Adaptación de Técnicas Hacker para la impartición de Seguridad Informática

José María Alonso
Informática64, S.L
C/ Juan Ramón Jiménez 8,
28932 Madrid
chema@informatica64.com

Antonio Guzmán
DATCCIA
Universidad Rey Juan Carlos
c/Tulipán, s/n, 28933 Madrid
antonio.guzman@urjc.es

Rodolfo Bordón
Informática64, S.L.
C/ Juan Ramón Jiménez 8
28932 Madrid
rodol@informatica64.com

Alejandro Martín
Informática64, S.L.
C/ Juan Ramón Jiménez 8,
28932 Madrid
amartin@Informatica64.com

Resumen

El presente artículo describe como se ha utilizado la creación de diversos retos hacking para mostrar el funcionamiento de las técnicas hacker. El conocimiento de las mismas es imprescindible para formar auditores de seguridad que puedan profesionalmente abordar proyectos detección de vulnerabilidades en sistemas. Todos los retos se han utilizado también como fuente de retroalimentación ya que se ha podido evaluar el conocimiento de la comunidad en Internet, las herramientas y tácticas utilizadas e incluso descubrir cómo implementar nuevos mecanismos de auditoría. La actividad sigue realizándose a día de hoy pero se presentan los trabajos realizados durante el curso 2006-2007 dónde tuvo lugar la primera fase de esta experiencia didáctica.

1. Introducción

La sociedad de la información evoluciona muy rápidamente, asimilando nuevas funcionalidades en los sistemas, en las comunicaciones y en la manera que se gestiona la información. Esta rápida evolución está sustentada por enormes avances en la tecnología. Esta tecnología ha permitido aumentos en las capacidades de cómputo y tasas de transferencia, y un abaratamiento de los costes. Sin embargo, además de todas las ventajas innegables que presenta, también surgen nuevos desafíos que deben resolverse para asegurar que la evolución de la sociedad de la información sea consistente y coherente.

Uno de los principales problemas que surgen de la evolución en la tecnología es la excesiva especialización. A medida que aumenta la divergencia y concreción de las distintas ramas que constituyen la tecnología al servicio de la sociedad de la información (criptografía, sistemas operativos, bases de datos, programación de aplicaciones, etc.), se hace más difícil la consecución de un conocimiento multidisciplinar robusto. Aunque desde el punto de vista de la productividad y la formación de profesionales resulta conveniente, no lo es cuando aparece una disciplina compleja. Estas disciplinas complejas normalmente reciben resultados de dos o más de estas ramas de conocimiento, siendo un ejemplo muy característico la seguridad informática.

La seguridad informática es el conjunto de servicios, protocolos y mecanismos que aseguran la integridad y privacidad de la información que los sistemas manejan y transmiten, así como la integridad de los propios sistemas. El estudio de la seguridad informática obliga a tener en cuenta todos estos aspectos de forma indivisible.

El otro gran problema que conlleva el abaratamiento de costes y la masiva distribución de tecnología, cada día más potente, es la proliferación de ataques a todo tipo de sistemas, instituciones e individuos. La evolución del perfil del atacante ha sido tan rápida como la de la propia tecnología, ganando en eficacia, anonimato y sofisticación. Las alternativas clásicas para la habilitación de contramedidas frente a estos ataques se han manifestado con muchas carencias, avanzando siempre unos cuantos pasos detrás de los mecanismos utilizados por los atacantes.

En los últimos años se ha desarrollado una alternativa que propone la utilización de técnicas propias de los hackers para la evaluación y determinación del nivel de seguridad de los sistemas. Esta alternativa requiere una fuerte capacitación de los profesionales ya que, como se ha indicado antes, es preciso dominar aspectos, técnicas y procedimientos de diversas ramas, tanto tecnológicas como científicas.

En este artículo se plantea una experiencia metodología docente que explota esta idea de emplear técnicas hacker para describir cuales son

los principios básicos de este tipo de auditoría. La metodología pasa por tres fases claramente diferenciadas: la formación, el análisis y la detección. La etapa de formación comprende el estudio de los aspectos relacionados con el tipo de tecnología incluido en el escenario objetivo de cada técnica de ataque a estudiar. El análisis supone determinar los aspectos vulnerables de cada escenario de la misma forma en que lo haría un atacante y detallar las técnicas de hacking a emplear. Y, por último, en la fase de detección se plantea al alumno un caso práctico para desarrollar lo aprendido en las fases anteriores.

En este artículo se ha centrado la atención en la auditoría de sistemas para detectar vulnerabilidades en aplicaciones web a ataques de inyección de código. Se han desarrollado cuatro experiencias docentes para validar su aplicación en un entorno formativo controlado.

Esta experiencia se ha abierto públicamente en Internet para conseguir la mayor cantidad de información que se pueda para retroalimentar los conocimientos a priori puestos en marcha.

Experiencia docente

Una vez que se ha proporcionado al alumno la descripción de las tecnologías asociadas al escenario, se plantea el análisis de los posibles puntos en los que existe la posibilidad de encontrar vulnerabilidades y de las técnicas que un atacante podría utilizar para implementar una ataque sobre dicho escenario. A partir de aquí se les plantea la fase de detección. Para ello se propone un caso real en el que aplicando técnicas hacking tienen que ganar el acceso a la aplicación web que propone cada experiencia docente.

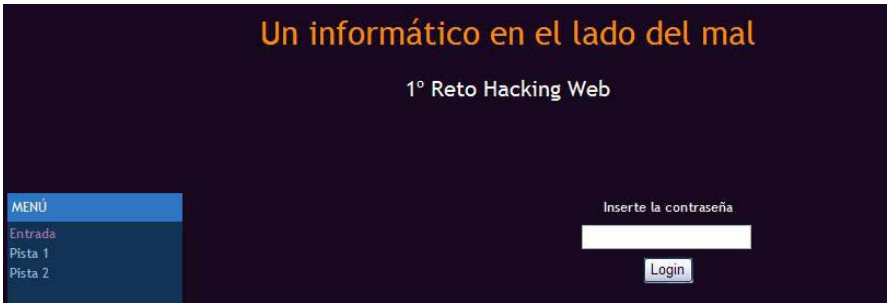

Figura 1. Reto Hacking I

1.1. Primera experiencia docente

La primera experiencia fue planteada en Noviembre del año 2006. El objetivo de este primer reto hacking, consistía en explicar y entender las técnicas de Blind SQL injection. Mediante el uso de estas técnicas los atacantes pueden extraer la información almacenada en motores de bases de datos relacionales mediante la inyección de código SQL que genera una página distinta ante la inyección de condiciones verdaderas y falsas. Así, el atacante puede extraer información de la base de datos con preguntas booleanas en SQL del tipo: "*¿Es la primera letra de la contraseña una 'a'*?".

Escenario

El escenario del primer reto hacking era simple. Una página web en la que se solicitaba la contraseña para superar el reto y una página con dos pistas. La primera pista indicaba mediante un texto de *"El lazarillo de Tormes"* que había que extraer la información como un ciego, es decir que nunca se iba a poder ver directamente el dato. La segunda pista indicaba que el motor de base de datos utilizado era Microsoft Access.

Técnicas hacking

Las técnicas hacking que se analizan en esta experiencia son:

- Blind SQL injection en base a cadenas de palabras.

Estas técnicas fueron descritas en profundidad por Cameron Hotchikies en su documento "Blind SQL injection automation techniques" [2].

Reto hacking I

La figura 1 muestra una captura del aspecto del reto [1]. Tiene la siguiente estructura:

- Base de datos Access.
- Tabla Pistas: Contiene los campos *id_pista* y *pista*. Es la que se consulta para mostrar cada uno de los mensajes accesibles por las pistas.
- Tabla *contrasena* con un único campo llamado contrasena que almacena el hash MD5 de la contraseña que soluciona el reto.
- *Pistas.aspx*: Programa vulnerable en .NET que recibe el parámetro id_pista con un número 1 o 2 para mostrar el texto de cada una de las pistas.
- *Id_pista* es el parámetro vulnerable.

Solución al reto

La solución al reto se alcanza mediante la inyección de código SQL que devuelva la pista 1 o la pista 2. Si devolvía la pista 1 la respuesta era falsa y si devolvía la pista 2 la respuesta es verdadera. Ejemplo:

Id_pista=2 and 'a'=(select
(mid(contrasena,1,1)) from contrasena)

Si se cumple que la primera letra es una 'a' entonces aparecía el texto de la pista 2. Si por el contrario no se cumplía entonces se devolvía el texto de la pista por defecto que era la pista 1.

Una vez obtenido el hash de la contraseña había que utilizar un sistema online de ruptura de hashes MD5 mediante tablas *rainbow* para acceder al valor de la contraseña.

Completando la proposición de este reto se publican las soluciones para facilitar la realimentación de aquellos alumnos que hubieran intentado la experiencia [3].

Resultados de la experiencia

A día de hoy ha sido superado por más de 40 personas en Internet y se han recibido más de 300.000 intentos de ataque. Todos los ataques han sido analizados y han permitido reutilizar el conocimiento aprendido para generar una metodología de trabajo que permita analizar la seguridad de una aplicación web ([4] y [5]).

Figura 2. Reto Hacking II

1.2. Segunda experiencia docente

El segundo reto hacking fue planteado en febrero de 2007 y estaba formado por una aplicación en la que los usuarios debían entrar en la zona de administración para completar la experiencia.

Escenario

Este reto contaba con una pequeña aplicación dónde los usuarios debían registrarse. Una vez registrados en la aplicación accedían a un listado de eventos. Cada evento tenía asociado un fichero que representaba la agenda del evento. Los usuarios podían registrarse al evento y añadir algún comentario para recordar algo sobre él.

Técnicas hacking

Este reto está preparado para ser solucionado mediante el uso de SQL Injection para descargar ficheros de un servidor de bases de datos. Los jugadores debían hacer que en lugar de descargarse la agenda de un evento se descargase la SAM del servidor, es decir, el fichero con las cuentas del servidor. Una vez obtenido el fichero debían obtener la contraseña para poder terminar la fase.

Reto hacking II

La figura 2 muestra el aspecto del segundo reto hacking [6]. La arquitectura es la siguiente:

- Base de datos: SQL Server 2000.
- Zona privada: Los usuarios una vez que entran en el reto tienen acceso a una zona privada con dos programas especiales:
  - descargar.aspx?id=xxx. Dónde id es el parámetro vulnerable a SQL Injection dónde se debe realizar la inyección.
  - getComentario.aspx?id=xxx. Dónde se puede acceder a los comentarios que cada usuario ha puesto.
- Zona de administración: Lugar dónde se podía acceder sólo si se conseguía la cuenta del sistema.

Solución al reto

El programa *descargar.aspx* descarga un fichero de la ruta que obtiene en la base de datos, es decir, en el parámetro id recibe un identificador de fichero. Si ese identificador no corresponde a ningún fichero entonces la *select* empleada no obtiene ningún resultado:

Select ruta from ficheros where id=0

El atacante debía poner un comentario con la ruta del fichero que quería descargar mediante un registro a un evento y recordar el id del comentario. Después, en la llamada al programa *descargar.aspx* debía realizar la siguiente inyección SQL:

descargar.aspx?id=0 union select comentario from comentarios where comentarios.id=XXX

Esta inyección devolvía el fichero del sistema que estuviera descrito en el comentario. Los ficheros necesarios eran:

- c:\windows\repair\sam
- c:\windows\repair\system

Una vez descargados los ficheros se debía utilizar un crackeador de ficheros de contraseñas Windows para poder accede a los usuarios y contraseñas. La solución más sencilla era utilizar OphCrack, programa de software libre que viene acompañado de las tablas *rainbow* para obtener las contraseñas de los ficheros [7].

Resultados de la experiencia

La experiencia ha contado con más de 200 usuarios únicos con más de 200.000 intentos distintos de ataque. De esta experiencia se escribió un artículo que explicaba el funcionamiento de estas técnicas de SQL Injection.

El reto tenía una protección contra la inyección de cadenas de caracteres para así impedir que se realizara una inyección con la ruta completa, es decir, que nadie pudiera poner algo como:

descargar.aspx?id=0 union select 'c:\windows\repair\sam" from cualquier_tabla

Sin embargo, el ganador se saltó esta protección usando una secuencia de escape y escribiendo la ruta en Hexadecimal:

descargar.aspx?id=0 UNION SELECT char(99)+char(58)+char(92)+char(119)+char(105)+…

Otro resultado curioso fue la cantidad de escáneres de vulnerabilidades que se recogieron en los ficheros de log de este reto, pero, ninguno de los usuarios que los utilizaron, fueron capaces de solucionar el reto, lo que puso de manifiesto las carencias de los mismos en las auditorías de seguridad de caja negra.

Figura 3. Reto Hacking III

1.3. Tercera experiencia docente

En el mes de mayo de 2007 se planteó el tercer reto hacking [8].

En este caso representaba un banco y los participantes debían superar tres fases distintas para poder solucionar el reto.

Escenario

El reto representa una entidad bancaria en la que los usuarios en primer lugar deben saltar un proceso de registro.

En la segunda fase deben superar una tarjeta de coordenadas para poder pasar a poder realizar transferencias bancarias.

En la tercera y última fase los jugadores se descargan un programa ejecutable para realizar transferencias bancarias que debían crackear el código ensamblador para poder terminar el reto.

Técnicas hacking

En este reto se estudian las siguientes técnicas hacker:

- Fase 1: Xpath Injection. El sistema de login está basado en un fichero XML y debe ser inyectado para entrar sin usuario al banco.
- Fase 2: fuerza bruta. La tabla de coordenadas no es lo suficientemente grande y además no hay un sistema de bloqueo en número de intentos por lo que es posible realizar un ataque de fuerza bruta.
- Fase 3: cracking. El código cheque el usuario y la contraseña en local antes de poder realizar la transferencia bancaria por lo que el jugador

debe modificar el código del programa para poder saltarse esa comprobación.

Reto hacking III

El reto tiene el aspecto que se ve en la figura 3 y está formado tiene la siguiente estructura:

- Sistema de login: Trabaja con un fichero XML y se realiza la comprobación con el lenguaje Xpath Injection. Para que los jugadores se dieran cuenta de esto, en los comentarios de la página de login podía leerse lo siguiente:

  <!-- XML Validation -->

- Sistema de coordenadas. Está basado en una tarjeta de 10 x 10 columnas con valores posibles de 3 cifras.
- Programa de transferencias: Una vez que se accede a la tercera fase el sistema te descarga un ejecutable para realizar las transferencias.

Solución al reto

- Fase 1: validación XML. Las técnicas de Xpath Injection & Blind Xpath Injection están ampliamente descritas en el documento "Blind Xpath Injection"[9]. Bastaba con realizar una inyección del tipo:

  User: a' or 1=1 or 'a'='b
  Pass: k

- Fase 2: Tarjeta de coordenadas de 10 x 10 con 1000 valores posibles por celda. Al poderse realizar intentos infinitos un ataque para encontrar un primer valor funcional debería realizar (10 columnas x 10 filas x 1000 posibles valores por columna) / 100 valores correctos, es decir, que realizando 1000 intentos se accede, en media, a un valor útil que permita pasar a la siguiente fase.
- Fase 3: Programa de transferencias. Está escrito en ensamblador y bastaba con quitar la comprobación utilizando Ollydbg o cualquier otro desensamblador.

Una solución completa, escrita por uno de los jugadores está disponible de forma pública [10].

Resultados de la experiencia

Este reto estaba preparado para estudiar las técnicas de Xpath Injection, mostrar la importancia de las políticas de seguridad en la segunda fase y recordar las técnicas de cracking.

Los jugadores tuvieron que realizar un control de la caducidad de la sesión utilizando sus propios programas o utilizando algún programa pensado para la fuerza bruta de aplicaciones web.

Tras plantearse el ataque de Xpath Injection se comenzó a estudiar los ataques de Blind Xpath Injection para Xpath 2.0 que en el documento referido para solucionar este reto aparece como no implementado. Se realizó una implementación y se comprobó cómo sí se podía extraer la base de datos completa en XML.

Figura 4. Reto Hacking IV

1.4. Cuarta experiencia docente

En Septiembre de 2007 se puso en un reto hacking un trabajo de investigación que habíamos estado realizado durante el curso. Para ambientar el reto, y hacerlo más amigable se utilizó una serie de televisión en la que se utilizan técnicas hacker.

Escenario

El reto está formado por una supuesta terminal en la que la protagonista de la serie se conecta y debe entrar dentro del sistema. Para ello debe extraer en primer lugar el nombre del usuario del sistema y luego, utilizando un sistema de recuperación de contraseñas acceder a la password.

Técnicas hacking

En este reto se presentaba la técnica de Blind LDAP Injection, hasta el momento desconocida. Existía un documento antiguo sobre técnicas de LDAP Injection, pero lo allí descrito no funciona en los entornos LDAP más usados hoy en día, a saber: OpenLDAP y Microsoft ADAM. Además, la segunda fase estaba pensada para pasarla usando Hacking Google, es decir, realizando búsquedas avanzadas en Internet.

Reto hacking IV

El reto [11] tenía la estética que se muestra en la figura 4, y estaba formado por los siguientes componentes:

- Base de datos LDAP sobre Microsoft ADAM. En ella se almacenaba la contraseña del usuario y una lista de impresoras y terminales disponibles en distintas plantas.
- default.aspx?recurso=XX&planta=YY: Este programa cuenta con dos parámetros, ambos vulnerables a inyección LDAP que permiten mostrar el número de impresoras y/o terminales que están disponibles en cada planta.
- default.aspx?uid=ZZ: Dado el nombre de un usuario nos muestra la pregunta secreta. Si se contesta correctamente se accede a la contraseña que da acceso a superar el reto.

Solución al reto

Para pasar la primera fase había que realizar un ataque de Blind LDAP Injection, para ello se utilizaba el buscador de recursos. Utilizando el lenguaje LDAP se pueden construir inyecciones del tipo:

default.aspx?recurso=)(uid=a*))&planta=YYY*

Esa inyección devolverá todos los objetos que tenga un atributo *uid* cuyo valor empiece por la letra a. Si existe algún usuario que cumpla esa restricción la aplicación mostrará algún recurso. Si no, entonces no se obtendrá ningún recurso. Desplegando todas las letras se puede acceder al nombre del usuario simplemente comprobando si se obtienen recursos o no.

Para solucionar la segunda fase simplemente había que buscar en Internet una pregunta sobre la serie que ambientaba el reto [12].

Resultados de la experiencia

De este reto se extrajeron varias experiencias positivas que quedan reflejadas en la publicación "LDAP Injection & Blind LDAP Injection" [13].

La primera de ellas fue la no detección de la vulnerabilidad en código por ninguna herramienta de análisis estático de código, herramientas utilizadas hoy en día para buscar fallos de seguridad en código fuente.

La segunda es que ningún escáner de vulnerabilidades de caja negra fue capaz de detectar estas vulnerabilidades. Esto se debe a que la mayoría de los escáneres se basaban en los estudios existentes, que no son funcionales en los entornos actuales.

Una tercera optimización del ataque fue realizar la reducción del alfabeto a la hora de explotar la vulnerabilidad. Este proceso fue realizado por uno de los jugadores y consiste en desplegar las letras que conforman un valor, pero una vez descubiertas las letras que forman parte del dato. Un ejemplo sería:

Default.aspx?recurso=)(uid=*a*))&planta=*

Esto permite reducir el tiempo de extracción de información al conocer previamente las letras que forman parte del resultado.

Como última aportación un jugador realizó, en lugar de buscar la respuesta usando Google Hacking, un ataque de diccionario construyendo el diccionario con todos los subtítulos de la serie que ambientaba el reto.

Los resultados de este trabajo han sido aceptados para ser mostrados en las conferencias Blackhat Europe 2008, unas de las más importantes en el ámbito de la seguridad informática.

2. Conclusión

El uso de Retos Hacking para la enseñanza de técnicas hacker permite probar disciplinas de difícil enseñanza en entornos simulados y controlados.

Los retos se han demostrado como un estrategia motivadora de los alumnos que disfrutan realizando todas las pruebas.

La popularidad de los mismos, con más de un millón de intentos entre todos, permite descubrir, además, el conocimiento de la comunidad de Internet.

Otras de las ventajas es poder depurar todos los estudios de investigación sobre seguridad con la retroalimentación que aporta el poder contar con todos los ficheros de registro dónde se puede estudiar los pasos que van realizando los jugadores.

Después de estos retos tuvo lugar un evento gratuito para todos los participantes en los retos en los que se estudiaban las técnicas de los retos y como solucionarlos [14].

Después de estas experiencias se han realizado dos retos hacking más, que están dentro de las actividades del curso 2007-2008.

Referencias

[1] http://www.informatica64.com/retohacking.
[2] Hotchikies, Cameron, *Blind SQL Injection Automation Techniques*, Blackhat, 2004.
[3] http://elladodelmal.blogspot.com/2007/01/solucin-al-1er-reto-hacking-web-por.html
[4] Alonso, José María & Guzmán, Antonio, *Metodología para la Evaluación de la Seguridad de Aplicaciones Web frente a Ataques Blind SQL Injection*, URJC, 2007.
[5] Alonso, José María & Guzmán, Antonio, *Metodología para la Evaluación de la Seguridad de Aplicaciones Web frente a Ataques Blind SQL Injection*, CIBSI, 2007
[6] http://retohacking2.elladodelmal.com
[7] http://elladodelmal.blogspot.com/2007/04/solucin-al-2o-reto-hacking-web-por.html
[8] http://retohacking3.elladodelmal.com
[9] Klein, Amit, *Blind Xpath Injection,*2004.
[10] http://elladodelmal.blogspot.com/2007/06/solucin-al-reto-hacking-iii-por.html
[11] http://retohacking4.elladodelmal.com
[12] http://elladodelmal.blogspot.com/2007/10/solucion-reto-hacking-iv-por-mandingo.html
[13] Alonso,J.M.; Guzmán A., *LDAP Injection & Blind LDAP Injection*, URJC 2008.
[14] http://elladodelmal.blogspot.com/2007/11/asegurait-i-asegurate-que-ests-seguro.html