Practica09

Auditoría de seguridad en WP

Creación de una máquina con Kali Linux

Vamos a crear una máquina con Kali Linux para poder utilizar el WPScan.

Creamos nuestros directorios para la elaboración de la práctica:

```
mkdir practica09
cd practica09
```

Creamos el Vagrant file ndicando el nombre del box que vamos a utilizar, que ya tiene instalado Kali Linux:

En nuestro caso al ser Windows, utilizaremos el `gitbash` para utilizar la consola de linux:

```
vagrant init Sliim/kali-2017.2-light-amd64 --box-version 1
```

El paso siguiente será descargar la clave privada ssh para poder controlar nuestra máquina de Kali Linux pero, antes de eso vamos a añadir la funcionalidad `wget` a nuestro `gitbash`. El proceso es el siguiente:

- Desscargar el ultimo `wget binary` para Windows desde https://eternallybored.org/misc/wget/.
- Extraer el zip.
- Renombrar el archivo a `wget.exe`
- Mover `wget.exe` al directorio `Git\mingw64\bin\`.

Una vez descargado podremos descargar la clave:

```
wget https://raw.githubusercontent.com/Sliim/pentest-env/master/ssh-keys/pentest-env
```

Y justo despues haremos un `vagrant up` y `vagrant ssh` para acceder a neustra máquina.

WPScan.

La herramienta que vamos a utilizar para realizar la auditoría de nuestro sitio web WordPress es wpscan.

```
root@kali:~# wpscan  --help
_______________________________________________________________
        __          _______   _____
        \ \        / /  __ \ / ____|
         \ \  /\  / /| |__) | (___   ___  __ _ _ __
          \ \/  \/ / |  ___/ \___ \ / __|/ _` | '_ \
           \  /\  /  | |     ____) | (__| (_| | | | |
            \/  \/   |_|    |_____/ \___|\__,_|_| |_|

        WordPress Security Scanner by the WPScan Team
                       Version 2.6
          Sponsored by Sucuri - https://sucuri.net
   @_WPScan_, @ethicalhack3r, @erwan_lr, pvdl, @_FireFart_
_______________________________________________________________

Help :

Some values are settable in a config file, see the example.conf.json

--update                            Update to the database to the latest version.
--url       | -u <target url>       The WordPress URL/domain to scan.
--force     | -f                    Forces WPScan to not check if the remote site
is running WordPress.
--enumerate | -e [option(s)]        Enumeration.
  option :
    u        usernames from id 1 to 10
    u[10-20] usernames from id 10 to 20 (you must write [] chars)
    p        plugins
    vp       only vulnerable plugins
    ap       all plugins (can take a long time)
    tt       timthumbs
    t        themes
    vt       only vulnerable themes
    at       all themes (can take a long time)
  Multiple values are allowed : "-e tt,p" will enumerate timthumbs and plugins
  If no option is supplied, the default is "vt,tt,u,vp"

--exclude-content-based "<regexp or string>"
                                    Used with the enumeration option, will exclude
all occurrences based on the regexp or string supplied.
                                    You do not need to provide the regexp
delimiters, but you must write the quotes (simple or double).
--config-file  | -c <config file>   Use the specified config file, see the
example.conf.json.
--user-agent   | -a <User-Agent>    Use the specified User-Agent.
--cookie <String>                   String to read cookies from.
--random-agent | -r                 Use a random User-Agent.
--follow-redirection                If the target url has a redirection, it will
be followed without asking if you wanted to do so or not
--batch                             Never ask for user input, use the default
behaviour.
--no-color                          Do not use colors in the output.
--wp-content-dir <wp content dir>   WPScan try to find the content directory (ie
wp-content) by scanning the index page, however you can specified it.
                                    Subdirectories are allowed.
--wp-plugins-dir <wp plugins dir>   Same thing than --wp-content-dir but for the
plugins directory.
                                    If not supplied, WPScan will use wp-content-
```

```
dir/plugins. Subdirectories are allowed
--proxy <[protocol://]host:port>    Supply a proxy. HTTP, SOCKS4 SOCKS4A and
SOCKS5 are supported.
                                    If no protocol is given (format host:port),
HTTP will be used.
--proxy-auth <username:password>    Supply the proxy login credentials.
--basic-auth <username:password>    Set the HTTP Basic authentication.
--wordlist | -w <wordlist>          Supply a wordlist for the password brute
forcer.
--username | -U <username>          Only brute force the supplied username.
--usernames     <path-to-file>      Only brute force the usernames from the file.
--threads  | -t <number of threads> The number of threads to use when multi-
threading requests.
--cache-ttl       <cache-ttl>       Typhoeus cache TTL.
--request-timeout <request-timeout> Request Timeout.
--connect-timeout <connect-timeout> Connect Timeout.
--max-threads     <max-threads>     Maximum Threads.
--help     | -h                     This help screen.
--verbose  | -v                     Verbose output.
--version                           Output the current version and exit.


Examples :

-Further help ...
ruby ./wpscan.rb --help

-Do 'non-intrusive' checks ...
ruby ./wpscan.rb --url www.example.com

-Do wordlist password brute force on enumerated users using 50 threads ...
ruby ./wpscan.rb --url www.example.com --wordlist darkc0de.lst --threads 50

-Do wordlist password brute force on the 'admin' username only ...
ruby ./wpscan.rb --url www.example.com --wordlist darkc0de.lst --username admin

-Enumerate installed plugins ...
ruby ./wpscan.rb --url www.example.com --enumerate p

-Enumerate installed themes ...
ruby ./wpscan.rb --url www.example.com --enumerate t

-Enumerate users ...
ruby ./wpscan.rb --url www.example.com --enumerate u

-Enumerate installed timthumbs ...
ruby ./wpscan.rb --url www.example.com --enumerate tt

-Use a HTTP proxy ...
ruby ./wpscan.rb --url www.example.com --proxy 127.0.0.1:8118

-Use a SOCKS5 proxy ... (cURL >= v7.21.7 needed)
ruby ./wpscan.rb --url www.example.com --proxy socks5://127.0.0.1:9000
```

```
-Use custom content directory ...
ruby ./wpscan.rb -u www.example.com --wp-content-dir custom-content

-Use custom plugins directory ...
ruby ./wpscan.rb -u www.example.com --wp-plugins-dir wp-content/custom-plugins

-Update the DB ...
ruby ./wpscan.rb --update

-Debug output ...
ruby ./wpscan.rb --url www.example.com --debug-output 2>debug.log

See README for further information.
```