

Segurança Informática e nas Organizações

Projeto 1: Exploração de Vulnerabilidade

**Autores:**

André Gual nº 88751

João Laranjo nº 91153

# 

[**Objetivos**](#_rebp3rr3ft6y) **3**

[**Introdução**](#_dnuvd0ot1oh2) **3**

[O que são CVE:](#_58g36ivj0u6m) 3

[O que é uma “Vulnerability”:](#_kjtdw2zg08e) 3

[O que é uma “Exposure”:](#_7o55kub1fb6x) 3

[O que é um identificador CVE:](#_dfnfc11eps1e) 3

[**Preparação**](#_b6cum2wmitb7) **4**

[**Trabalho**](#_azxln3ddqs9i) **5**

[Enumere os portos abertos, descrevendo a sua funcionalidade.](#_nx85q45j2hm8) 5

[Enumere o sistema operativo e aplicações disponíveis na máquina, incluindo versões de aplicações. Descreva a função de cada aplicação e valide que se encontram em operação.](#_7gatvto1j7es) 6

[Enumere e descreva potenciais vulnerabilidades encontradas nos serviços, com nível superior a 7 (http://cvedetails.com). Consegue validar a existência de alguma vulnerabilidade?](#_a1g56nd8ptr6) 7

[Analise a página web e descreva potenciais vulnerabilidades encontradas. Existe um leque variado, relacionadas com validações de entrada, autorizações, más configurações, más práticas.](#_fmskzlqqzdqw) 8

[Login Bypass (Sql injection)](#_lltaywiwtsww) 8

[Sql map](#_7b697eusbmw7) 9

[Sql injection através do url](#_vbfsiejolhhi) 11

[Nikto Scan](#_45hjusrxcunf) 13

[Obtenção de credenciais da base de dados](#_l9579y63v3vd) 13

[XSS](#_h4x1jvmdfdgh) 15

[Explore as vulnerabilidades encontradas, descrevendo cada passo tomado, a razão dela existir e qual o potencial impacto.](#_fn7ostde8b1e) 16

[Crie uma ferramenta (script python) para automatizar os passos que levam à exploração da vulnerabilidade mais grave que encontrar.](#_dxuo12ecucc5) 16

[A primeira opção permite fazer update dos dados da conta do administrador ( nome e password).](#_ral7yoxvda3r) 17

[A segunda opção permite fazer um novo post no blog em nome do administrador. Visto que o utilizador pode inserir o conteúdo que quiser, poderá inserir conteúdo que irá ser representado na forma de XSS cross-site scripting.](#_wol1mn7v52oj) 17

[A última opção permite ao utilizador obter o ficheiro config.php que contém as credenciais da base de dados utilizada no website.](#_5u2qymo0za3v) 17

[O script utiliza as bibliotecas mechanize e urllib para fazer pedidos http e obter informação do website.](#_29g5uy3fejya) 17

[**Extra**](#_tj6xyclg5xdp) **17**

[**Bibliografia**](#_bbuo2n786tof) **18**

# 

# Objetivos

* Determinação de vulnerabilidades existentes.
* Determinação do risco e impacto.
* Exploração de vulnerabilidades.

# Introdução

Com a elaboração deste projeto temos como objetivo principal a exploração dos CVE (Common Vulnerabilities and Exposures) existentes na página de uma pequena loja.

Tanto página como sistemas associados possuem várias e variadas vulnerabilidades.

## O que são CVE:

CVE ou Common Vulnerabilities and Exposures é uma lista de vulnerabilidades e exposições. O principal objetivo do CVE é facilitar a partilha de informação relacionada com problemas de vulnerabilidades mantendo uma “enumeração comum”.

## O que é uma “Vulnerability”:

Uma vulnerabilidade (em segurança informática) é um erro no software que pode ser usado diretamente por um atacante (“hacker”) para obter acesso a um sistema ou rede.

## O que é uma “Exposure”:

Uma exposição (em segurança informática) é um erro no software que permite o acesso a informações e ou recursos que podem ser explorados por um atacante (“hacker”) como ferramenta para aceder ao sistema ou rede.

## O que é um identificador CVE:

Também conhecidos por nomes, números ou IDs, os identificadores CVE são únicos e servem para identificar vulnerabilidades de segurança publicamente conhecidas.

Cada identificador CVE inclui:

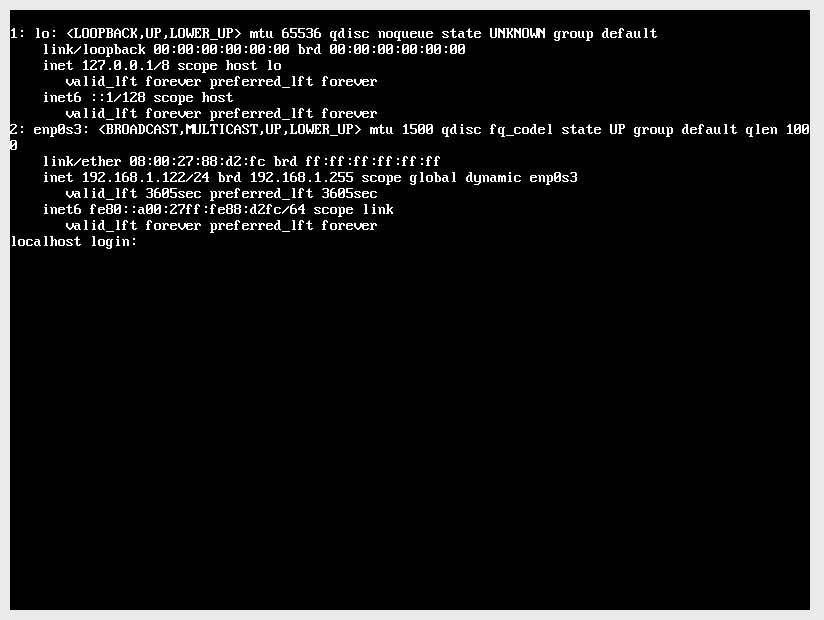
* Número ("CVE-1999-0067")
* Indicação do estado ("Entry", "Candidate")
* Breve descrição
* Referências
* São também utilizados como método standard por produtores, vendedores e investigadores para identificar vulnerabilidades.

# Preparação

Foi obtida a imagem da máquina virtual disponibilizada na página da disciplina.

Após criada a máquina virtual foi ligada em modo Bridge ou HostOnly.

(De notar que os IPs da máquina virtual variaram para as várias sessões de desenvolvimento do projeto.)



# Trabalho

## Enumere os portos abertos, descrevendo a sua funcionalidade.

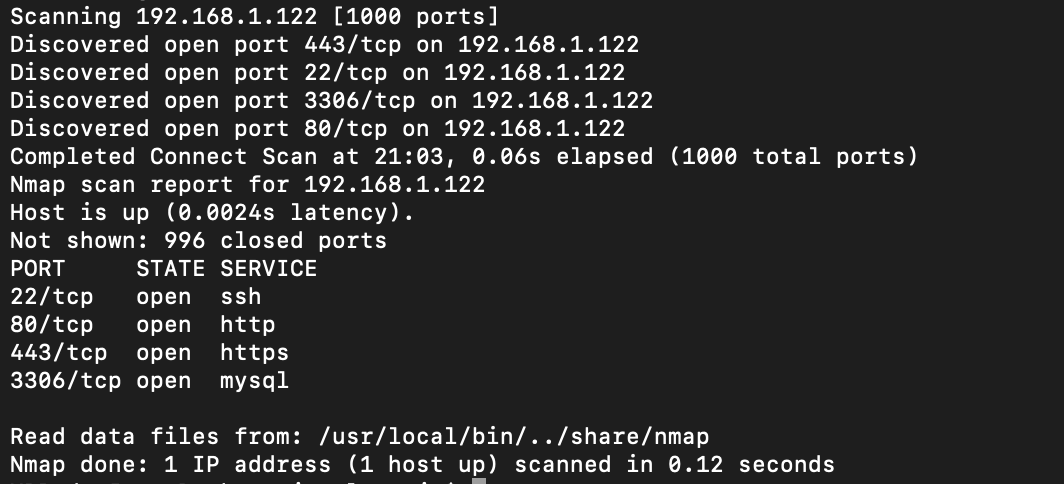
Um porto é um “endpoint” de comunicação. Ao nível de software, num sistema operativo, um porto é uma saída/ligação que identifica um processo ou um tipo de serviço network. Portos são definidos por um protocolo e um endereço conhecido como número do porto.

Com auxílio da ferramenta nmap foram encontrados os seguintes portos abertos:

*nmap 192.168.1.122*



*nmap -v 192.168.1.122*



# 

# 

Conseguimos identificar os seguintes portos:

* 22 - Protocolo TCP - Serviço SSH - Secure Shell - Protocolo de rede criptográfico que fornece uma ligação segura entre cliente-servidor.
* 80 - Protocolo TCP - Serviço HTTP - HTTP é um protocolo de comunicação, é também a base para a troca de informação da WWW.
* 444 - Protocolo TCP - Serviço HTTPS - HTTPS é uma implementação do [protocolo](https://pt.wikipedia.org/wiki/Protocolo_(ci%C3%AAncia_da_computa%C3%A7%C3%A3o)) [HTTP](https://pt.wikipedia.org/wiki/HTTP) sobre uma camada adicional de segurança. Permite que os dados sejam transmitidos de forma [criptografada](https://pt.wikipedia.org/wiki/Criptografia) e que se verifique a autenticidade do [servidor](https://pt.wikipedia.org/wiki/Servidor) e do cliente por meio de [certificados digitais](https://pt.wikipedia.org/wiki/Certificados_digitais).
* 3306 - Protocolo TCP - Serviço MYSQL é o porto default para o protocolo MySQL que é usado pelo MySQL Client. SGBD MySQL.

## Enumere o sistema operativo e aplicações disponíveis na máquina, incluindo versões de aplicações. Descreva a função de cada aplicação e valide que se encontram em operação.

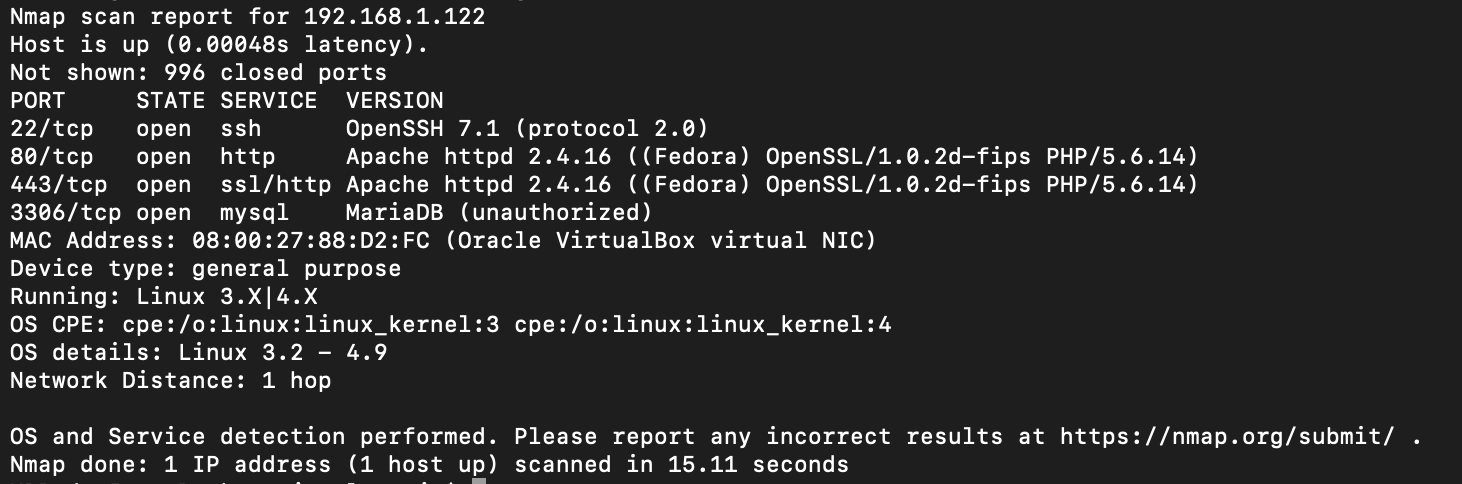
Através da ferramenta *nmap* e do comando que se segue conseguimos identificar as versões para cada uma das aplicações disponíveis assim como do sistema operativo.

A função de cada aplicação já foi previamente descrita (ponto 1 deste trabalho).

Podemos também verificar que se encontram através da coluna “STATE” para a qual todas apresentam o estado “open”.

Host is Up.

*sudo nmap -sV -O 192.168.1.122*



## Enumere e descreva potenciais vulnerabilidades encontradas nos serviços, com nível superior a 7 (http://cvedetails.com). Consegue validar a existência de alguma vulnerabilidade?

Novamente, com o auxílio do nmap e com os scripts nmap-vulners e vulscan foram identificadas e classificadas as seguintes vulnerabilidades:

*nmap -sV --script vulners* [*192.168.1.106*](http://192.168.1.106/)

*nmap -sV --script=vulscan/vulscan.nse 192.168.1.106*

Nota: Resultaram mais vulnerabilidades no entanto não foram listadas por apresentarem um CVSS Score (nível) superior a 7. Segue em anexo todo o output nos ficheiros:

*nmap-vulscan.txt*

*nmap-vulners.txt*

Nmap scan report for 192.168.1.106

Host is up (0.00062s latency).

Not shown: 996 closed ports

PORT STATE SERVICE VERSION

22/tcp open ssh OpenSSH 7.1 (protocol 2.0)

| vulners:

| cpe:/a:openbsd:openssh:7.1:

|\_ CVE-2016-8858 7.8 <https://vulners.com/cve/CVE-2016-8858>

80/tcp open http Apache httpd 2.4.16 ((Fedora) OpenSSL/1.0.2d-fips PHP/5.6.14)

|\_http-server-header: Apache/2.4.16 (Fedora) OpenSSL/1.0.2d-fips PHP/5.6.14

| vulners:

| cpe:/a:apache:http\_server:2.4.16:

| CVE-2017-7679 7.5 <https://vulners.com/cve/CVE-2017-7679>

| CVE-2017-7668 7.5 <https://vulners.com/cve/CVE-2017-7668>

| CVE-2017-3169 7.5 <https://vulners.com/cve/CVE-2017-3169>

|\_ CVE-2017-3167 7.5 <https://vulners.com/cve/CVE-2017-3167>

443/tcp open ssl/http Apache httpd 2.4.16 ((Fedora) OpenSSL/1.0.2d-fips PHP/5.6.14)

|\_http-server-header: Apache/2.4.16 (Fedora) OpenSSL/1.0.2d-fips PHP/5.6.14

| vulners:

| cpe:/a:apache:http\_server:2.4.16:

| CVE-2017-7679 7.5 <https://vulners.com/cve/CVE-2017-7679>

| CVE-2017-7668 7.5 <https://vulners.com/cve/CVE-2017-7668>

| CVE-2017-3169 7.5 <https://vulners.com/cve/CVE-2017-3169>

|\_ CVE-2017-3167 7.5 <https://vulners.com/cve/CVE-2017-3167>

3306/tcp open mysql MariaDB (unauthorized)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .

Nmap done: 1 IP address (1 host up) scanned in 13.27 seconds

## Analise a página web e descreva potenciais vulnerabilidades encontradas. Existe um leque variado, relacionadas com validações de entrada, autorizações, más configurações, más práticas.

Em relação à página web primeiramente fizemos uma análise detalhada a todas as páginas acessíveis. Encontramos facilmente uma vulnerabilidade à primeira vista.

Uma das vulnerabilidades é o facto do email da conta “admin”, que supomos que seja a conta do administrador, estar presente numa das páginas.



Consideramo-lo uma vulnerabilidade com algum grau de gravidade porque visto que a autenticação do website é feita através de um email e de uma password, sendo que o email da conta admin é dado só será necessário fazer o bypass da password do mesmo.

### Login Bypass (Sql injection)

Seguidamente decidimos testar o formulário de autenticação do website. Rapidamente descobrimos que o formulário era vulnerável a SQL injection (injeção de comandos sql).

Encontrámos duas formas de fazer o bypass do login.

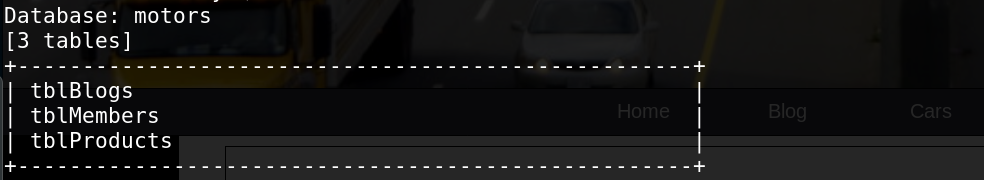
A primeira forma seria simplesmente colocar o email do admin ([admin@expressmotors.net](mailto:admin@expressmotors.net)) e no campo da password injetar o código “ ‘ or 1=1 -- ‘.

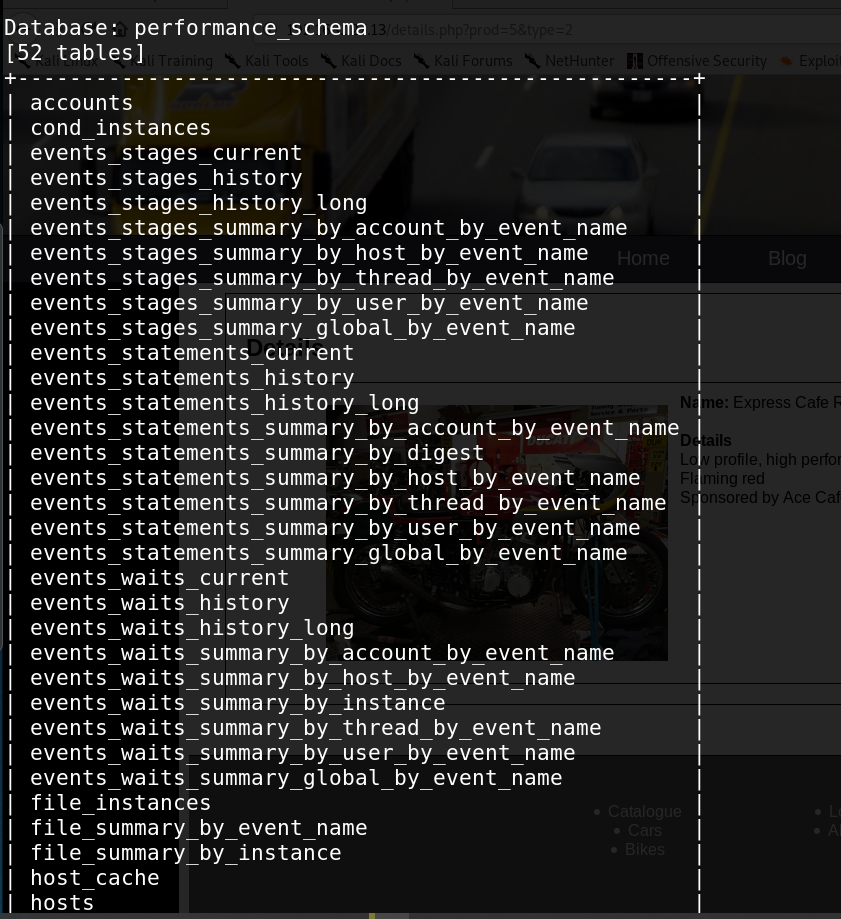
Para a segunda forma reparámos que os campos do formulário apenas tinham validação do lado do cliente (html). Desta forma para não utilizarmos qualquer tipo de email era também possível remover a validação de email, simplesmente indo à developer console e alterar o tipo de input no html para text. O input passa então a aceitar qualquer caractere. Assim agora podemos simplesmente injetar o código ‘ or 1=1 -- ‘ no input do email e inserir qualquer caractere no input da password. O login é ultrapassado com sucesso.

### Sql map

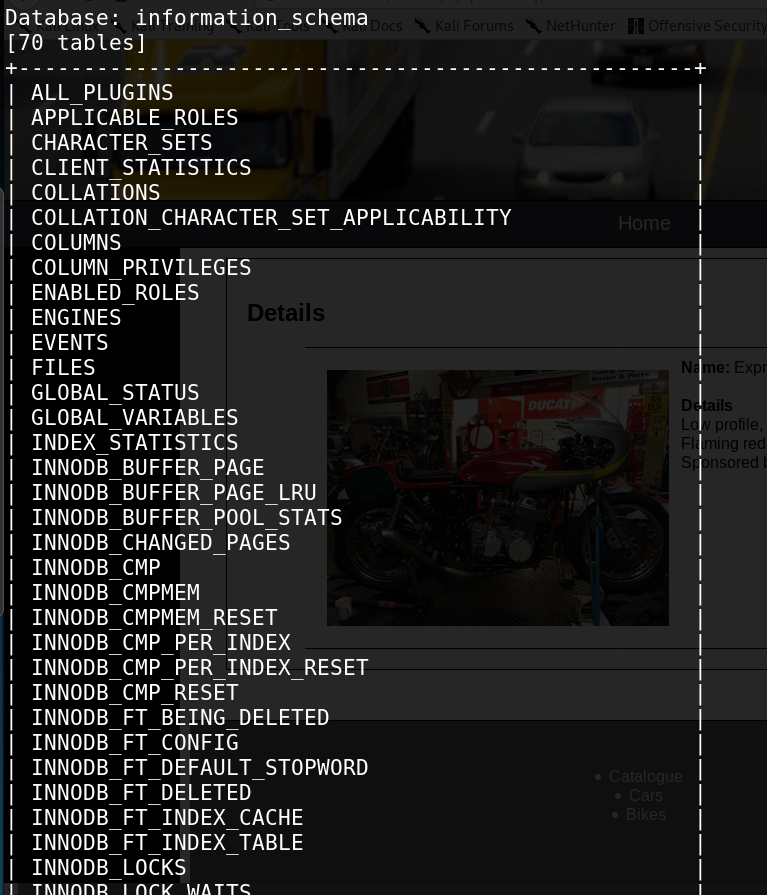
Seguidamente decidimos correr a ferramenta sqlmap. Esta permite a automatização do processo de procura de vulnerabilidades SQL injection e recolha de informação da base de dados. Ao corrermos o seguinte comando conseguimos obter todas as bases de dados e tabelas de cada uma delas existentes no servidor:

*sqlmap -u <URL> --data usermail=”or 1=1” --tables*





…

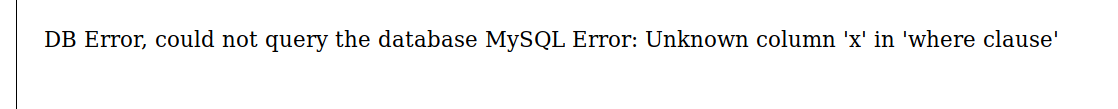


### Sql injection através do url

Encontrámos também forma de injetar código sql no url e obter informações relativas à base de dados.

O url *http://192.168.127.14/details.php?prod=<prod\_num>&type=<type\_num>* é vulnerável a injeções de código através do argumento para o pedido get <prod\_num> ao php. Ao variar o número tanto dos pedidos, é apresentada informação diferente na página.

Para além disso ao darmos um produto não existente a página retorna um erro que nos diz que não é encontrada a coluna x na cláusula where. Isto significa que está a ser efetuada uma query do tipo “where id=<prod\_num>”

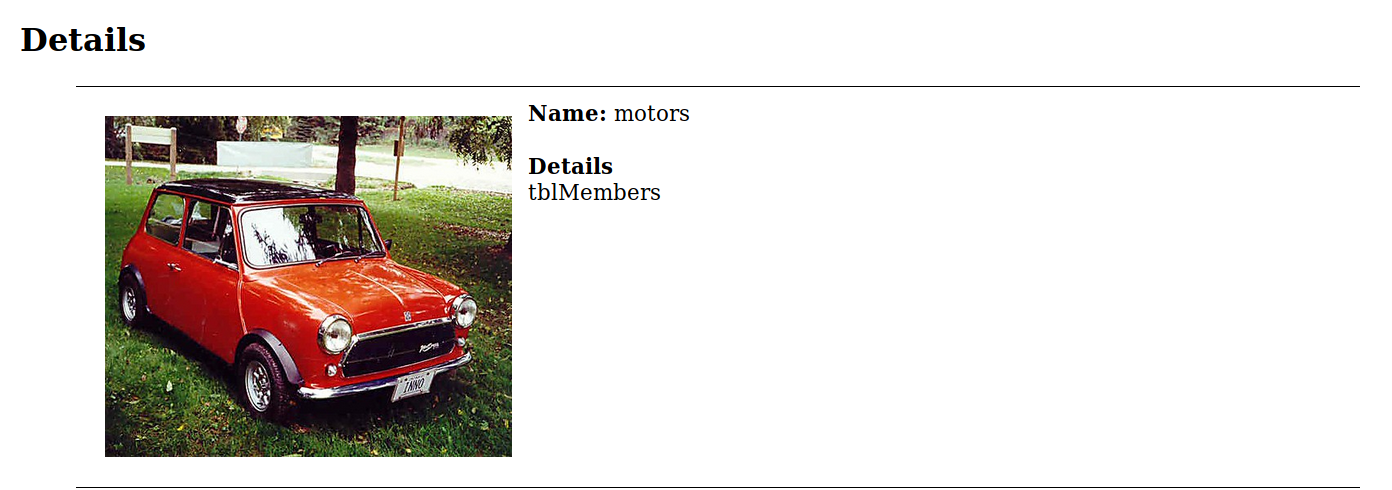


Após alguma exploração confirmámos a vulnerabilidade obtendo na página informações da base de dados como por exemplo :

Query:



Resultado: nome do schema que retorna a informação e do nome da primeira tabela desse schema.

****

Query:

****

Resultado (host:localhost e user:root da base de dados):



A partir da injeção de código neste url a maior parte das querys que retornam informação da base de dados são possíveis(leitura).

### 

### 

### 

### 

### Nikto Scan

Para além das ferramentas de scan já utilizadas, utilizámos também o nikto para correr uma análise ao website. Esta ferramenta para além de várias informações úteis acerca das tecnologias a serem utilizadas (PHP, Apache, SSL), detetou também que o site é vulnerável a XSS scripting, e dá-nos também informações acerca dos diretórios que compõe o website.

Uma informação importante é o facto de referir que existe um ficheiro config.php que contém informações acerca das credenciais da base de dados.

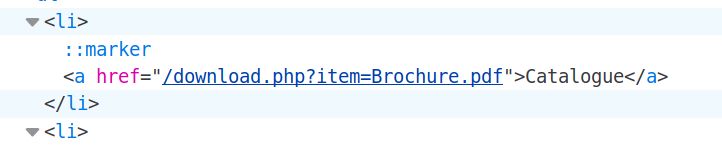
Executado: ***nikto -h 192.168.127.14***

Resultados:



### Obtenção de credenciais da base de dados

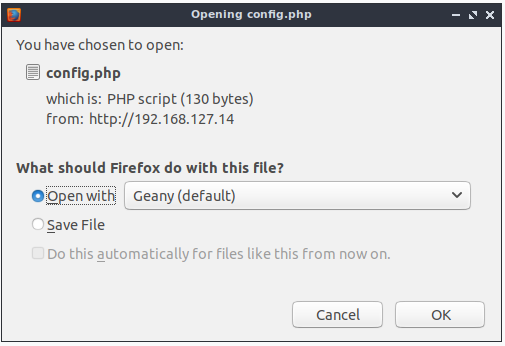
Ao clicarmos no catalogue na bottom nav bar do website reparamos que é feito o download de um ficheiro pdf através de um url. Ao inspecionarmos o elemento verificamos que é utilizado um php que recebe um item como argumento.



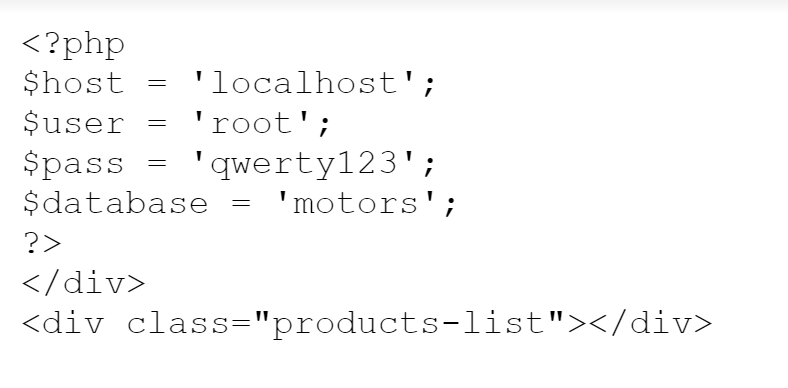
Através da técnica de **path traversal** é possível obter o ficheiro config.php apresentado pelo scan da ferramenta nikto como um ficheiro que contém credenciais da base de dados.

Ao efetuarmos o request para o url <http://192.168.127.9/download.php?item=../config.php>

obtemos o seguinte resultado:

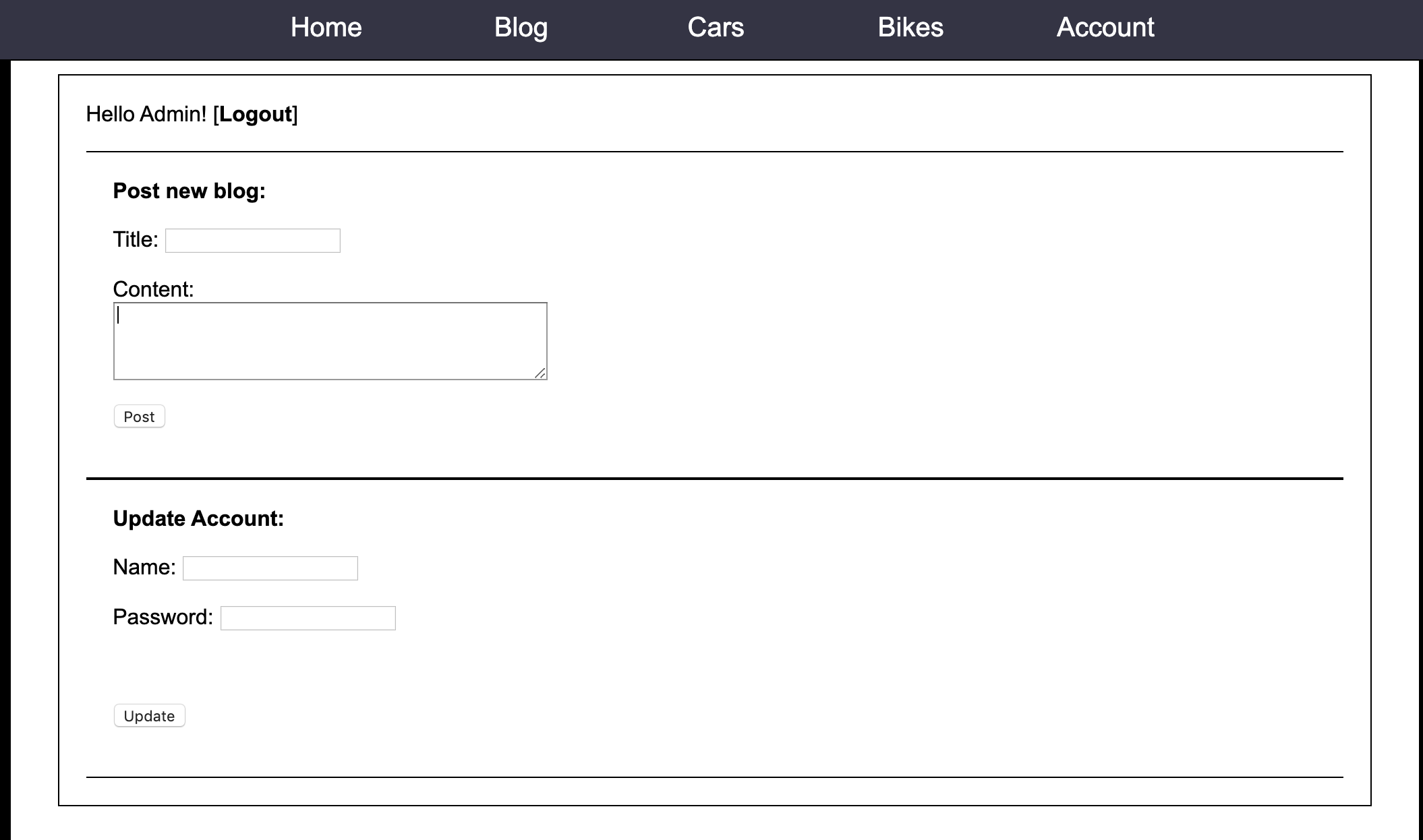


É dada a possibilidade de obter o ficheiro config.php. Este ficheiro contém as credenciais da base de dados.



### XSS

XSS - Cross-Site Scripting são ataques do tipo injeção. São inseridos scripts maliciosos em sites confiáveis e benignos. Os ataques XSS ocorrem normalmente de um browser e afetam usuários externos. Através deste tipo de ataques é possível o roubo de qualquer input, isto ocorre quando a aplicação utiliza inputs de um utilizador sem os validar ou codificar.

<http://192.168.1.122/account.php>

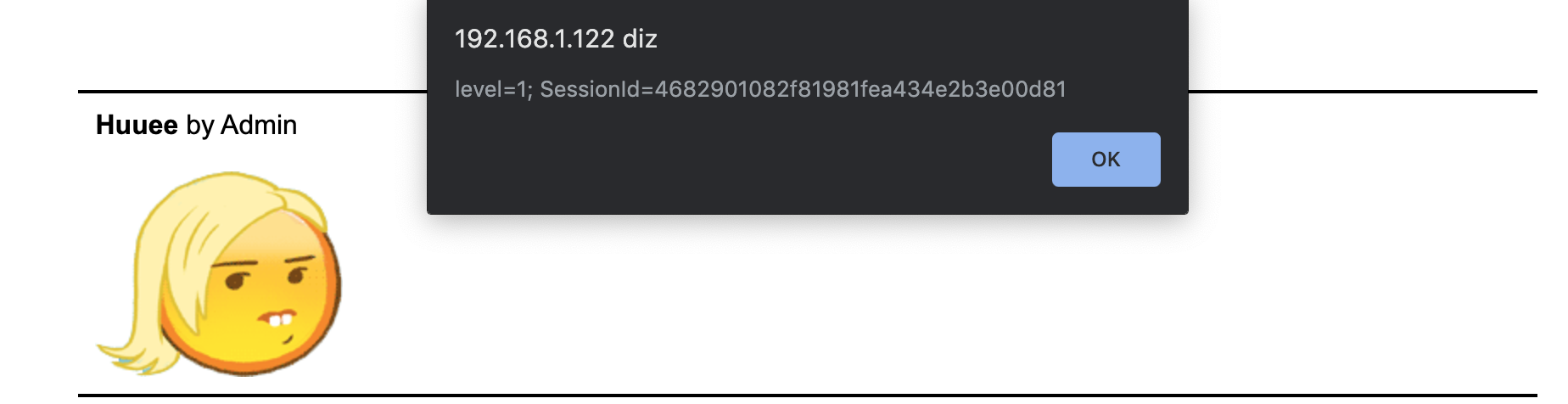
Se na página “Account” fizermos um “Post new blog” com más intenções é possível verificar um tipo de ataque XSS.(Tanto o campo “title” como o “content” são vulneráveis).

<img onmouseover=alert(document.cookie); src="https://i.imgur.com/4CwRA81.png">

Através do código anterior é possível colocar uma imagem, sempre que se passa o rato por cima da imagem é mostrado o cookie.Num cenário mais realista, podemos fazer o envio e receção destes dados adquiridos para um servidor.

O campo “Name” do update account é também vulnerável a este tipo de ataque.

Para que isto não aconteça todos os dados de input devem ter um limite e não permitir certos caracteres.

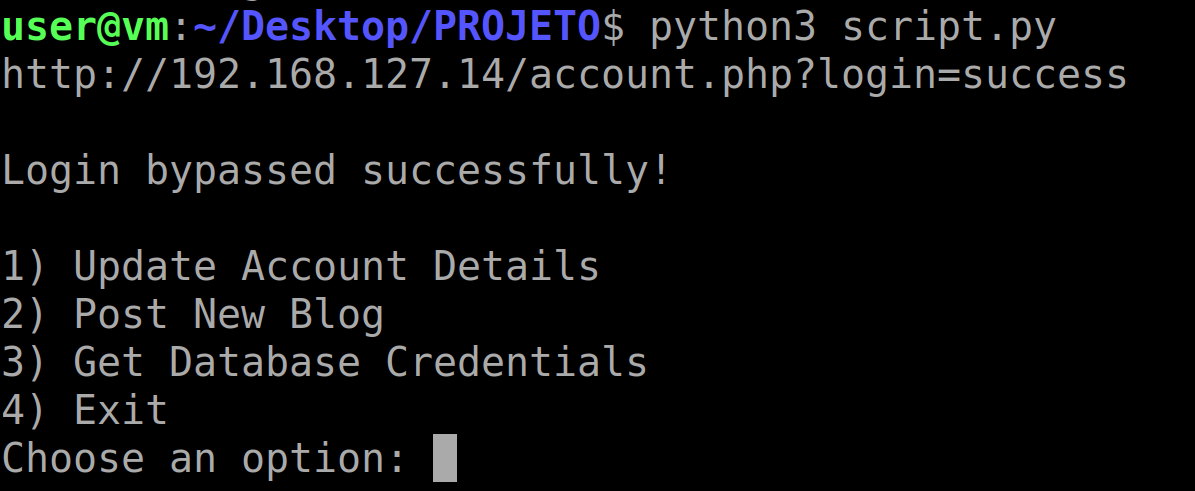


## Explore as vulnerabilidades encontradas, descrevendo cada passo tomado, a razão dela existir e qual o potencial impacto.

Explicado na alínea anterior.

## Crie uma ferramenta (script python) para automatizar os passos que levam à exploração da vulnerabilidade mais grave que encontrar.

Neste ponto decidimos criar uma ferramenta que consegue fazer o bypass do login do website e de seguida permite ao utilizador realizar 3 operações:



## A primeira opção permite fazer update dos dados da conta do administrador ( nome e password).

## A segunda opção permite fazer um novo post no blog em nome do administrador. Visto que o utilizador pode inserir o conteúdo que quiser, poderá inserir conteúdo que irá ser representado na forma de XSS cross-site scripting.

## A última opção permite ao utilizador obter o ficheiro config.php que contém as credenciais da base de dados utilizada no website.

## O script utiliza as bibliotecas mechanize e urllib para fazer pedidos http e obter informação do website.

## 

# Extra

Para finalizar o trabalho tentámos também (sem sucesso) aceder remotamente à máquina. Foi utilizada a ferramenta hydra com alguns utilizadores e passwords. Segue em anexo (ficheiros).

*hydra -s 22 -v -q -L ~/Desktop/users -P ~/Desktop/pass.txt -e nsr -t 9 -w 5 192.168.1.122 ssh*

# Bibliografia

<https://www.cvedetails.com/cve-help.php>

<https://en.wikipedia.org/wiki/Port_(computer_networking)>

<https://en.wikipedia.org/wiki/Secure_Shell>

<https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol>

<https://en.wikipedia.org/wiki/HTTPS>

<https://dev.mysql.com/doc/mysql-port-reference/en/mysql-ports-reference-tables.html>

<https://www.owasp.org/index.php/Path_Traversal>

<https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)>

<https://cwe.mitre.org/data/definitions/1004.html>