en la figura 3.50:

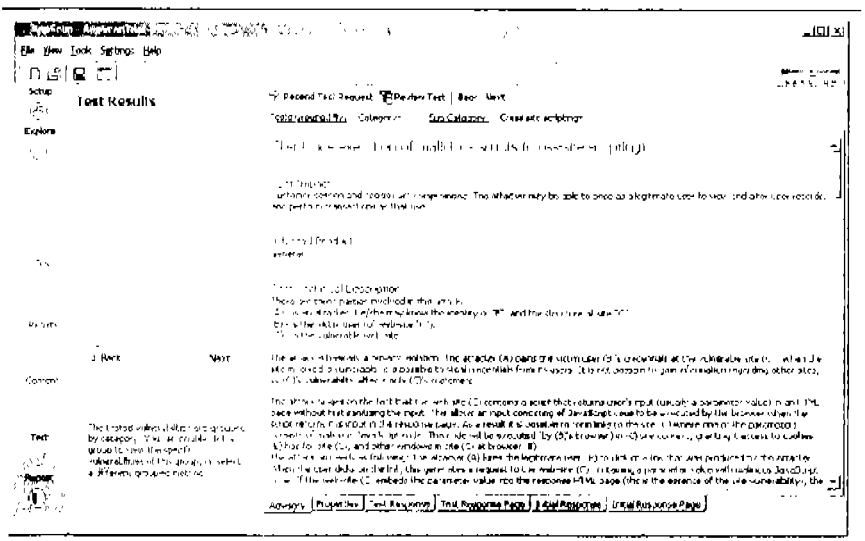


Figura 3.50. Resultados detallados **(Sanctum, 2004)**.

Métodos Agrupados. La página de los resultados de la Prueba se abre por default con una lista de nivel 1 de los resultados de la prueba agrupados por Resultado. El menú de ***Grouping Methods*** deja seleccionar diferentes métodos agrupados para los resultados de la Prueba, como se muestra en la siguiente figura 3.51:

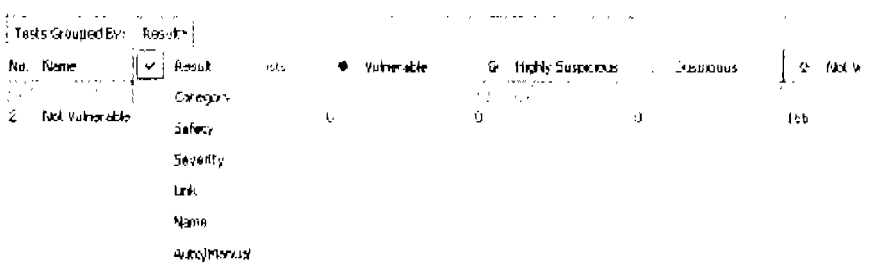


Figura 4

3.51. Menú de métodos agrupados **(Sanctum, 2004)**.

Estos métodos de búsqueda están explicados en la Tabla que se muestra a continuación:

Tabla 9 Grouping methods summary

Agrupado por	Nombre del Grupo	Explicación
	None	Las Pruebas planeadas que no se mandaron
	Not vulnerable	Las pruebas a que el sitio ciertamente casi no es vulnerable
Result	Suspicious	Pruebas a que el sitio puede ser vulnerable
	Highly suspicious	Pruebas a que el sitio es probablemente vulnerable
	Vulnerable	Pruebas a que el sitio es casi seguro vulnerable
Category		Son los mismos elementos explicados en la tabla 5
Safety	Safe	Ningún peligro de dañar al sitio
	Not safe	Puede probablemente dañar al sitio
Severity	High Medium Low	La calificacion de la Severidad se relaciona con la Seguridad
Link		Artículos que se dieron por el URL principal
Name		Artículos que se dieron por el nombre del consultor

Resultado del Color del Código. Cuando se selecciona un método para buscar información, **AppScan DE** crea una tabla en el cual cada grupo en un renglón. En cada renglón hay una columna que muestra el número total de Pruebas en el grupo, seguido por cuatro columnas de color de código. Estas columnas muestra el número de respuestas de la Prueba en el grupo en cual indica que una liga es: *Vulnerable, Highly suspicious, Suspicious, or Not vulnerable.*

Se debe notar que la columna de Peticiones de las Pruebas contiene la suma de las 4 columnas de color, porque incluye las peticiones de las pruebas que no fueron mandadas. Como se muestra en la figura 3.52:

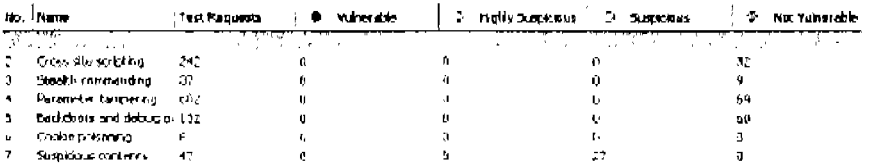


Figura 3.52. Columna de peticiones de las pruebas **(Sanctum, 2004)**.

Examinando los Resultados con un Grupo. Cuando se ha seleccionado un método de búsqueda, se puede examinar los artículos con un grupo (nivel 2). Por ejemplo, si se selecciono *Category* como el método de búsqueda, una lista de categorías se despliega. Luego para cada categoría aparece el número de vulnerabilidades en esa categoría. Se puede abrir la lista completa de resultados en una categoría particular.

Incluso se puede abrir la "tarjeta Indice" para cada resultado (nivel 3). Para examinar los resultados con un grupo se siguen los pasos siguientes:

1. Usar la lista que muestra los métodos, como se muestra en la figura 3.53:

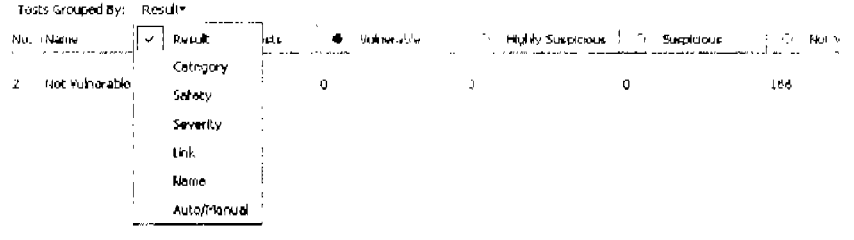


Figura 3.53. Examinando los resultados con un grupo seleccionado **(Sanctum, 2004)**.

Los resultados de la Prueba se listan de acuerdo al método escogido.

2. Mover el ratón hacia abajo en los nombres de la lista (el cursor se vuelve una mano y el nombre seleccionado se vuelve activo) y hacer clic en el nombre para ver los resultados individuales que pertenecen a ese grupo.

3. La lista completa de los resultados para el grupo seleccionado aparece en la pantalla, así se muestra en la figura 3.54:

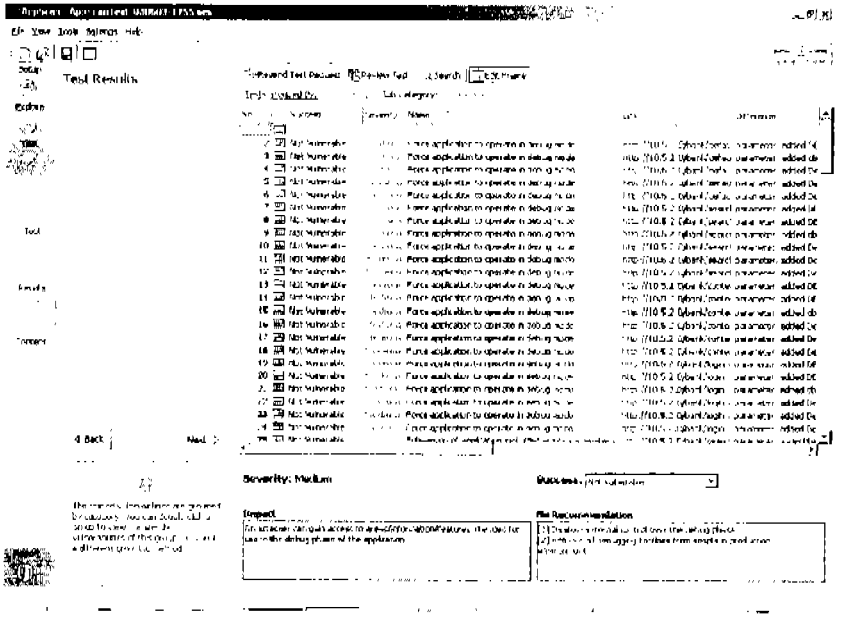


Figura 3.54. Lista completa de los resultados del grupo seleccionado **(Sanctum, 2004)**.

4. Hacer clic en un artículo para ver el consejo asociado. El renglón seleccionado aparece de un color distinto a los demás, y el consejo asociado aparece en la parte de abajo/medio en la pantalla.

5. Doble clic en un artículo para abrir la tarjeta índice de **AppScan DE** para ese artículo.

Tarjeta Índice de Resultados de AppScan DE. Hay una tarjeta índice para cada resultado de la Prueba. Incluso hay una tarjeta índice para cada comentario html del cual **AppScan DE** sospecha que contenga información sensible. Notar que en el caso de que haya comentarios sospechosos en el código html, ninguna petición fue mandada, y antes de eso las etiquetas de Respuestas están vacías.

Haciendo doble clic en un artículo individual (en la lista de resultados del nivel 2) abre la tarjeta índice para ese artículo, como se muestra en la figura 3.55:

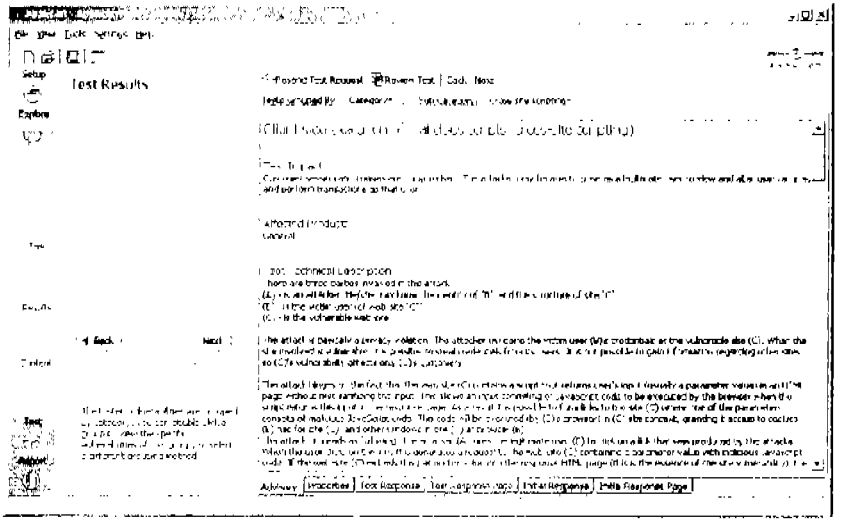


Figura 3.55. Tarjeta índice de resultados **(Sanctum, 2004).**

Notar que para que para cerrar la tarjeta índice y regresar a la lista de artículos, hacer clic en el botón de ***Back*** en el panel de instrucciones.

Las etiquetas de la tarjeta índice son explicadas en la Tabla que se muestra a continuación:

Tabla 10. Index card tabs summary

Nombre de la Etiqueta	Contenido
Advisory	El texto completo del consejo asociado con esta vulnerabilidad
Properties	Presenta la severidad y el resultado de la prueba también contiene un campo de comentarios donde se puede escribir comentarios que aparecerán en el reporte final
Test Response Source	Contiene el código fuente del http de la página de respuesta del sitio a la Prueba
Test Response Page	Contiene el sitio de respuesta a la Prueba que aparece como un Web Browser
Initial Response Source	Contiene el código del http de la respuesta inicial del sitio valida/espera la petición del http que se mando durante la etapa "Explore"
Initial Response Page	Contiene la respuesta inicial del sitio valida/espera la petición del http que aparece como un Web Browser

Etiqueta de Consejos. Cuando se hace doble clic en una entrada en la lista de resultados en el nivel 2, se abre la tarjeta índice para ese resultado.

La etiqueta más relevante en esta tarjeta es la etiqueta de consejos, como se muestra en la figura 3.56:

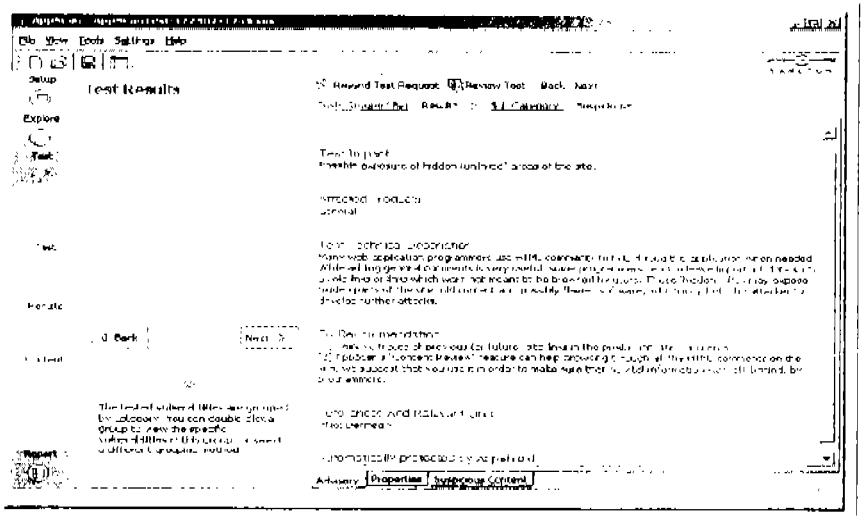


Figura 3.56. Etiqueta de consejos **(Sanctum, 2004).**

En la Tabla siguiente se explica cada punto que aparece en la etiqueta de Consejos:

Tabla 11 Advisory tab field summary

Nombre	Explicación
Header	Nombre del consejo
Category.	La categoría de los resultados de la Prueba que ese consejo aplica
Test Impact	El posible impacto en el sistema de un hacker utilizando esta vulnerabilidad
Affected Products	Productos afectados por esta vulnerabilidad
Technical Description	Descripcion detallada de un pasible ataque utilizando esta vulnerabilidad.
Recommended Fix	Una acción recomendada que puede resolver esta vulnerabilidad
References & links	Referencia adicional con respecto a esta vulnerabilidad
AppShield	¿Protege AppShield contra esto? Sí/no

Cambiar el *Score* de la Prueba. Se puede cambiar manualmente el *score* o resultado, esto se puede hacer de dos formas:

Método 1:

1. Seleccionar el resultado de la prueba que se quiere editar.

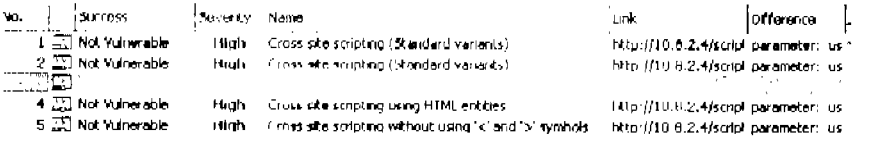


Figura 3.67. Seleccionar el resultado **[Sanctum, 2004].**

2. Hacer clic con el botón derecho en el record seleccionado.
3. En el menú que aparece hacer clic en ***Success.***

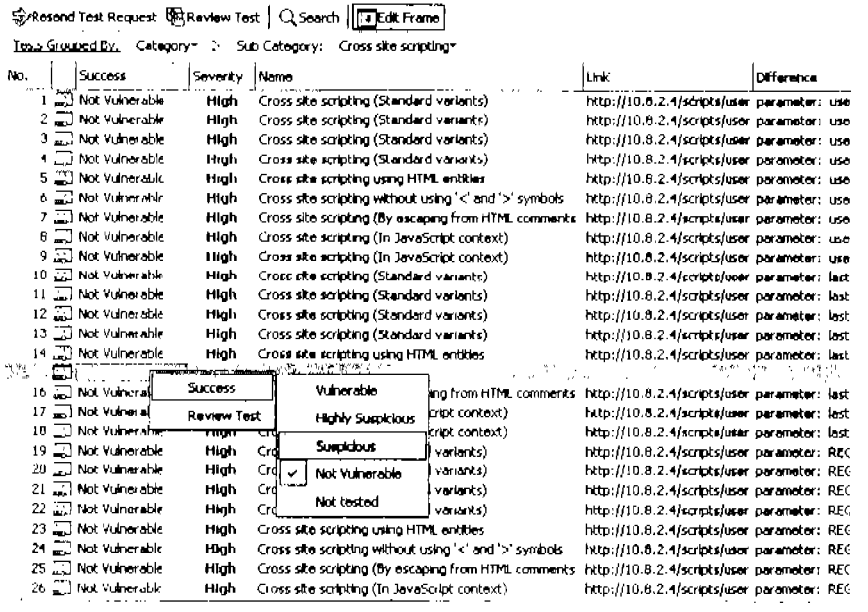


Figura 3.58. Menú **(Sanctum, 2004).**

4. En el menú que se abre hacer clic en el *score* que se quiere dar.

Método 2:

1. Seleccionar el resultado de la prueba que se quiere editar.
2. En el ***Edit Frame***, se puede editar el valor dado por la prueba, el cual aparece en la parte de abajo/medio en el área de administración donde dice ***Success,*** como se muestra en la figura a continuación:

Notar que el Edit Frame puede abrirse y cerrarse haciendo clic en

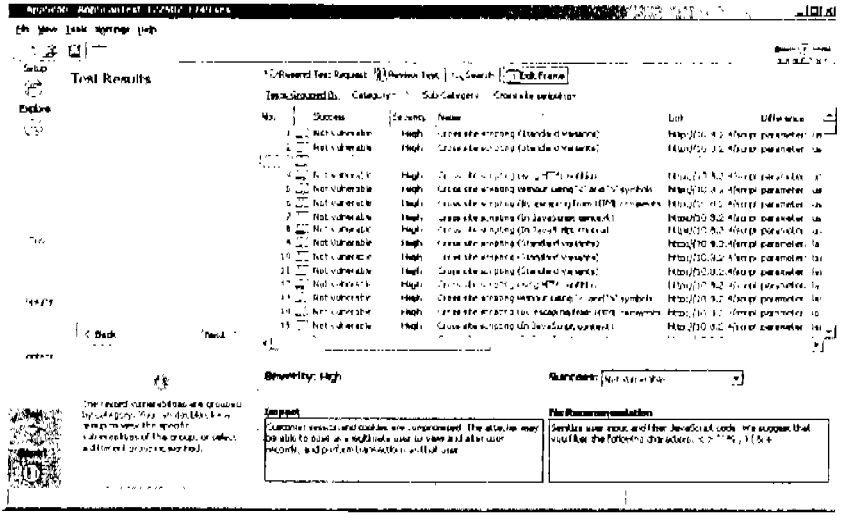


Figura 3.59. Editar frame **(Sanctum, 2004).**

Volver a mandar una Petición de la Prueba. En ocasiones se necesita volver a mandar una petición de la prueba sin que se edite, por ejemplo, si el sitio no estaba en la red cuando la petición se mando originalmente. Se puede volver a mandar una o más peticiones de la prueba de la ventana de resultados de la Prueba a la vista de nivel 2 o nivel 3. Esto se hace de la manera siguiente:

1. En la lista de Resultados en el nivel 2, seleccionar la Prueba (s) que se quiere volver a mandar, (usar ***Shift + Control*** para seleccionar más de un artículo). Los artículos seleccionados se ponen de otro color.

2. Clic en 

. La petición o peticiones se mandan otra vez, y los nuevos resultados son incluidos en la página actual de Resultados.

Revisar una Prueba. La caja de diálogo de revisar una prueba permite volver a revisar la petición de una prueba seleccionada. Para hacer estos se siguen estos pasos:

1. De la lista de peticiones de la Prueba, seleccionar el texto que se desea revisar.
2. Clic en ***Review Test.***
3. Para cerrar el panel de revisión hacer el clic en ***Close.***

Facilidades de Búsqueda. El área de administración de resultados de la prueba tiene un panel de búsqueda, el cual puede abrirse o cerrarse haciendo clic en él. El panel de búsqueda permite hacer una búsqueda por el *Path, Success, Request, Response and Keywords.* Como se muestra en la figura 3.60:

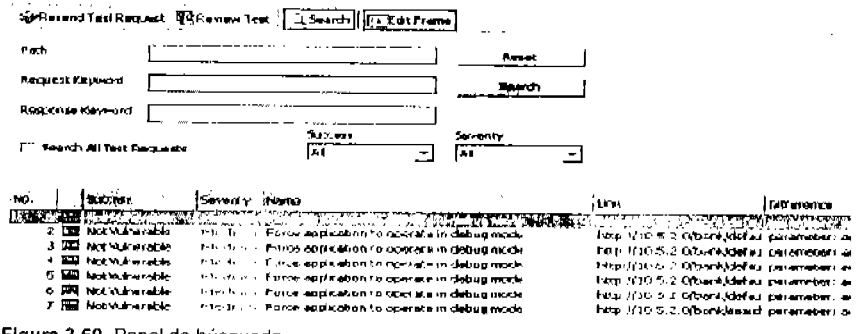


Figura 3.60. Panel de búsqueda.

c) **Contenido.**

Content **(Revisar el Contenido)**: el paso final en la etapa de Prueba es revisar el contenido. Esto ayuda a crear futuras pruebas para el sitio, basado en el contenido y la estructura del sitio. Se puede revisar *Scripts* para comentarios vulnerables escodidos, los cuales **AppScan DE** no descubrió automáticamente; también se puede ver *cookies* y CGI's.

La página que Revisa el Contenido aparece como se muestra en la figura 3.61:

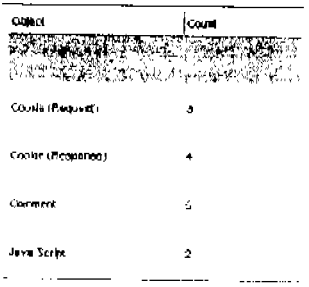


Figura 3.61. Página que revisa el contenido **(Sanctum, 2004).**

Y la Tabla siguiente explica los objetos que aparecen en la página que revisa el contenido:

Tabla 12. Content Review page objects

Objeto	Descripción
Scripts	Lista de todas las peticiones que se mandaron durante la etapa "Exploración" que contiene parametros (estos pueden indicar vulnerabilidades potenciales)
Cookie (Response)	Lista de todas las "cookies" recibidas del sitio en respuesta a la petición inicial
Cookie (Request).	Lista de todas las "cookies" mandadas al sitio (basado en la respuesta de las "cookies" recibidas), durante la etapa de "Exploración"
Comment.	Lista de todos los comentario encontrados en el texto html
Java Script	Lista de todos los Java Script encontrados en el texto html

Scripts. En esta sección se lista todas las ligas encontradas en el sitio que incluyen parámetros. Sobre cada *Script* hay una lista de parámetros asociados con la petición. Para cada parámetro se muestra el nombre, valor (es) y el tipo. Por ejemplo, sobre el *Script* seleccionado *"radio_post"* en la figura 3.62 el parámetro es: *"on"*.

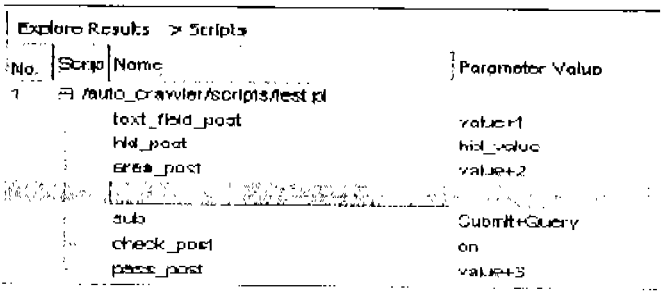


Figura 3.62. Scripts **(Sanctum, 2004).**

***Cookies* (Respuesta/Petición).** Las dos lista de *Cookies*, Respuesta y petición, listan todas las *cookies* mandadas por o para el sitio, respectivamente. Para cada *cookie*, el nombre, valor, y URL (de la primera petición/respuesta en la cual fue encontrada) es mostrada. En el caso de respuesta de *cookies*, la parte de abajo/medio en la pantalla muestra el Dominio, Expiración Fecha/Tiempo, y valor de la Seguridad (*Yes=Secure, No=Not Secure)* de la *cookie* seleccionada. (Esto es relevante para la petición de la *cookie*). Esto se muestra en la figura 3.63:

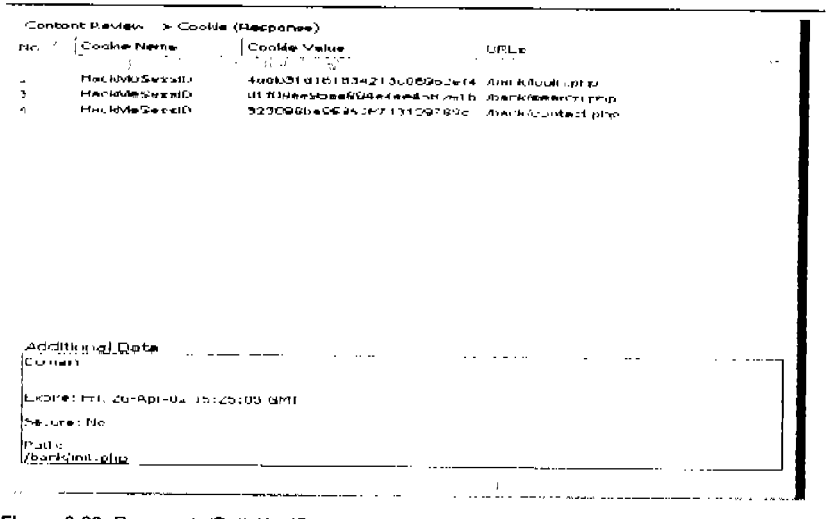


Figura 3.63. Respuesta/Petición **(Sanctum, 2004).**

Comentarios. Comentarios escondidos en una página html pueden contener información útil para los hackers. En ocasiones el autor del html, intencionalmente o por accidente, deja comentarios para él mismo o para otros desarrolladores en la página final, y esta queda en línea, asumiendo que nadie más lo verá. Un hacker puede cosechar información interna útil de esos comentarios, tal como un *"debug password"*. La ventana de Comentarios lista el inicio de cada comentario y su *path*. La parte de abajo/medio de la pantalla muestra el texto completo del comentario seleccionado, como se muestra en la figura 3.64:

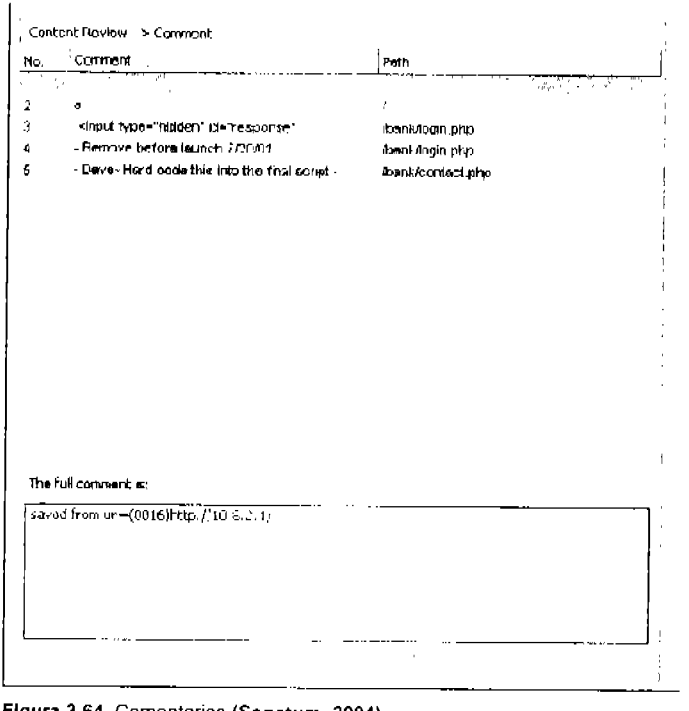


Figura 3.64. Comentarios (**Sanctum, 2004**).

JavaScript. Comentarios con un *JavaScript* puede revelar información sensible para un hacker, como un resultado analizando *JavaScript* en un HTML, el archivo puede revelar vulnerabilidades potenciales.

La ventana de *JavaScript* lista el inicio de cada comentario y su relevante URL. En la parte de abajo/medio de la pantalla muestra el texto completo del *strip* seleccionado, como se muestra en la figura 3.65:

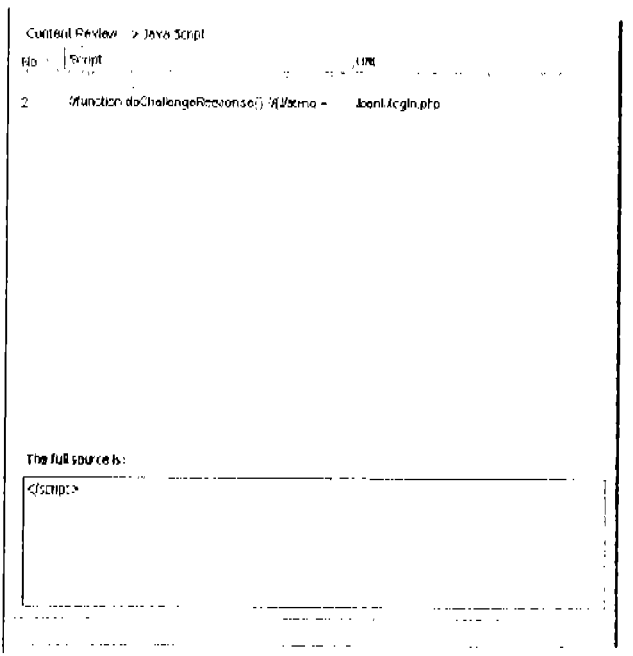


Figura 3.65. JavaScript **(Sanctum, 2004).**

3.6.4. Etapa del Reporte.

La tercera y final parte de **AppScan DE** es la etapa del Reporte, en la cual se recibe un reporte detallada del resultado del Scan. La etapa del Reporte esta subdivido en tres partes:

a) Configuración.

Configure: conjunto de información para ser desplegado y despliega un formato.

Reporte de la Prueba. El reporte de la prueba contiene una lista completa de todas las vulnerabilidades (listadas por título de la Prueba). Los títulos de la Prueba están listados por Consejos. Cada relevante Consejo es detallado completamente, y seguido por una lista de la Prueba realizada, y de las ligas vulnerables por cada prueba. La figura 3.66 muestra la pantalla de la etapa del Reporte:

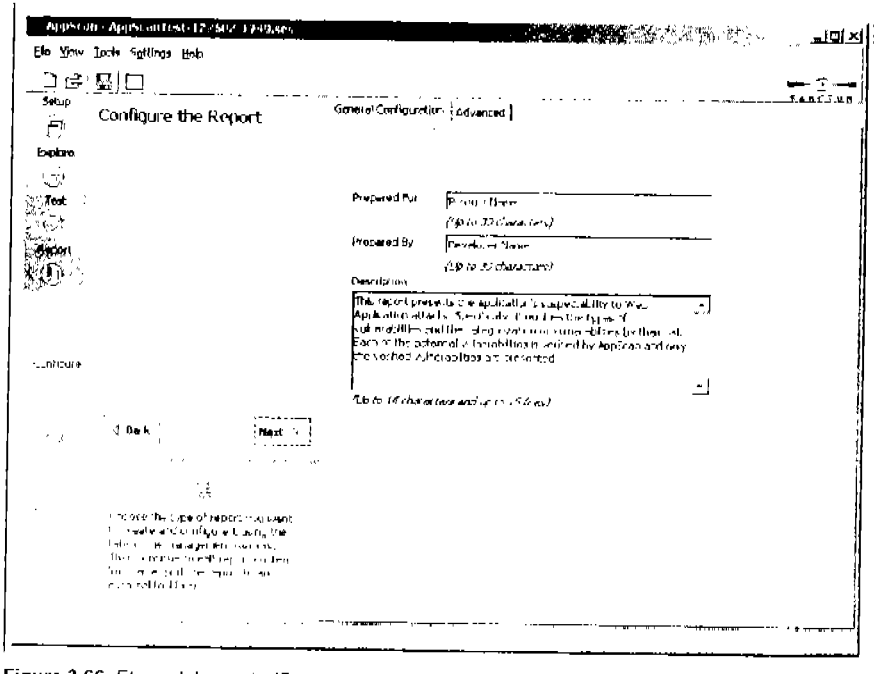


Figura 3.66. Etapa del reporte **(Sanctum, 2004).**

Y la Tabla siguiente explica las secciones del Reporte de **AppScan DE:**

Tabla 13 Report fields

Artículo	Explicación
General	Título, compañía Inicio/Final de Datos y Tiempos y una breve descripcion del contenido del reporte
Vulnerabilities per Host	Tabla de todos los hosts encontrados vulnerables y el número de vulnerabilidades por hosts
Vulnerability Highlights	Tabla de todas la vulnerabilidades encontradas donde están agrupadas de acuerdo al riesgo éxito título y categoria
Scan Statistics	Los URLs en la aplicacion todos los links descubiertos/escaneados presentados como tabla y gráfica
	Vulnerabilidades de la aplicación (sitio) las potenciales y verificadas son presentadas como tabla y grafica
	Contenido de la aplicacion averia estadistica del número total de contenido de Script Comentarios Java Script peticiones de "cookies" y respuesta de "cookies"
Vulnerabilities	Lista completa de todos los links vulnerables, clasificados por el consultor y el título de la vulnerabilidad
Cookies	Lista completa de las "cookies" de petición y respuesta, clasificadas por nombre del "cookie"
Scripts	Lista completa de los Script encontrados en en sitio, clasificados por título del Script.
Comments	Lista completa de los comentarios encontrados en el sitio, clasificados por título del comentario
JavaScript	Lista completa de los Java Scripts encontrados en el sitio, clasificados por título del Script
	(Solo para información detallada del Reporte del Scan)
Terms and Definitions	Apéndice uniforme que contiene los términos y las definiciones añadido a todo reporte

Configuración del Reporte. El primer paso de la etapa del Reporte es hacer la apariencia del reporte y proveer a **AppScan DE** con la información necesaria para ser llenada en la plantilla del reporte. (La plantilla del reporte se define durante la etapa de *Setup*).

Configuración General del Reporte. En la etiqueta de la configuración general del reporte se puede dar a **AppScan DE** información general para ser añadida al reporte del Scan, como se aprecia en la figura anterior:

- Nombre del Producto.
- Nombre del Desarrollador.
- Editar una corta descripción del reporte. Se puede cambiar opcionalmente el comentario original que aparece en el campo de comentarios.

Toda esta información será añadida a la página principal del reporte una vez seleccionada.

Configuración Avanzada del Reporte. En la etiqueta de configuración avanzada del reporte, se puede decidir entre incluir o no, información adicional en el reporte del Scan:

Información General:

- Reporte Ejecutivo. Esta versión explicada del reporte del Scan es detallada automáticamente al inicio del Reporte de la Prueba.
- Estadísticas del Scan. La información de las Estadísticas del Scan, se presenta en varios modos gráficos, como se muestra en la figura 3.67:

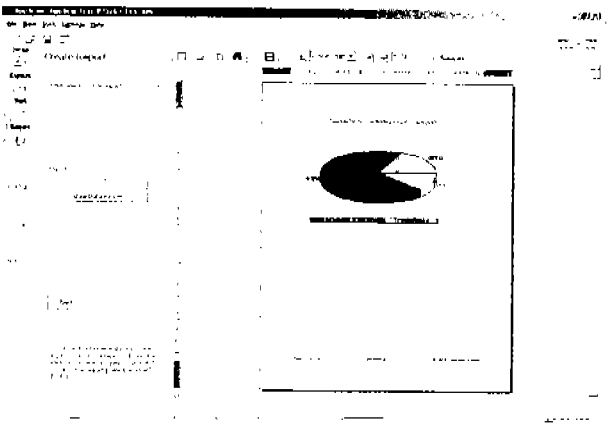


Figura 3.67. Estadísticas del Scan **[Sanctum, 2004]**.

• Términos y Definiciones. Una sección de términos y definiciones, explica varios términos y definiciones en el reporte.

Contenido Técnico:

• *Cookies* mandadas por Peticiones.
• *Cookies* recibidas por peticiones.
• Lista de *Scripts*.
• Lista de Comentarios.
• Lista de *JavaScripts*.

Todo lo anterior se puede apreciar en la figura 3.68:

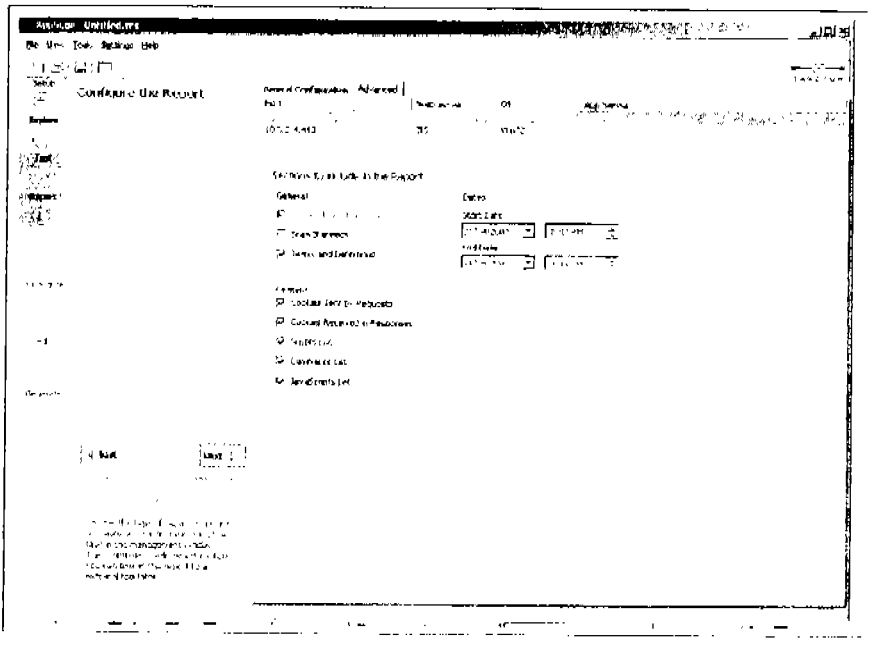


Figura 3.68. Contenido técnico **(Sanctum, 2004)**.

Signatura de la fecha y el tiempo:

Tal vez se tenga que cambiar el inicio y el final de la signatura de la fecha y el tiempo en el Reporte.

La fecha puede ser cambiada tecleando en su respectiva caja de selección, o seleccionado en el menú la fecha del calendario.
El tiempo puede ser cambiado usando las flechas de arriba y abajo para los campos de Inicio/Final del tiempo. A continuación se muestra en la figura 3.69:

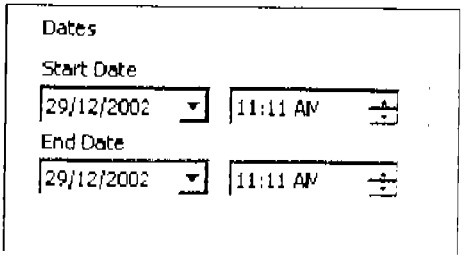


Figura 3.69. Fecha y tiempo **[Sanctum, 2004].**

b) Editar.

Edit: se editan los resultados antes de generar el Reporte. (Solo si el Reporte Detallado es seleccionado). Un reporte interno de **AppScan DE** aparece en la pantalla.

Las categorías pueden ser expandidas y colapsadas. Esta pantalla se muestra en la figura 3.70:

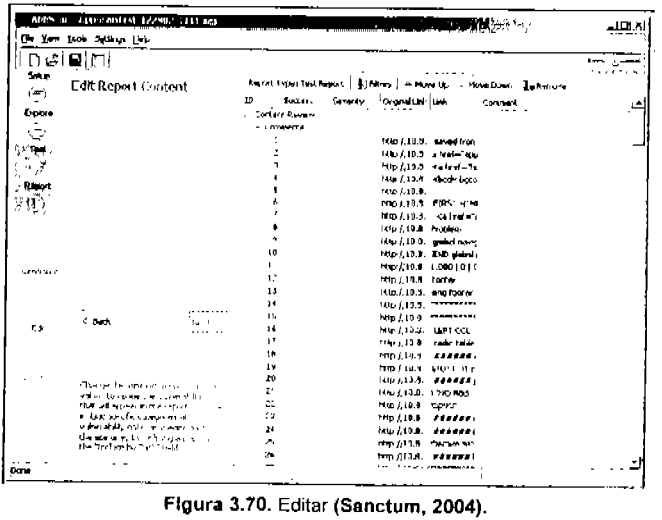


Figura 3.70. Editar **(Sanctum, 2004).**

Estructura Interna del Reporte. Las entradas en el Reporte Interno tiene una jerarquía de tres niveles. Los niveles pueden ser expandidos y colapsados en la pantalla, haciendo clic en "+/-", puesto al inicio de una entrada.

- Nivel 1: entradas de los nombres del consejo.
- Nivel 2: entradas de los titulos de Verificación/Prueba.
- Nivel 3: entradas de las peticiones individuales de la prueba.

La figura 3.71 muestra la estructura de los niveles:

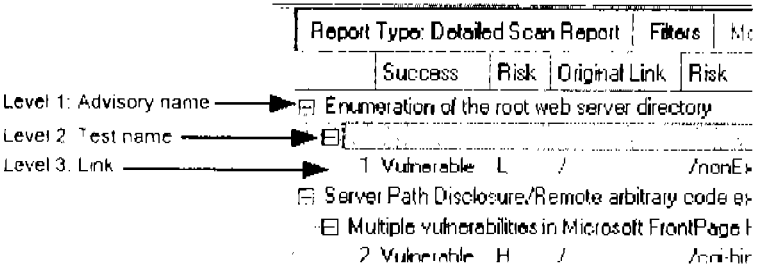


Figura 3.71. Estructura de los niveles **(Sanctum, 2004).**

La última entrada de Nivel 1 es única, es llamada *"Content Review"* y contiene cinco entradas de Nivel 2: CGI's, Comentarios, *JavaScripts*, Peticiones de *cookies* y Respuestas de *cookies*. Sobre cada Nivel 2 todos los artículos que pertenecen a esa categoría son listados.

Barra del Menú del Reporte Interno. El reporte interno tiene su propia barra de menú, la cual se puede usar para preparar el reporte final, como se muestra en la figura 3.72:

Figura 3.72. Barra del menú del reporte interno **(Sanctum, 2004).**

Y la Tabla a continuación explica los artículos de la barra de menú:

Tabla 14 Internal Report menubar items

Artículo	Función
Filters	Abre un diálogo del Filtro para filtrar artículos del reporte
Move Up / Move Down	Mueve el consultor seleccionado un lugar arriba o abajo en el reporte
Remove	Borra el link o artículo seleccionado en el reporte

Editando el Reporte Interno:

Se puede Cambiar el orden de los Consejos en el Reporte. Se puede cambiar el orden en el cual los consejos aparecen en el Reporte Detallado del Scan. Para promover/degradar un consejo, se siguen los pasos siguientes:

1. Seleccionar el consejo que se quiere mover, seleccionándolo. El consejo seleccionado se pone de otro color.
2. Clic en ***Move up/Move down*** para promover/degradar el consejo. El consejo es reasignado en la pantalla y aparece en el nuevo orden en el Reporte. notar que solo los artículos (consejos) del Nivel 1 pueden ser movidos.

Borrar artículos del Reporte. Se puede borrar ligas individuales del Reporte, para hacer esto se siguen los pasos a continuación:

1. Seleccionar el artículo que se quiere borrar, haciendo clic en él. El artículo seleccionado se pone de otro color.
2. Hacer clic en ***Remove.*** El artículo seleccionado es borrado y no será incluido en el Reporte. Notar que solo los artículos del Nivel 3 pueden ser borrados.
3. Para generar el reporte, clic en .

Contenido Filtrado en el Reporte. En adición, para escoger cuales secciones aparecen en el reporte, se puede incluso seleccionar el *nivel* de contenido en el reporte.

Usando la caja de diálogo de Filtro en la barra de menú en la Edición Interna del Reporte, en la figura 3.73 se muestra esta caja de diálogo:

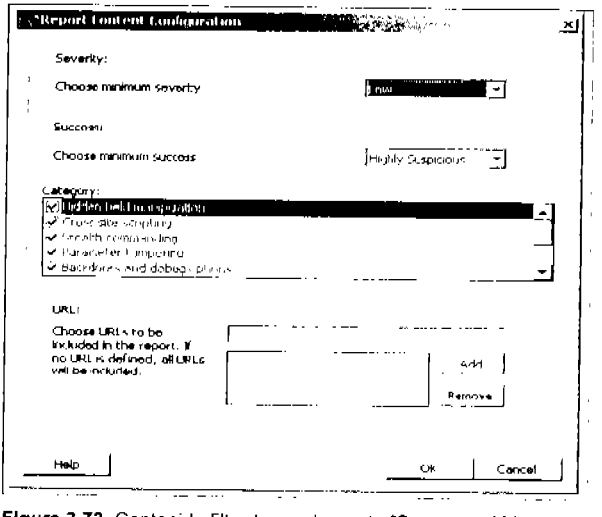


Figura 3.73. Contenido filtrado en el reporte **[Sanctum, 2004]**.

La Tabla siguiente explica los objetos de esta caja de diálogo:

Tabla 15 Report Content Configuration

Campo	Función
Severity	Selecciona el nivel mínimo que será incluido en el reporte High/Medium/Low
Result	Selecciona la cuenta mínima del exito que se incluirá en el reporte Vulnerable / Highly Suspicious / Suspicious / Not Vulnerable
Category	Se utiliza para incluir/excluir las categorías de la Prueba en el reporte
Path	Se puede incluir un path en el reporte poniendo uno en el campo de texto que aparece y haciendo click en Add. Si el campo esta vacío todos los path serán incluidos por default

c) Generar el Reporte.

Generate: se genera el reporte de **AppScan DE** para verlo en la pantalla, imprimirlo y exportarlo a otros formatos, esto se muestra en la figura 3.74:

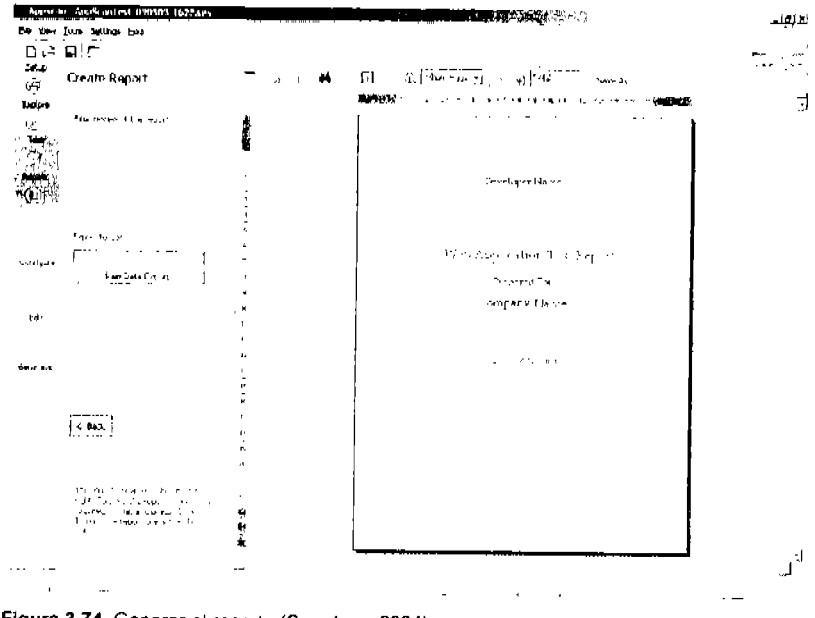


Figura 3.74. Generar el reporte **(Sanctum, 2004).**

Barra de Herramientas del Reporte. El Reporte tiene una barra de herramientas intuitiva para ayudar a navegar el documento, como se muestra en la figura 3.75:

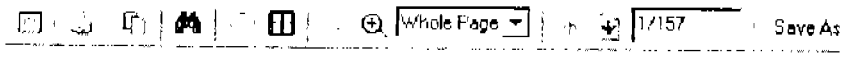


Figura 3.75. Barra de herramientas del reporte **(Sanctum, 2004).**

Exportar el Reporte para Leer/Editar en otro Programa. El botón de ***Save As*** permite salvar el Reporte en PDF/Excel/HTNL/RTF/Text/Tiff formats, **Exportar el Reporte a un Formato CSV.** Para hacer esto se siguen los pasos a continuación:

- En el panel de instrucciones, seleccionar un directorio para salvar el reporte hecho.
- Presionar ***Raw Data Export***. Los datos del reporte son salvados en el disco en formato CSV con cada sección del reporte exportada en un archivo separado.

Ejemplos de Reportes.

Vulnerabilidades por *Host*. Una tabla muestra el número de ligas por cada nivel de riesgos por *Host*. La tabla es seguida por un gráfico.

Vulnerabilities Per Host

Host	Low Risk	Medium Risk	High Risk	Total
0.8.3.1	25	129	143	298
0.8.4.443	0	10	2	12
0.8.2.19030	0	29	1	30

Figura 3.76. Vulnerabilidades por host **[Sanctum, 2004].**

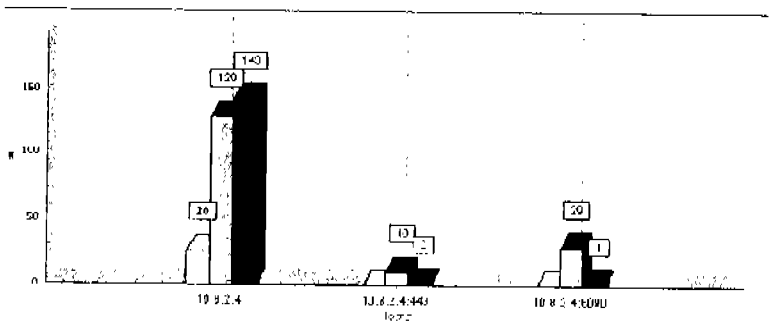


Figura 3.77. Vulnerabilidades por host **(Sanctum, 2004).**

Medidas de las Vulnerabilidades. La tabla de medidas de las vulnerabilidades lista las ligas en orden de acuerdo al nivel de la vulnerabilidad y el nivel de riesgo, como se muestra en la figura 3.78, seguido por detalles aconsejables y el número de ligas en esa categoría.

Vulnerability Highlights

Success	Severity	Name	Category	Impact	#
Highly Suspicious	Medium	Filename references found in HTML comments	Suspicious contents	[1] The computer's file system structure can be deduced from the filenames mentioned in the hidden content [2] Exposure of filenames related to the web application.	33
Highly Suspicious	Medium	References to sensitive information found in HTML comments	Suspicious contents	Exposure of vital information such as debugging information, usernames or passwords in HTML comments.	7
Highly Suspicious	Medium	Debugging remnants found in HTML comments	Suspicious contents	Exposure of vital information such as debugging information, usernames or passwords in HTML comments.	2
Highly Suspicious	Medium	Calls to 'document.write()' found in JavaScript	Suspicious contents	Exposure of application logic to the attacker may enable him/her to bypass security mechanisms	7
Highly Suspicious	Medium	References to cookies found in JavaScript	Suspicious contents	Attackers may be able to manipulate authorization mechanism	3

Figura 3.78. Medidas de las vulnerabilidades **(Sanctum, 2004).**

URL´s en la Aplicación. Los URL´s en la aplicación (sitio) muestra el número total de URL´s encontrados, y el número total escáneado.

URLs in the Application	Total
URLs (links) Discovered by AppScan	2039
URLs scanned by AppScan (of the total URLs discovered)	361

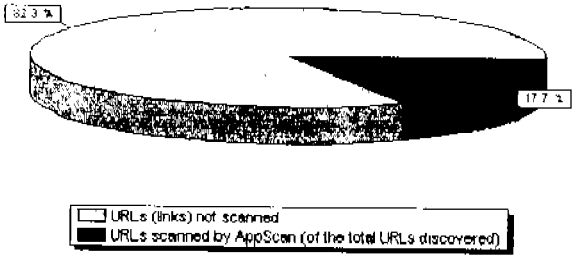


Figura 3.79. URL´s en la aplicación **(Sanctum, 2004)**.

Vulnerabilidades de la Aplicación.

Vulnerabilities of the Application	Total
Potential Vulnerabilities	6763
Verified Vulnerabilities (of the total Potential Vulnerabilities)	2826

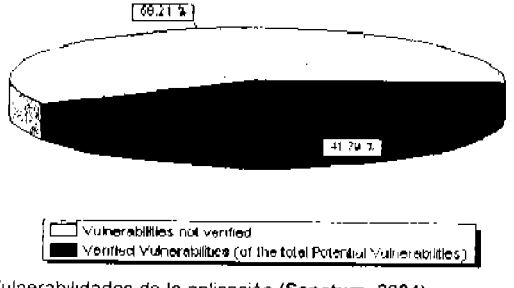


Figura 3.80. Vulnerabilidades de la aplicación **(Sanctum, 2004)**.

Contenido de la Aplicación.

Application Content	Total
Scripts Scanned in the Application	51
Comments in the Application detected by AppScan	102
JavaScript Scripts detected by AppScan	170

Figura 3.81. Contenido de la aplicación **(Sanctum, 2004).**

En este capitulo hemos visto porque razones es importante contar con una herramienta de apoyo, como nos puede ayudar una herramienta contra las perversiones del Web. También analizamos cuales son las necesidades del Centro de Computo y que beneficios traería esta herramienta al Centro de Computo. Posteriormente analizamos todo el funcionamiento de la herramienta de AppScan DE.

Capítulo 4.

Lineamientos de Seguridad para lograr la Confiabilidad y Calidad del Software basado en Internet.

En este capítulo plantearemos una serie de lineamientos, estos ayudarán a los investigadores del Centro de Computo para que sus Aplicaciones Web sean cada vez más seguras y confiables y que al mismo tiempo ofrezcan calidad al usuario. Estos lineamientos pueden ser planteados en la práctica, ya sea al inicio, en el desarrollo o al final de la construcción de una Aplicación Web.

4.1. Planificar la Seguridad.

La planificación de la seguridad es la etapa en donde se realiza la identificación actual de los recursos informáticos, los alcances de los servicios que estos brindan a los usuarios a nivel de aplicaciones, con lo cual se hace una proyección en base al crecimiento de los servicios que se ofrecen, identificando los requerimientos necesarios para implementar y controlar el funcionamiento del mismo. Para lo cual se deben tener en cuenta las siguientes consideraciones:

Acciones.

- Demostrar que los costos de los recursos de seguridad están entendidos e incorporados en la planeación del ciclo de vida del sistema.
- Incorporar un plan de seguridad que comprenda:

1. Las reglas de utilización del sistema y las consecuencias al violar dichas reglas.

2. Los métodos para identificar, limitar apropiadamente, y administrar los límites de las interconexiones con otros sistemas y los procedimientos específicos para vigilarlos.
3. Procedimiento para el monitoreo de la eficacia de los controles de la seguridad.
4. Plan de contingencia en la presencia de un ataque o accidente.

4.2. Alinearse a la Arquitectura de información de la Institución.

Define la necesidad de considerar dentro de la infraestructura física y funcional de los sistemas de información, el desarrollo de un plan de seguridad en función al crecimiento de esta arquitectura.

Acciones.

- Los controles de la seguridad para los componentes, las aplicaciones, y los sistemas deben ser parte integral de la arquitectura tecnológica de los sistemas de información de la institución.

4.3. Tomar control de los riesgos.

Los riesgos son constantes y se necesita tener los mecanismos necesarios para analizarlos y posteriormente aplicar las estrategias precisas frente a estos riesgos, y no esperar a que ocurran para construir una solución

Acciones.

- Establecer un método específico y entendible para evaluar continuamente los riesgos potenciales, con la finalidad de mantener la seguridad a un nivel aceptable, así como los procedimientos para asegurar un control eficaz con los tiempos de respuesta necesarios.
- Identificar de ser necesario controles adicionales de seguridad para reducir al mínimo el riesgo potencial de pérdida de los sistemas a promover o permitir acceso público.

4.4. Proteger la privacidad.

Es necesario el establecimiento de mecanismo que ayuden a proteger la privacidad, que realizan comunicación para acceder al uso de recursos de información de las entidades públicas.

Acciones.

- Mantener herramientas eficaces para el control de la seguridad.
- Asegurar que la información personal este respaldada por políticas del gobierno.
- Controlar las aplicaciones y la información a que tienen acceso los usuarios y cómo pueden llegar hasta ellas. Controlar quien puede abrir cuentas o crear identificaciones de usuarios en un sistema, auditar las cuentas con frecuencia en busca de identificaciones o cuentas que no corresponden a la realidad de tener a manos personas capaces de llevar a cabo una auditoría.

4.5. Mantener estándares de seguridad.

Los estándares en informática son las recomendaciones técnicas que facilitan una mejor administración y crecimiento de los recursos informáticos de información a nivel nacional.

Acciones.

- Para la seguridad de las aplicaciones, asegurar el uso de estándares, al implementar productos y herramientas.
- Usar productos de seguridad disponibles y probados en el mercado. Los productos basados en estándares abiertos han sido probados y aprobados. Lo más importante de todo es que los productos estandarizados de la industria están de modo típico, bien documentados para que se empleen bien.

4.6. Mantener simple para los usuarios la implementación de los planes de seguridad.

Si el sistema es muy complicado, los usuarios lo evitarán o tratarán de darle un rodeo con lo que se anularán las medidas se seguridad y se reducirá su utilidad, las medidas de seguridad modernas pueden ser efectivas sin interferir.

4.7. Desarrollar y cumplir de forma proactiva políticas, procedimientos y sanciones.

Se deben establecer métodos para tener la certeza de que las políticas de seguridad establecidas, se están cumpliendo correctamente.

Acciones.

- Diseñar un sistema de seguridad que se base en las necesidades del usuario, la naturaleza de las aplicaciones y la información que se asegura.
- Aplicar medidas de seguridad constantemente, pues tener políticas de seguridad que no se aplican es peor que no tener ninguna.

4.8. Probar, auditar, inspeccionar sitios e investigar continuamente.

La seguridad de la información es una actividad constante y necesita del establecimiento de acciones continuas, por ellos es necesario tomar en cuenta consideración las siguientes acciones:

Acciones.

- Realizar de forma periódica análisis de vulnerabilidades de los sistemas de información para protegerse de las amenazas internas y externas.
- Usar una metodología para examinar y probar códigos para bloquear las puertas traseras de los sistemas informáticos.
- Usar un sistema de auditoría automática y programas de vigilancia.
- Dar a conocer las amenazas y las respuestas que se les da.

En este capítulo se plateó una serie de lineamientos para apoyar a los investigadores del Centro de Computo a tener un desarrollo de Aplicaciones Web más seguras.

Poniendo en práctica estos lineamientos se puede llegar a tener una seguridad de las Aplicaciones Web aceptable.

Capítulo 5.

Conclusiones.

Durante este trabajo de tesis hemos tenido la posibilidad de valorar los diferentes estudios y criterios sobre la seguridad informática para aplicaciones basadas en Internet, para asegurar su confiabilidad y calidad. Desgraciadamente, los desperfectos de la seguridad se dan con mucha frecuencia, y los cortafuegos corporativos a menudo son inútiles cuando un hoyo de la seguridad existe en una aplicación. Sin embargo, existe la esperanza en la forma que **AppScan DE** ayuda a que los desarrolladores prueben sus aplicaciones en tiempo de diseño. **AppScan DE** es el primer instrumento de su clase que ayuda con esto. Permite a los desarrolladores a probar la seguridad cuando ellos están en el desarrollo de la aplicación, así como con en el producto terminado. Enseña e impone mejores prácticas que las organizaciones pueden construir en sus pautas del desarrollo y diseño.

Se llevó a cabo un análisis detallado de herramientas de apoyo para asegurar las Aplicaciones Web del Centro de Cómputo, particularmente **Manager PKI** y **AppScan DE**. La comparación de ambas se encuentra en el apéndice. Concluimos que **AppScan DE** podrá servir como base a futuras Aplicaciones Web. Por esto realizamos las siguientes recomendaciones:

1. El primer paso que se debe de seguir para asegurar una aplicación Web, es saber cuán vulnerable es. Cuando las aplicaciones llegan a ser más sofisticadas, las presiones del tiempo en el mercado son una parte cada vez más significativa de los problemas actuales de la seguridad. Además, cuanto más las compañías exponen sus sistemas a clientes, se hacen a sí mismas más responsables de la seguridad.
2. **AppScan DE** explora un sitio Web, busca potenciales defectos de seguridad, "ataca" el sitio basado en el conocimiento que reunió durante la

búsqueda, e informa sus éxitos y los fracasos. De acuerdo con la investigación hecha en este trabajo de tesis, encontrar un defecto durante el despliegue de la aplicación es 100 veces más costoso que la modificación en el código en tiempo de diseño. Si no se resuelven estos problemas permitirán a piratas informáticos tomar fácilmente un sistema, robar identidades e incluso cambiar los datos críticos.

3. Con **AppScan DE**, los desarrolladores tienen acceso a un instrumento integrado que hace un código seguro que es una meta accesible, los clientes pueden recibir aplicaciones de calidad. Arreglar hoyos de seguridad en el despliegue de la aplicación es muy costoso, equiparse con una herramienta como esta será una fuerte inversión que ayudara a crear mejores y más seguras aplicaciones Web. **AppScan DE** puede hacer pruebas de defectos de seguridad mientras los desarrolladores trabajan en la aplicación. Esto se hace en tres pasos muy fáciles, como se muestra en la figura 5.1:

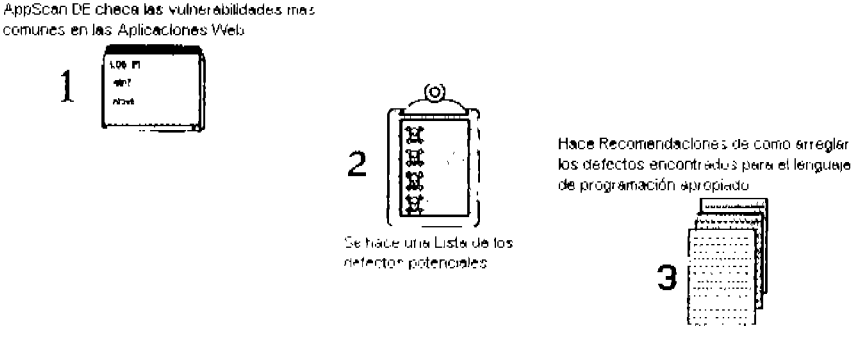


Figura 5.1. Pasos para detectar defectos de seguridad **(Sanctum, 2004)**

4. **AppScan DE** es un instrumento para probar la seguridad de las aplicaciones Web. Al saber más del producto de **AppScan DE**, será difícil de evitar la impresión de que **AppScan DE** es como un tipo de "pirata informático". Es verdad en por lo menos un sentido: **AppScan DE** aplica el mismo tipo de técnicas como un pirata informático suele infiltrarse a un sitio

Web. Afortunadamente, **AppScan DE** es un mecanismo benigno, no una persona malévola, porque hace su trabajo muy bien.

5. Otro resultado valioso en este trabajo ha sido el establecimiento de un conjunto de de lineamientos de seguridad para lograr la confiabilidad y calidad del software basado en Internet. No hay menor duda que si se siguen los lineamientos planteados en el capítulo 4, será cada vez menos la preocupación constante que tienen los investigadores del Centro de Cómputo de acuerdo a la seguridad de sus Aplicaciones Web, ya que estos lineamientos son un buen consejo para tener una seguridad aceptable en un sistema.

De esta manera, se concluye que **AppScan DE** es la mejor opción para asegurar las Aplicaciones Web del Centro de Computo descubriendo sus vulnerabilidades y generando recomendaciones para arreglarlas en la etapa de desarrollo y de algunas aplicaciones que ya han sido desplegadas y al mismo tiempo siguiendo los lineamientos que se plantearon en el capítulo 4, esto ahorrará tiempo y costos significativos para el Centro de Computo.

Apéndice.

Comparación entre MPKI y AppScan DE.

Manager PKI (Public Key Infrastructure).

La seguridad y la confianza son partes integrales de los negocios hoy en día.

Hoy más que nunca, los negocios deben proteger sus recursos de ataque así como la implementación en medidas de seguridad en procesos administrativos, de personal, clientes y socios de negocios.

El managed PKI engloba los valores de:

- Autenticación.
- Autorización.
- Confidencialidad.
- Integridad.
- Irrefutabilidad.

El managed PKI es un servicio administrado, por lo tanto se puede implementar una solución PKI de una forma rápida, para englobar los componentes importantes de confianza. Actualmente el managed PKI da servicio a millones de negocios y usuarios a nivel internacional **[VeriSign, 2004].**

El managed PKI asegura sus:

- Aplicaciones.
- Comunicaciones.
- Transacciones.

De esta manera, asegurando estos tres conceptos, logra una infraestructura confiable entre uno y el mundo.

¿En dónde empieza la confianza?

La confianza empieza con la Autoridad Certificadora (CA) durante una implementación Managed PKI. La CA establece políticas de confianza y controla los certificados digitales, los cuales únicamente se entregarán autenticados y verificados.

Debido a que la CA, cumple con los parámetros de verificación y autenticación, todos los certificados emitidos por ésta son confiables, el Managed PKI permite comunicaciones y transacciones seguras entre uno y personas confiables, usando métodos de criptografía probados **[VeriSign, 2004]**.

La empresa que hospeda y administra los CA's es VeriSign, empresa reconocida mundialmente; esta se basa en su experiencia e infraestructura de seguridad, bajo los estándares más estrictos:

- Centro de datos de alta disponibilidad
- Escalables a millones de usuarios.
- Servicio al Cliente las 24 horas del día.
- Infraestructura de alta seguridad.
- Controles internos auditables.
- Cuenta con los mejores expertos de seguridad en la industria.

Todos estos respaldados por los mejores Acuerdos de Servicio en la Industria, para que uno desarrollé sus requerimientos legales y de seguridad.

Dependiendo de las políticas de confianza de la empresa, VeriSign hospedará CA's privadas o públicas. Las privadas permiten que el usuario personalice sus propias políticas confianza, mientras que las públicas se convierten, de forma inmediata, en parte de la Red de Confianza de Verisign (VTN).

Cuando un usuario pertenece a la Red de Confianza de Verisign, inmediatamente puede acceder a millones de usuarios confiables, sin tener que establecer su propia comunidad de confianza.

El servicio de Managed PKI permite que el usuario obtenga certificados digitales firmados por una Autoridad Autenticadota hospedada por Verisign, aprovechando las capacidades de infraestructura, confianza y seguridad que brinda esta, para que el usuario pueda enfocarse solamente a su negocio.

A continuación veremos como funciona el Managed PKI:

Por ejemplo si un nuevo usuario intenta acceder a una aplicación segura en un servidor empresarial, su acceso será denegado si éste no cuenta con el certificado digital correspondiente. Para que pueda acceder al servidor seguro, el usuario será diseccionado a un servidor de inscripción para obtener su certificado digital, que será hospedado en Verisign o en la empresa **[VeriSign, 2004]**.

Cuando el usuario navega en la página de inscripción, llena un formato personalizado para solicitar su certificado digital, y lo manda al servidor de inscripción.

La petición de certificado será guardada para una aprobación manual, o se mandará a un servidor de autenticación, el cual la aprueba o la rechaza en base a la información personal proporcionada en la inscripción. Si se aprueba la petición Verisign genera un certificado firmado digitalmente por una Autoridad Certificadora, y es mandado directamente al usuario. El acceso a la aplicación segura es concedido en base al nuevo certificado digital del usuario y a la llave privada correspondiente, y el usuario podrá acceder a la aplicación segura por el tiempo en que el certificado digital sea válido.

El servicio administrado Managed PKI puede soportar las aplicaciones de seguridad más fuertes del mercado, tales como:

- Banca Electrónica.
- Comercio Electrónico.
- Administración de Proveedores.
- Servicios Médicos.
- Gobierno.

AppScan DE.

AppScan DE brinde una solución para automatizar los análisis de vulnerabilidades y pruebas de penetración de sus aplicaciones y plataformas Web.

Elimina los exámenes manuales que eran necesarios antes de implementar una aplicación, genera reportes que determinan la mejor manera de cumplir con estas auditorías para asegurar sus aplicaciones, antes de su implementación.

AppScan DE es una poderosa herramienta de pruebas que permite el rápido desarrollo de la seguridad. Esta herramienta ayuda a hacer que la lógica de la aplicación sea resistente a ataques sin tocar su presentación o eficacia. **AppScan DE** detecta los defectos de la seguridad automáticamente; como un componente integrado al desarrollo de la empresa, esta herramienta, automatiza las pruebas de creación de escritura, modificación y proceso de mantenimiento, asegurando confiabilidad y pruebas que son repetibles **[Sanctum, 2004]**.

AppScan DE es una herramienta que ayuda a las empresas a reducir costos y a crear aplicaciones confiables y resistentes contra hackers, en el ambiente de desarrollo.

AppScan DE crea "Vulnerabilidades Potenciales" que son los defectos potenciales de la seguridad en el código y entonces los prueba para verificar que ellos existen.

AppScan DE reporta los errores o defectos de la aplicacion, luego se los proporciona al usuario para empezar a arreglar estos errores.

Después de estas breves descripciones de estos dos sistemas, podemos decir que las dos son muy buenas opciones para asegurar las Aplicaciones Web del Centro de Cómputo. Sin embargo, considero que **AppScan DE** es una opción mejor que el Managed PKI porque se enfoca más a descubrir cuáles son las vulnerabilidades potenciales que tienen las Aplicaciones Web, y es ahí, donde los hackers pueden atacar el sistema. Como observamos, Manager PKI asegura las Aplicaciones creando certificados digitales, pero no sabe cuales son las vulnerabilidades del sistema; si un hacker encuentra estas vulnerabilidades puede ser un gran problema para la empresa, o bien, si emplea cualquier técnica de hackers avanzadas para penetrar al sitio, rompiendo el esquema del certificado digital, sería un caos total **[Sanctum, 2004]**.

Bibliografía.

[Aslam, 1996]

Taimur Aslam, Ivan Krsul, and Eugene H. Spafford. Use of a taxonomy of security faults. Technical Report TR-96-051, Purdue University Department of Computer Science, 1996.

[Bellovin, 1993]

Steven M. Bellovin. Packets found on an internet. *Computer Communications Review*, 23(3):26-31, Julio 1993.

[Benson, 2001]

Benson, Christopher. 2001. Security Entities Building Block Architecture. Prentice Hall.

[Caja, 1982]

Valentín Sanz Caja. *Vulnerabilidad y seguridad de los sistemas informáticos*. Fundación Citema, 1982.

[Diffie, 1976]

W. Diffie and M. E. Hellman. New directions in cryptography.
IEEE Transactions on Information Theory, IT-22:644-654, Noviembre 1976.

[Garfinkel, 1996]

Simson Garfinkel and Eugene H. Spafford. *Practical Unix & Internet Security*. O'Reilly & Associates, 2nd edition. Abril 1996.

[Garfinkel, 1999]

Garfinkel, Simson and Spafford. Gene. Seguridad y Comercio en el Web, México, McGraw – Hill, 1999.

[Hecht, 1988]

Matthew S. Hecht, Abhai Johri, Radhakrishna Aditham, and T. John Wei. Experience adding C2 Security Features to Unix. In *USENIX Conference Proceedings*, pages 133- 146. The USENIX Association, Verano 1988.

[Heady, 1990]

Richard Heady, George Luger, Arthur Maccabe, and Mark Servilla.
The architecture of a Network Level Intrusion Detection System.
Technical Report CS90-20, University of New Mexico, Agosto 1990.

[Icove, 1995]

David Icove, Karl Seger, and William VonStorch. *Computer Crime. A Crimefighter's handbook*. O'Reilly & Associates, 1995.

[Kernighan, 1984]

Brian W. Kernighan and Rob Pike. *The Unix Programming Environment*.
Prentice Hall, 1984.

[Landwher, 1994]

Carl E. Landwher, Alan R. Bull, John P. McDermott, and William S. Choi. A taxonomy of computer program security flaws, with examples. *ACM Computing Surveys*, 3(26), Septiembre 1994.

[Lunt, 1990]

Teresa F. Lunt. Detecting Intruders in Computer Systems. In *Proceedings of the Sixth Annual Symposium and Technical Displays on Physical and Electronic Security*, 1990.

[Meyer, 1989]

Gordon R. Meyer. *The Social Organization of the Computer Underground*.
PhD thesis, Northern Illinois University, 1989.

[Olovsson, 1992]

Tomas Olovsson. A structured approach to computer security. Technical Report 122, Chalmers University of Technology, 1992.

[Parker, 1981]

Donn B. Parker. *Computer Security Management*. Prentice Hall, 1981.

[Ranum, 1995]

Marcus J. Ranum. Firewalls Frequently Asked Questions, 1995.

[Ritchie, 1986]

Dennis M. Ritchie. On the security of UNIX. In *UNIX System Manager's Manual, 4.3 BSD, Virtual VAX-11 Version*, pages 17:1-3. University of California, Berkeley, CA, Abril 1986.

[Serlin, 1991]

Omri Serlin. SVR4 may become the first truly secure Unix. *UNIXWORLD*, VIII(11):39- 40, Noviembre 1991.

[Schimmel, 1997]

John Schimmel. A historical look at firewall technologies. *;login.*, 22(1), Febrero 1997.

[Stoll, 1989]

Cliff Stoll. *The Cuckoo's Egg*. Doubleday, 1989.

[Schneier, 1998]

Bruce Schneier and John Kelsey. Cryptographic support for secure logs on untrusted machines. In *Proceedings of the 7th USENIX Security Symposium*. The USENIX Association, Enero 1998.

[Wack, 1994]

John P. Wack and Lisa J. Carnahan. Keeping your site comfortably secure: an introduction to Internet Firewalls. Technical report, National Institute of Standards and Technology (NIST), Diciembre 1994. Special Publication 800-10.

[Barzanallana, 2004]

Barzanallana, Rafael. La Seguridad en Informática, Informática Aplicada a la Gestión Pública.
http://dis.um.es/~barzana/Informatica/IAGP/IAGP_seguridad.html

[Ponce, 2004]

Ponce, Hugo. Seguridad Informática, Seguridad en Aplicaciones Web.
http://www.hispasec.com/directorio/servicios/formacion/SeguridadAplicaciones.NET

[Gallo, 2001]

Gallo, Gontzal. Delitos Informáticos, Artículos sobre Seguridad.
http://www.delitosinformaticos.com/seguridad/

[Romero, 1999]

Romero, Luis. Seguridad en el Web. Conceptos de acceso a la información.
http://cdec.unican.es/libro/SeguridadCGI.htm

[Castillo, 2004]

Castillo, Carlos. Protocolos Seguros para el Web. Seguridad en la Web.
http://www.tejedoresdelweb.com/307/article-5670.html

[Miguel, 1998]

Miguel A. R. H. Seguridad en Internet, Enlaces sobre Seguridad.
http://www.geocities.com/CapeCanaveral/2566/

[Martín, 2004]
Martín, Julio. Seguridad en Internet, Tipos de Ataques.
http://webs.ono.com/usr016/Agika/3internet/seg_internet.htm

[Anónimo, 2004]
Anónimo. Unix y Seguridad Informática, Seguridad: Criptología.
http://andercheran.aiind.upv.es/toni/

[Lizárraga, 2001]
Lizárraga, Carlos. Seguridad en Unix y Redes, Documentos sobre Seguridad.
http://www.fisica.uson.mx/carlos/Security/

[Maldonado, 2001]
Maldonado, Enrique. Criptografía y Seguridad en Computadores, Seguridad en Unix.
http://www.criptonomicon.com/documentos/2001/09102001.html

[Ripoll, 2002]
Ripoll, Ismael. Administración Segura de Sistemas Unix.
http://bernia.disca.upv.es/~iripoll/CFP-seguridad/links.html

[Sanctum, 2004]
Sanctum. Perversiones del Web.
http://www.sanctuminc.com/demo/webperversion/index.html

[Sanctum, 2004]
Sanctum. Seguridad en tu negocio.
http://www.sanctuminc.com/index.html

[VeriSign, 2004]

VeriSign. Soluciones de la seguridad informática.

http://www.verisign.com/

[Sanctum, 2004]

Sanctum. Soluciones de AppScan DE.

http://www.sanctuminc.com/solutions/appscande/index.html

[Sanctum, 2004]

Sanctum. AppScan De ayuda a la seguridad.

http://www.crn.com/sections/software/software.asp?ArticleID=46422