

<0p3n/>char(45)[53c]%2037h1c41%20h4ck3r



Modelamiento Básico del Hacker

Linux : Red

```
$ ifconfig
eth0      Link encap:Ethernet  HWaddr e8:9a:8f:9c:07:ac
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:40

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:32 errors:0 dropped:0 overruns:0 frame:0
          TX packets:32 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1880 (1.8 KB)  TX bytes:1880 (1.8 KB)

wlan0     Link encap:Ethernet  HWaddr 74:e5:0b:0c:65:88
          inet addr:192.168.1.170  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::76e5:bff:fe0c:6588/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:61198 errors:0 dropped:0 overruns:0 frame:0
          TX packets:59467 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:55840784 (55.8 MB)  TX bytes:15082545 (15.0 MB)

$ route -n
Kernel IP routing table
Destination        Gateway           Genmask          Flags Metric Ref    Use Iface
192.168.1.0        0.0.0.0          255.255.255.0    U        2      0      0 wlan0
169.254.0.0        0.0.0.0          255.255.0.0      U       1000    0      0 wlan0
0.0.0.0            192.168.1.1     0.0.0.0          UG        0      0      0 wlan0

$ cat /etc/resolv.conf
# Generated by NetworkManager
nameserver 8.8.8.8
nameserver 8.8.4.4
```

- Revisar configuración básica :
 - ifconfig
 - route -n
 - cat /etc/resolv.conf

Linux : Red

- Configurar interfaz de red
 - **Ifconfig interfaz direccion_ip netmask mascara**
 - Ejemplo : ifconfig eth0 192.168.1.120
- Configurar router default
 - **route add default gw ip_router**
 - Ejemplo : route add default gw 192.168.1.1
- Configurar servidores DNS :
 - **Editar el /etc/resolv.conf y colocar**
 - nameserver ip_DNS_1
 - nameserver ip_DNS_2

Linux : Red

- Ver estado de conexiones de red
 - **netstat -nap | more**
 - **netstat -nap | grep nombre_servicio**
 - Ejemplo : **netstat -nap | grep firefox**
- Listar tabla de enrutamiento
 - **route -n**

OSEH

OP3N-53C
37H1G41
H4CK3R

Linux

Unusual Processes and Services

Look at all running processes:

```
# ps -aux
```

Get familiar with "normal" processes for the machine. Look for unusual processes. Focus on processes with root (UID 0) privileges.

If you spot a process that is unfamiliar, investigate in more detail using:

```
# lsof -p [pid]
```

This command shows all files and ports used by the running process.

Unusual Accounts

Look in /etc/passwd for new accounts in sorted list by UID:

```
# sort -nk3 -t: /etc/passwd | less
```

Normal accounts will be there, but look for new, unexpected accounts, especially with UID < 500.

Also, look for unexpected UID 0 accounts:

```
# egrep ':0+: ' /etc/passwd
```

On systems that use multiple authentication methods:

```
# getent passwd | egrep ':0+: '
```

Look for orphaned files, which could be a sign of an attacker's temporary account that has been deleted.

```
# find / -nouser -print
```


Linux

The Network

/etc/network/interfaces	Network interface configuration.
ifup, ifdown [device]	Start, stop network interfaces according to files above.
/sbin/ip	Show and manipulate network interfaces and routing, needs iproute.
ssh -X user@host	Login at other machine.
scp files user@host: path	Copy files to other machine (and vice versa).

Daemons and System

/etc/init.d/file restart	Restart a service, system daemon.
/etc/init.d/file stop	Stop a service, system daemon.
/etc/init.d/file start	Start a service, system daemon.
halt, reboot, poweroff	Halts, reboots, shuts down system.
/var/log/	All log files are under this directory.
/etc/default/	Default values for many daemons and services.

Dpkg

dpkg -l [names]	List packages.
dpkg -l pkg.deb	Show package information.
dpkg -c pkg.deb	List contents of package file.
dpkg -S filename	Show which package a file belongs to.
dpkg -i pkg.deb	Install package files.
debsums	Audit check sums of installed packages, needs debsums.
dpkg-divert [options] file	Override a package's version of a file.

APT

apt-get update	Update packages listings from package repositories as listed in /etc/apt/sources.list. Required whenever that file or the contents of the repositories change.
apt-cache search search-string	Search packages and descriptions for search-string.
apt-cache policy package-names	Show versions and priorities of available packages.
apt-cache show package-names	Show package information incl. description.

APT

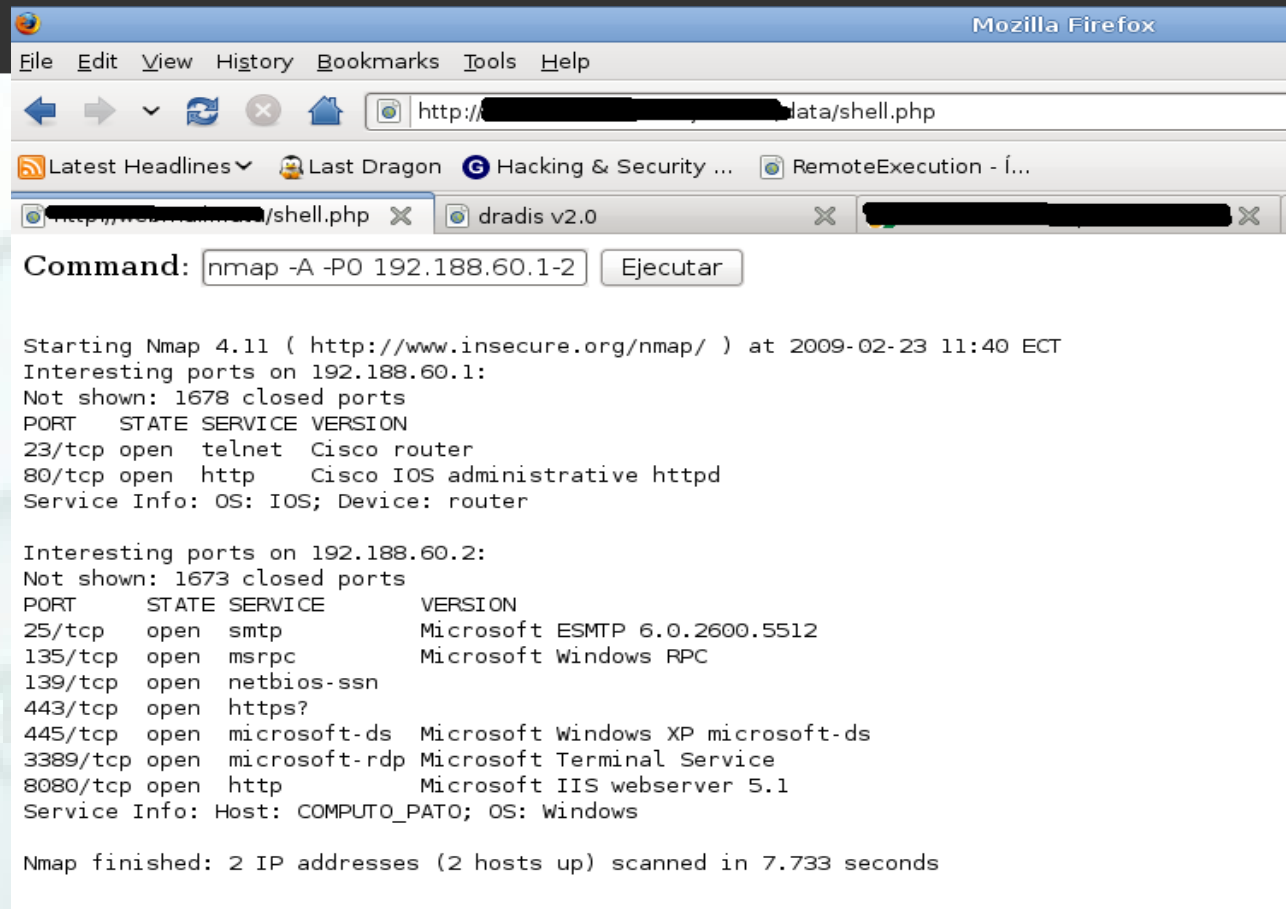
apt-cache showpkg package-names	Show package dependencies (needed packages).
apt-get install package-names	Install packages from repositories with all dependencies.
apt-get upgrade	Install newest versions of all packages currently installed.
apt-get dist-upgrade	Like apt-get upgrade , but with advanced conflict resolution.
apt-get remove package-names	Remove packages with all needed packages.
apt-get autoremove	Remove packages that no other packages depend on.
apt-cache depends package-names	List all packages needed by the one given.
apt-cache rdepends package-names	List all packages that need the one given.
apt-file update	Update content listings from package repositories, see apt-get update
apt-file search file-name	Search packages for file.
apt-file list package-name	List contents of a package.
auto-apt	Installs packages automatically if needed, can replace apt-file , needs auto-apt.
aptitude	Console interface to APT, needs aptitude.
synaptic	GUI interface to APT, needs synaptic.

OSEH

OP3N-53C
37H1G41
H4CK3R

Shell Scripting

Qué es un shell ?



```
Command: nmap -A -PO 192.188.60.1-2 [Ejecutar]

Starting Nmap 4.11 ( http://www.insecure.org/nmap/ ) at 2009-02-23 11:40 ECT
Interesting ports on 192.188.60.1:
Not shown: 1678 closed ports
PORT      STATE SERVICE VERSION
23/tcp    open  telnet  Cisco router
80/tcp    open  http    Cisco IOS administrative httpd
Service Info: OS: IOS; Device: router

Interesting ports on 192.188.60.2:
Not shown: 1673 closed ports
PORT      STATE SERVICE      VERSION
25/tcp    open  smtp        Microsoft ESMTTP 6.0.2600.5512
135/tcp    open  msrpc       Microsoft Windows RPC
139/tcp    open  netbios-ssn
443/tcp    open  https?
445/tcp    open  microsoft-ds Microsoft Windows XP microsoft-ds
3389/tcp    open  microsoft-rdp Microsoft Terminal Service
8080/tcp    open  http        Microsoft IIS webserver 5.1
Service Info: Host: COMPUTO_PATO; OS: Windows

Nmap finished: 2 IP addresses (2 hosts up) scanned in 7.733 seconds
```

Qué es un script ?

```
wcuestas@[redacted]:~$ cat tecsup.sh
ifconfig eth1 [redacted]
route add default gw [redacted]
cp /etc/resolv.conf /root/resolv.conf.antestecsup
echo "nameserver [redacted]" > /etc/resolv.conf
echo "nameserver [redacted]" >> /etc/resolv.conf
```


Bash Programming Cheat Sheet

- Lo básico :
 - Todos los scripts en Bash deben indicar al sistema operativo qué usar como interprete. La primera línea de cualquier script debería ser :
 - `#!/bin/bash`
 - Los scripts deben hacerse ejecutables :
 - `chmod +x filename`
- Variables :
 - Para crear una variable, solamente se le asigna un valor. Las variables no se basan en tipos, una variables puede contener caracteres, números, etc. Sin una definición previa.
 - `nombrevariable=valor`
 - Para acceder a una variable, colocar `$` delante del nombre de la misma :
 - `echo $nombrevariable`

Bash Programming Cheat Sheet

- Paso de parámetros :
 - comando var1 var2 var3 varX
 - \$1 contiene lo que era var1, \$2 contiene lo que era var2, etc.
 - Variables imbuídas :
 - \$1-\$N Almacenan los argumentos pasados al script desde la línea de comandos.
 - \$0 Almacena el nombre el script

Bash Programming Cheat Sheet

• Comillas

- Comillas Dobles (“como estas”) hace que se ignore el espacio en blanco y pasa como un solo argumento
- Comillas Simples ('como estas') hace que se ignoren los caracteres especiales
- Comillas Simples Invertidas (`comando`) hacen que la salida del comando sea enviada como parámetro. Ejemplo : listado=`ls -la`

• Lógica y Comparaciones

- El comando test es usado para evaluar expresiones condicionales como las usadas en las sentencias if-then
- test expresion
- O también se puede usar sencillamente : [expresion]

Bash Programming Cheat Sheet

- Comparaciones Numéricas
- `int1 -eq int2` Igual
- `int1 -ge int2` Mayor o Igual
- `int1 -gt int2` Mayor que
- `int1 -le int2` Menor o Igual
- `int1 -lt int2` Menor que
- `int1 -ne int2` No es igual

OSEH

OP3N-53C
37H1G41
H4CK3R

Bash Programming Cheat Sheet

- Comparaciones de Strings
 - `str1 = str2` Igual
 - `str1 != str2` Diferentes
 - `str` Devuelve verdadero sí no es nulo
 - `-n str` Devuelve verdadero sí el tamaño de `str` es mayor que cero
 - `-z str` Devuelve verdadero sí el tamaño de `str` es cero (cero no es igual a nulo).
- Comparaciones de Archivos
 - `-d filename` Devuelve verdadero sí `filename` es un directorio
 - `-f filename` Devuelve verdadero sí `filename` es un archivo
 - `-r filename` Devuelve verdadero sí `filename` puede ser leído
 - `-s filename` Devuelve verdadero sí `filename` tiene una longitud diferentes a cero
 - `-w filename` Devuelve verdadero sí `filename` puede ser escrito
 - `-x filename` Devuelve verdadero sí `filename` tiene permiso de ejecución

Bash Programming Cheat Sheet

•Comparación de Expresiones

- !expresion Devuelve verdadero si expresion no es verdadero
- expr1 -a expr2 Devuelve verdadero si expr1 y expr2 son verdaderos (&&, and)
- expr1 -o expr2 Devuelve verdadero si expr1 o expr2 son verdaderos (||, or)

•Más Lógica !

- If...then
 - if [expresion]
 - Then
 - comandos
 - fi

Bash Programming Cheat Sheet

• If..then...else

- if
[expresion]
- then
 - comandos
- else
 - comandos
- fi

-If..then...else If...else

- if [expresion]
- then
 - comandos
- elif [expresion2]
- then
 - comandos
- else
 - comandos
- fi

Bash Programming Cheat Sheet

- **Case select**
 - case string1 in
 - str1)
 - comandos;;
 - str2)
 - comandos;;
 - *)
 - comandos;;
 - esac

Bash Programming Cheat Sheet

- **Loops**

- for var1 in lista

- do
 - comandos
 - done

- while [expresion]

- do
 - comandos
 - done

- until [expresion]

- do
 - comandos
 - done

Bash Programming Cheat Sheet

- Funciones

- Crear una función

- `fname(){`
 - comandos
 - `}`

- Llamarla usando su nombre : `fname`

- Crear una función que acepta parámetros:

- `fname2 (arg1,arg2...argN){`
 - comandos
 - `}`

- Y llamarla : `fname2 arg1 arg2 ... argN`

OSEH

OP3N-53C
37H1G41
H4CK3R

Empecemos con algo simple en la LAN : ping y direcciones MAC

```
#!/bin/bash
```

She bang

```
fping -g $1 $2 > fping_lista.txt 2>/dev/null
```

Parámetros

```
grep "is alive" fping_lista.txt > activos.txt
```

```
echo "Lista de Direcciones IP Activas"
```

```
echo "-----"
```

```
for host in `cat activos.txt | cut -d " " -f 1`
```

```
do
```

```
    echo $host
```

```
done
```

```
arp -an | grep ether | cut -d " " -f 2,4,7
```

Loop "for"

Empecemos con algo simple en la LAN : ping y direcciones MAC

fping -g \$1 \$2 > fping_lista.txt 2>/dev/null

Salida y Error Estándar :

- 0 Entrada
- 1 Salida
- 2 Error

Parámetros :

- \$0 el programa mismo
- Los demás, del \$1 en adelante

Empecemos con algo simple en la LAN : ping y direcciones MAC

for host in `cat activos.txt | cut -d " " -f 1`

- For basado en una lista
- La lista se construye a partir de las entradas de un archivo, en este caso...
- Fijarse en las tildes invertidas `
 - Se ejecuta todo lo que esta dentro de las tildes invertidas
 - Eso devuelve la lista que requiere el For

OSEH

OP3N-53C
37H1G4I
H4CK3R

MS Windows

Process and Service Information

List all processes currently running:

```
C:\> tasklist
```

List all processes currently running and the DLLs each has loaded:

```
C:\> tasklist /m
```

Lists all processes currently running which have the specified [dll] loaded:

```
C:\> tasklist /m [dll]
```

List all processes currently running and the services hosted in those processes:

```
C:\> tasklist /svc
```

Query brief status of all services:

```
C:\> sc query
```

Query the configuration of a specific service:

```
C:\> sc qc [ServiceName]
```

```
tasklist /FI "USERNAME eq NT AUTHORITY\SYSTEM"
```

Reg Command

Adding Keys and Values:

```
C:\> reg add
```

```
[\\TargetIPAddr\] [RegDomain] \ [Key]
```

Add a key to the registry on machine

[TargetIPAddr] within the registry domain [RegDomain] to location [Key]. If no remote machine is specified, the current machine is assumed.

Export and Import:

```
C:\> reg export [RegDomain] \ [Key]
```

```
[FileName]
```

Export all subkeys and values located in the domain [RegDomain] under the location [Key] to the file [FileName]

```
C:\> reg import [FileName]
```

Import all registry entries from the file [FileName]

Import and export can only be done from or to the local machine.

Query for a specific Value of a Key:

```
C:\> reg query
```

```
[\\TargetIPAddr\] [RegDomain] \ [Key] /v  
[ValueName]
```

Query a key on machine [TargetIPAddr] within the registry domain [RegDomain] in location [Key] and get the specific value [ValueName] under that key. Add /s to recurse all values.

MS Windows

netstat

Displays information on the status of the network and established connections with remote machines.

Some options:

- a: To sample all the connections and listening ports
- n: to display addresses and port numbers in numeric form
- e: to sample Ethernet statistics

For example: netstat - an

Useful Netstat Syntax

Show all TCP and UDP port usage and process ID:

```
C:\> netstat -nao
```

Look for usage of port [port] every [N] seconds:

```
C:\> netstat -nao [N] | find [port]
```

Dump detailed protocol statistics:

```
C:\> netstat -s -p [tcp|udp|ip|icmp]
```

OSEH

OP3N-53C
37H1G4I
H4CK3R

MS Windows

Interacting with the Network Using Netsh

Turn off built-in Windows firewall:

```
C:\> netsh firewall set opmode disable
```

Configure interface "Local Area Connection" with
[IPaddr] [Netmask] [DefaultGW]:

```
C:\> netsh interface ip set address  
local static [IPaddr] [Netmask]  
[DefaultGW] 1
```

Configure DNS server for "Local Area Connection":

```
C:\> netsh interface ip set dns local  
static [IPaddr]
```

Configure interface to use DHCP:

```
C:\> netsh interface ip set address  
local dhcp
```

File Search and Counting Lines

Search directory structure for a file in a specific directory:

```
C:\> dir /b /s [Directory]\[FileName]
```

Count the number of lines on StandardOut of
[Command]:

```
C:\> [Command] | find /c /v ""
```

Finds the count (/c) of lines that do not contain (/v) nothing (""). Lines that do not have nothing are all lines, even blank lines, which contain CR/LF

Command Line FOR Loops

Counting Loop:

```
C:\> for /L %i in  
([start],[step],[stop]) do [command]
```

Set %i to an initial value of [start] and increment it by [step] at every iteration until its value is equal to [stop]. For each iteration, run [command]. The iterator variable %i can be used anywhere in the command to represent its current value.

Iterate over file contents:

```
C:\> for /F %i in ([file-set]) do  
[command]
```

Iterate through the contents of the file on a line-by-line basis. For each iteration, store the contents of the line into %i and run [command].

MS Windows

Listar usuarios locales :

Net user

Listar miembros de un grupo :

Net localgroup Administrators

Adicionar grupo local :

Net localgroup h4x0rs /add

“Escalar” privilegios :

Net user h4x0r p455w0rd /add

Net localgroup Administrators h4x0r /add

OSEH

OP3N-53C
37H1G41
H4CK3R

Linux	Windows
command --help	command /h, command /?
man command	help command
cp	copy
rm	del
mv	move
mv	ren
more, less, cat	type
lpr	print
rm -R	deltree
ls	dir
cd	cd
mkdir	md
rmdir	rd
route	route print
tracert -l	tracert
ping	ping
ifconfig	ipconfig

Scripting en MS Windows

- Acabamos de penetrar un host MS Windows y veo que puedo alcanzar otros hosts de la red
 - Quiero escanear puertos ?
 - Instalo nmap ?
 - Puedo usar copy con ?
 - Puedo mandarlo con msf !
 - En realidad es acerca de scripting, así que veamos uno que escanea puertos

Loop For + ftp = port scanner.bat

@echo off

- **for /L %%p in (20,1,82) do echo
Chequeando Puerto %%p: >> puertos.txt &
echo open 192.168.1.171 %%p > comftp.txt
& echo quit >> comftp.txt & echo quit >>
comftp.txt & echo quit >> comftp.txt & ftp
-s:comftp.txt 2>> puertos.txt**
- **Wtf ?!**

Loop For + ftp = port scanner.bat

for /L %%p in (20,1,82) do

- **Loop basado en contador (/L)**
- **(inicio, incremento, fin)**
- **%%p ira tomando los valores : el contador**

Loop For + ftp = port scanner.bat

```
echo Chequeando Puerto %%p: >> puertos.txt &
echo open 192.168.1.171 %%p > comftp.txt & echo
quit >> comftp.txt & echo quit >> comftp.txt & echo
quit >> comftp.txt & ftp -s:comftp.txt 2>> puertos.txt
```

- & ejecuta comando tras comando
- En puertos.txt estará el reporte
- En comftp.txt estarán las instrucciones que usará ftp para INTENTAR abrir una conexión
- (doble quit para prevenir servicios que reciben el comando, lo encuentran válido o errado y se quedan esperando otro comando)

comftp.txt

```
open 192.168.1.171 28
quit
quit
quit
```

Volvamos a Linux : Port Scan “a mano”

- Acabamos de penetrar un host Linux y veo que puedo alcanzar otros hosts de la red
 - Quiero escanear puertos ?
 - Instalo nmap ?
 - Puedo usar echo ?
 - Puedo mandarlo con msf !
 - En realidad es acerca de scripting, así que veamos uno que escanea puertos

Loop While + /dev/tcp = portscanner.sh

```
#!/bin/bash
port=1
echo "Puertos abiertos en $1" > $2
echo "===== " >> $2
while [ $port -le 1024 ]
do
    (echo > /dev/tcp/$1/$port) 2>/dev/null
    if [ $? = 0 ]
    then
        echo "El puerto $port esta abierto" >> $2
    fi
    port=`expr $port + 1`
done
```

- ./portscanner.sh direccion_ip nombre_reporte

Loop While + /dev/tcp = portscanner.sh

while [\$port -le 1024]

- Loop basado expresión y comparación numérica
- -le ---> menor o igual
- Olvidense de los puertos privilegiados...

Loop While + /dev/tcp = portscanner.sh

```
(echo > /dev/tcp/$1/$port) 2>/dev/null
if [ $? = 0 ]
then
    echo "El puerto $port esta abierto" >> $2
fi
port=`expr $port + 1`
```

- /dev/tcp ---> pseudo-dispositivo para networking que bash los usa como cualquier otro dispositivo (device)
 - Pseudo-device --> no existe dispositivo como tal
- El mensaje de error que se puede producir por un puerto cerrado aparece en línea, no se puede redireccionar como stdout ni como stderr, PERO, se puede invocar un subshell y dejar “limpio” el stdout.
- expr evalúa y ejecuta expresiones : contador

```
# Echo Servidor
#Ejecutar : python server.py
import socket
import hashlib
```

```
HOST = "          # Nombre simbolico para 0.0.0.0
PORT = 50007      # Puerto no privilegiado, arbitrario
```

```
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((HOST, PORT))
s.listen(1)
while 1:
    conn, addr = s.accept()
    print 'Se conecto la maquina', addr
    while 1:
        data = conn.recv(1024)
        if not data: break
        h = hashlib.md5()
        h.update(data)
        #print h.hexdigest()
        conn.send("El MD5 Hash de
conn.close()
```

```
# Echo Cliente
# Ejecutar : python client.py "Hola Mundo"
import socket,sys
```

```
HOST = '127.0.0.1' # El host remoto
PORT = 50007      # El puerto en que escucha el server
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((HOST, PORT))
s.send(sys.argv[1])
data = s.recv(1024)
s.close()
print 'Recibiendo desde el Servidor...', repr(data)
```

Python usando HTTP

```
#!/usr/bin/python
```

El modulo que hace toda la magia

```
import urllib2
```

Defino la URL

```
headers = { 'User-Agent' : 'Mozilla/5.0' }
```

```
solicitud = urllib2.Request('http://www.google.com/search?  
q=python+es+la+voz',None, headers)
```

Realizo la solicitud

```
respuesta = urllib2.urlopen(solicitud)
```

```
payload = respuesta.read()
```

Leo la respuesta del
servidor web

```
print(payload)
```

Imprimo en Pantalla

Sentencias SQL

SAMPLE SELECT QUERIES

SELECT * FROM tablename	# Returns all columns
SELECT column FROM tablename	# Returns specific column
SELECT COUNT(*) FROM tablename	# Returns number of rows
SELECT SUM(column) FROM tablename	# Returns sum of column
SELECT DISTINCT column FROM tablename	# Returns unique values of column
SELECT * FROM tablename WHERE condition	# Returns rows that match condition
SELECT * FROM tablename WHERE BINARY condition	# Condition is case-sensitive
SELECT * FROM table1 INNER JOIN table2 on table1.id = table2.id	# Join two tables, return all columns
SELECT table1.* FROM table1 INNER JOIN table2 on table1.id = table2.id	# Only return columns from table1
SELECT LAST_INSERT_ID() as new_id	# Returns ID of last created row
SELECT max(column) AS alias	# Return maximum value in column as "alias"
SELECT * FROM table ORDER BY column	# Return all rows ordering by column
SELECT * FROM table LIMIT 10, 20	# Return first 20 rows after row 10

Compilación de Programas

- `tar xvzf fuente.tar.gz`
- `cd fuente`
- `./configure`
- `make`
- `make install` (opcional)