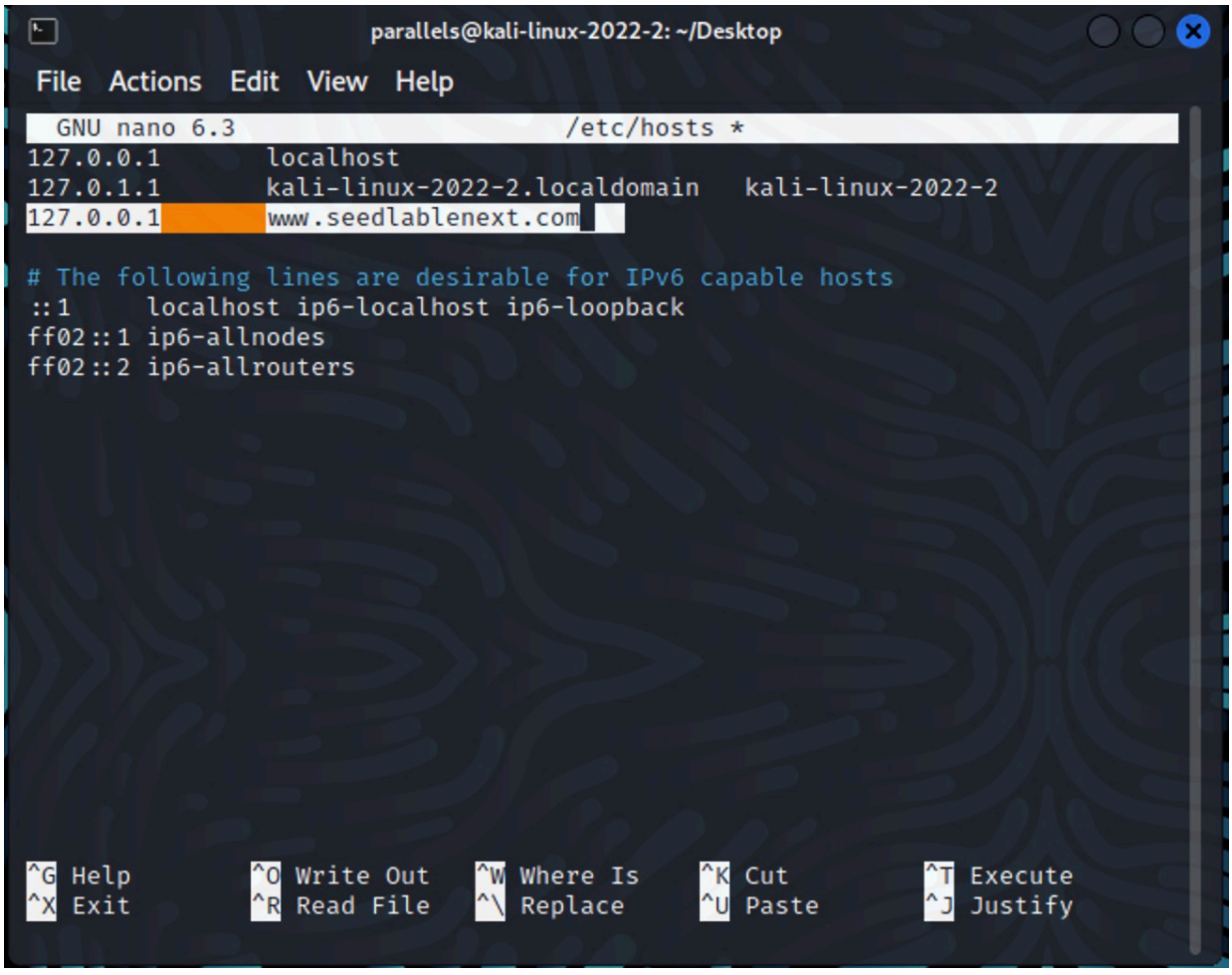
Applied Cryptography

Name	Naman Choudhary
SRN	PES2UG20CS209
Section	D

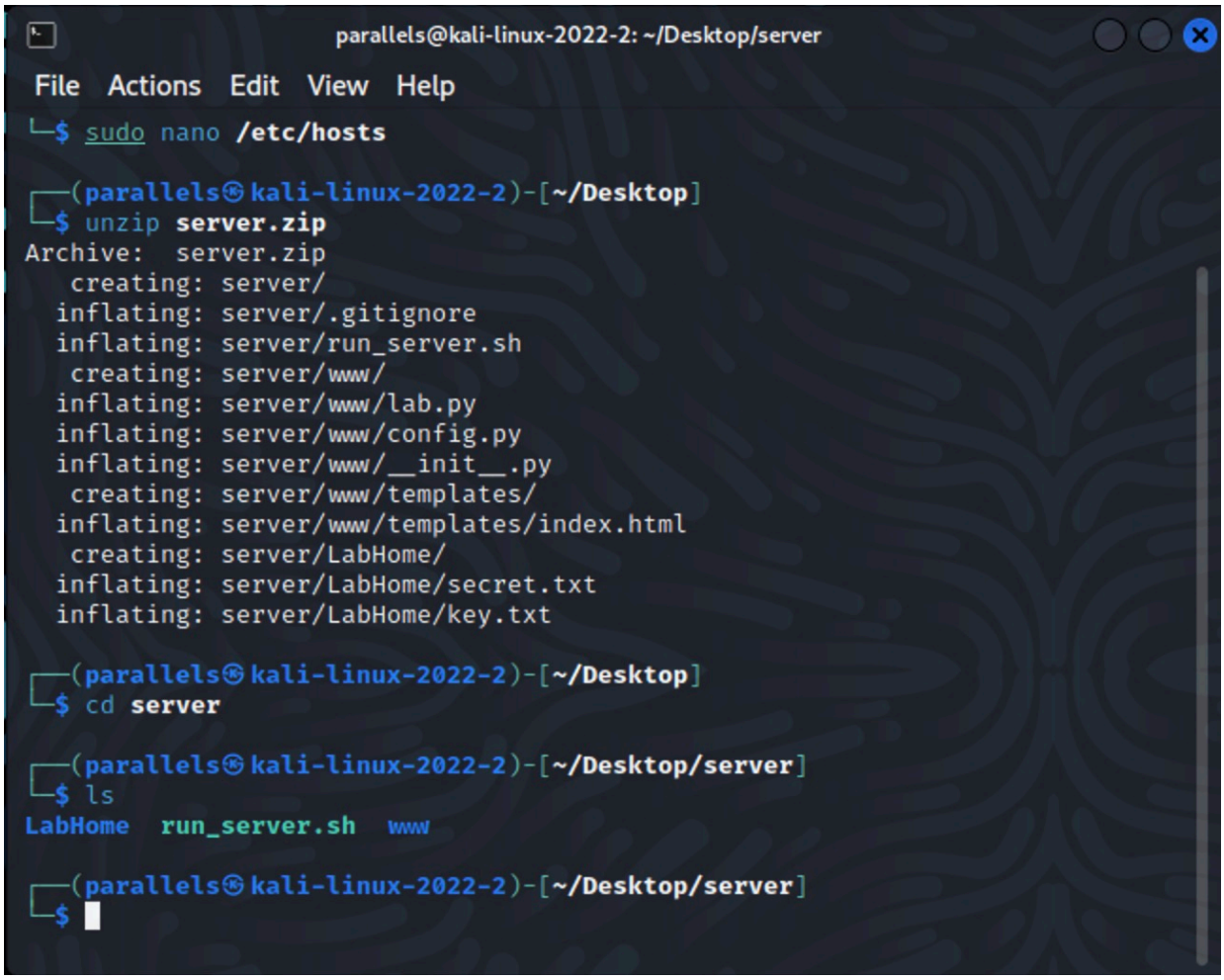
Hash Length Extension Attack Lab

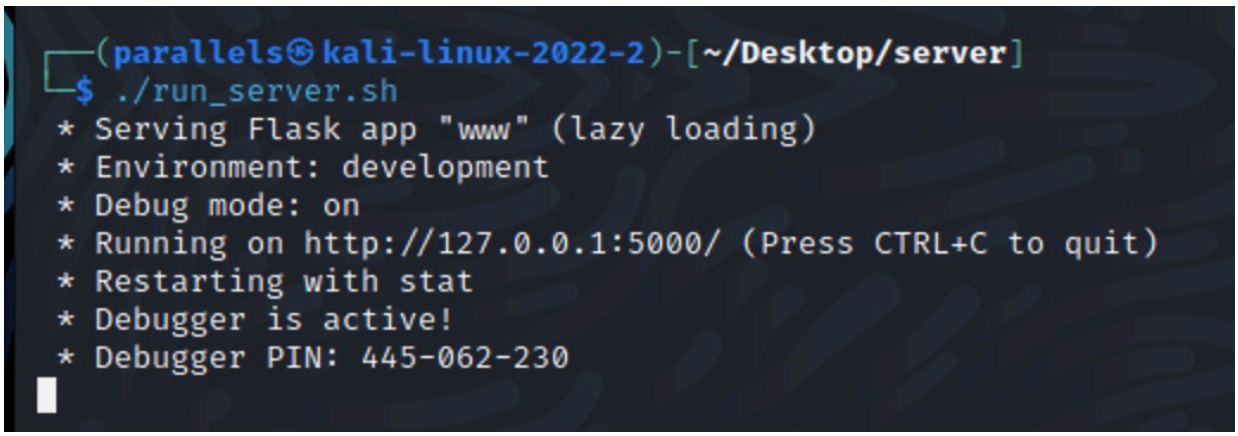
Lab Setup

Setting up the hostname



The server program



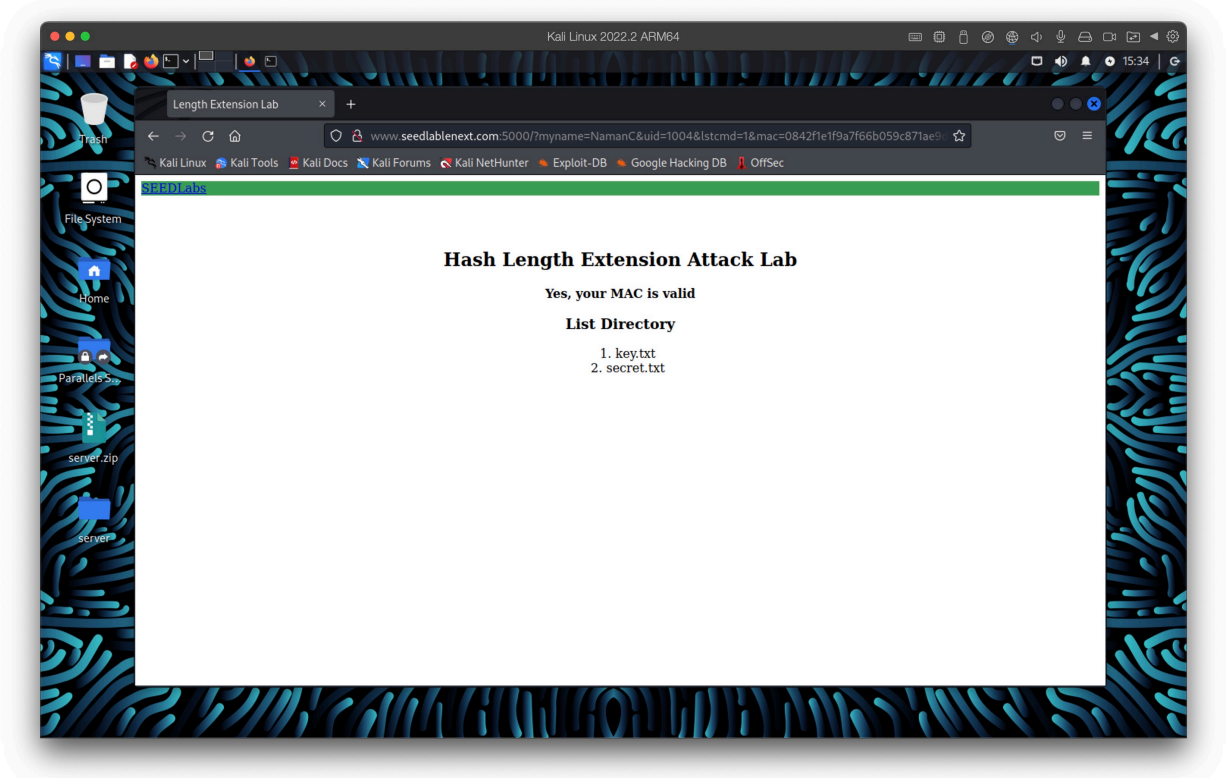


Task 1:Send Request to List Files

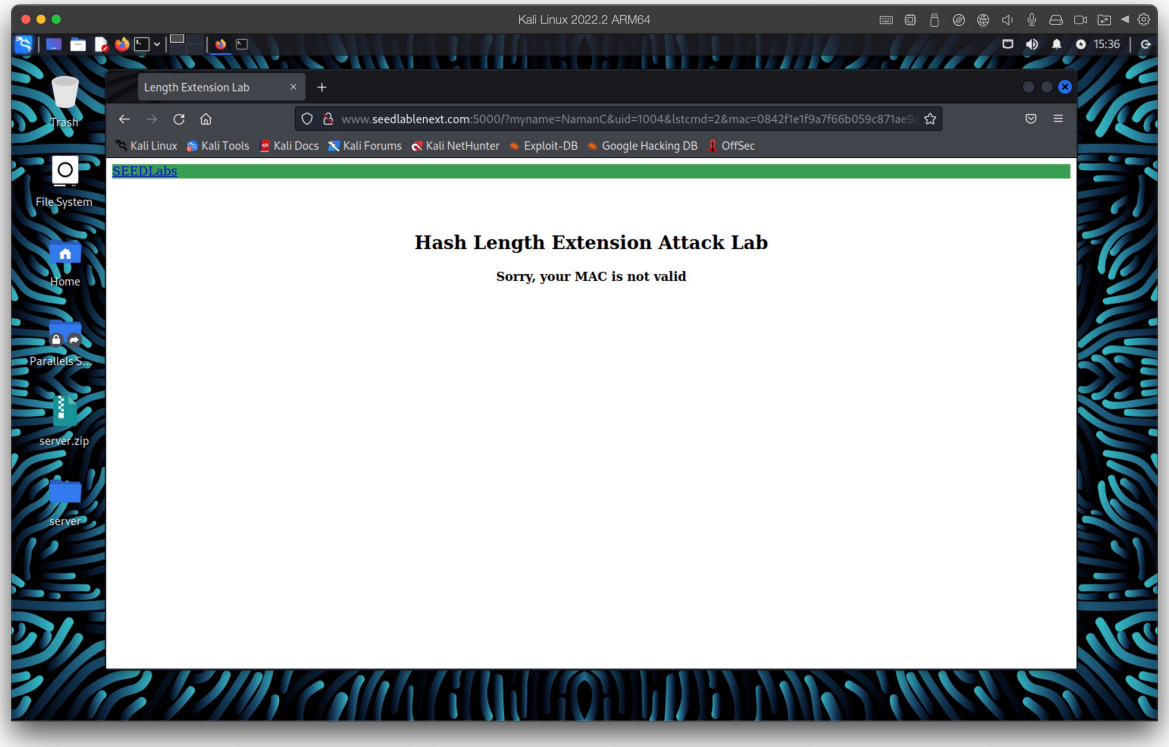
Calculating the MAC address

```
┌──(parallels㉿kali-linux-2022-2)-[~/Desktop/server]
└─$ cd LabHome

┌──(parallels㉿kali-linux-2022-2)-[~/Desktop/server/LabHome]
└─$ ls
key.txt  secret.txt

┌──(parallels㉿kali-linux-2022-2)-[~/Desktop/server/LabHome]
└─$ cat key.txt
1001:123456
1002:983abe
1004:98zjxc
1005:xciujk

┌──(parallels㉿kali-linux-2022-2)-[~/Desktop/server/LabHome]
└─$ echo -n "98zjxc:myname=NamanC&uid=1004&lstcmd=1" | sha256sum
0842f1e1f9a7f66b059c871ae9d7cd22311cf356173c3b4feab45e46dac5003f  -

┌──(parallels㉿kali-linux-2022-2)-[~/Desktop/server/LabHome]
└─$
```

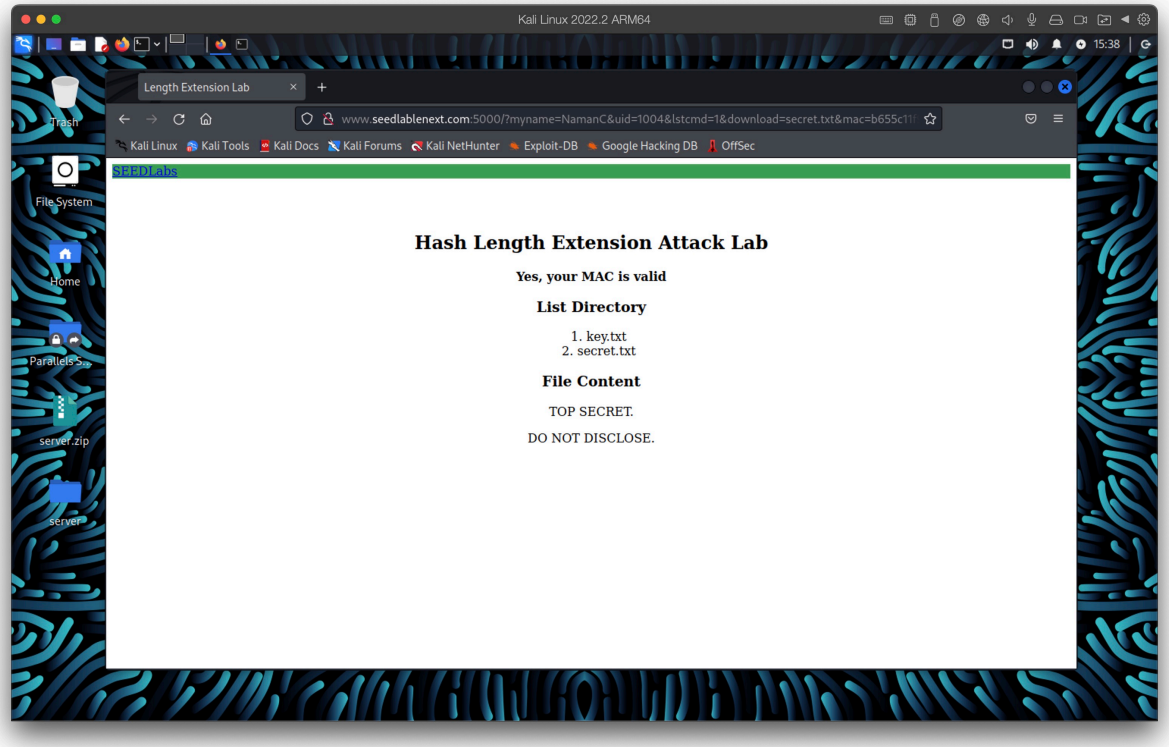
Result on the website



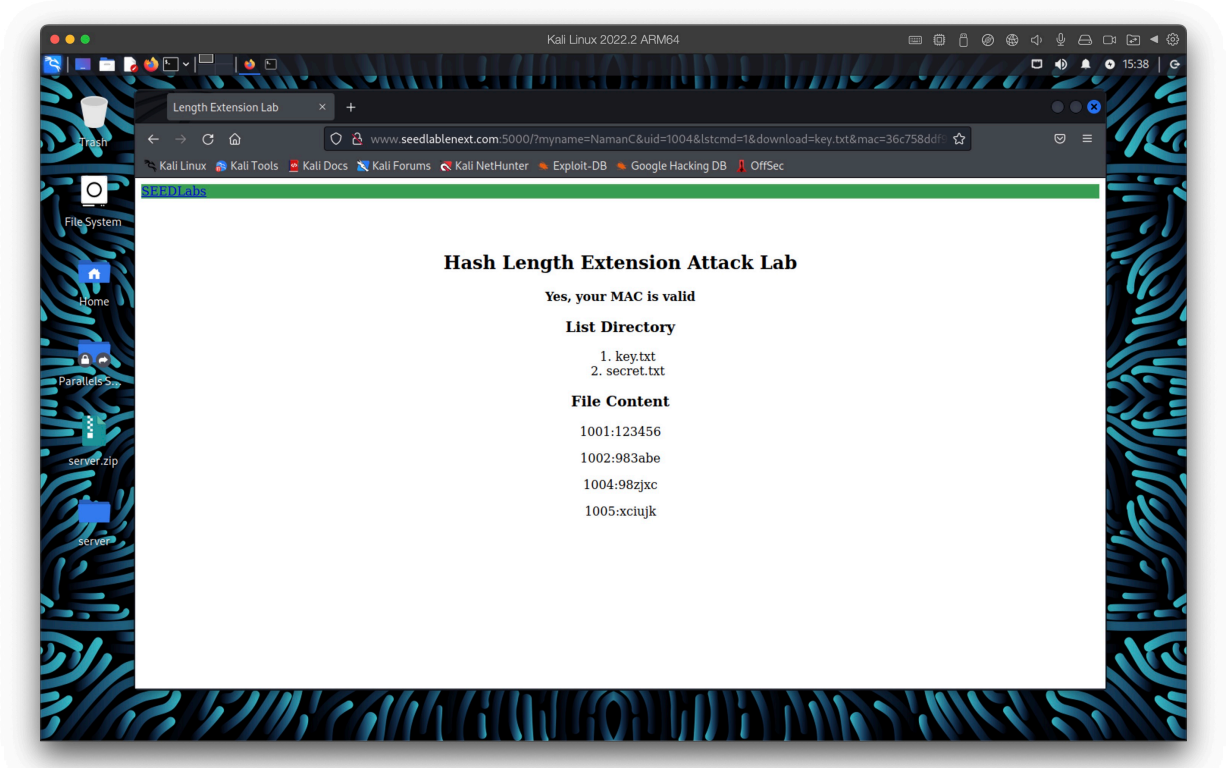
When an invalid MAC is given



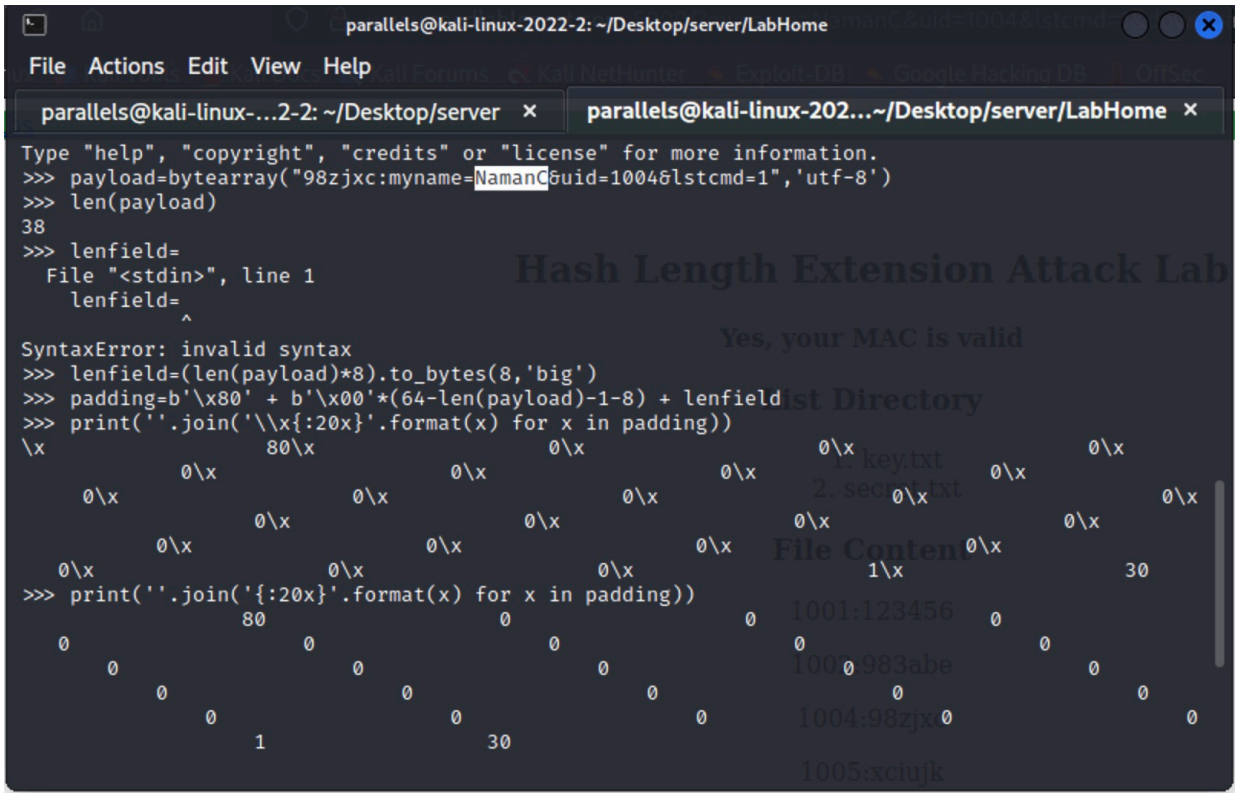
Contents of secret.txt



Contents of key.txt



Task 2:Create Padding



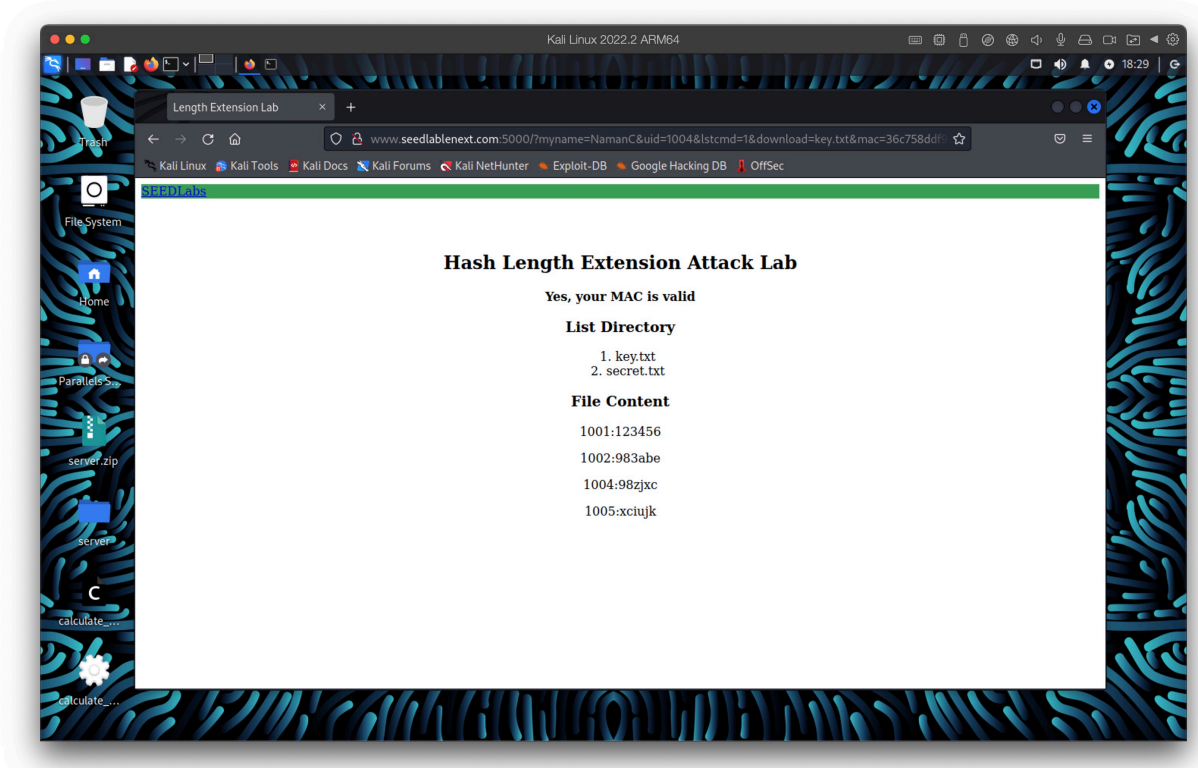
Task 3:Compute MAC using Secret Key

```c
/* calculate_mac.c */
#include <stdio.h>
#include <openssl/sha.h>
int main(int argc, const char *argv[])
{
   SHA256_CTX c;
   unsigned char buffer[SHA256_DIGEST_LENGTH];
   int i;
   SHA256_Init(&c);
SHA256_Update(&c,
   "This is a test message"
   "\x80"
   "\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
   "\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
   "\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
"\x00\x00\x00"
   "\x00\x00\x00\x00\x00\x00\x00\xB0"
   "Extra message",
   64+13);
SHA256_Final(buffer, &c);
for(i = 0; i < 32; i++) {
     printf("%02x", buffer[i]);
   }
   printf("\n");
   return 0;
}
```

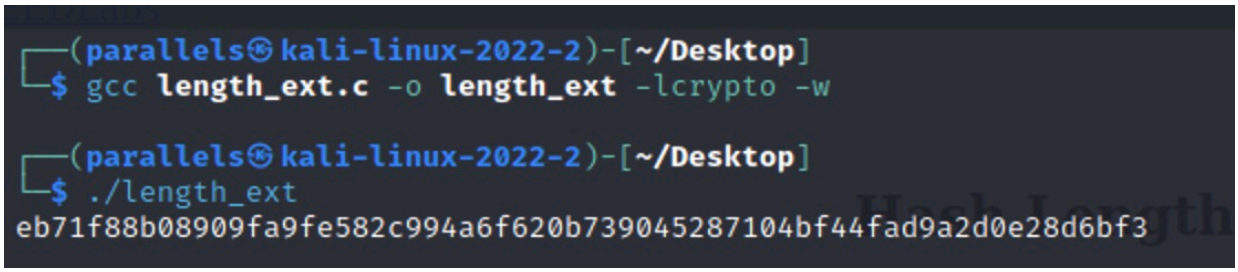
Using Python to create padding

```
┌──(parallels㉿kali-linux-2022-2)-[~/Desktop]
└─$ gcc calculate_mac.c -o calculate_mac -lcrypto -w

┌──(parallels㉿kali-linux-2022-2)-[~/Desktop]
└─$ ./calculate_mac
eb71f88b08909fa9fe582c994a6f620b739045287104bf44fad9a2d0e28d6bf3

┌──(parallels㉿kali-linux-2022-2)-[~/Desktop]
└─$
```

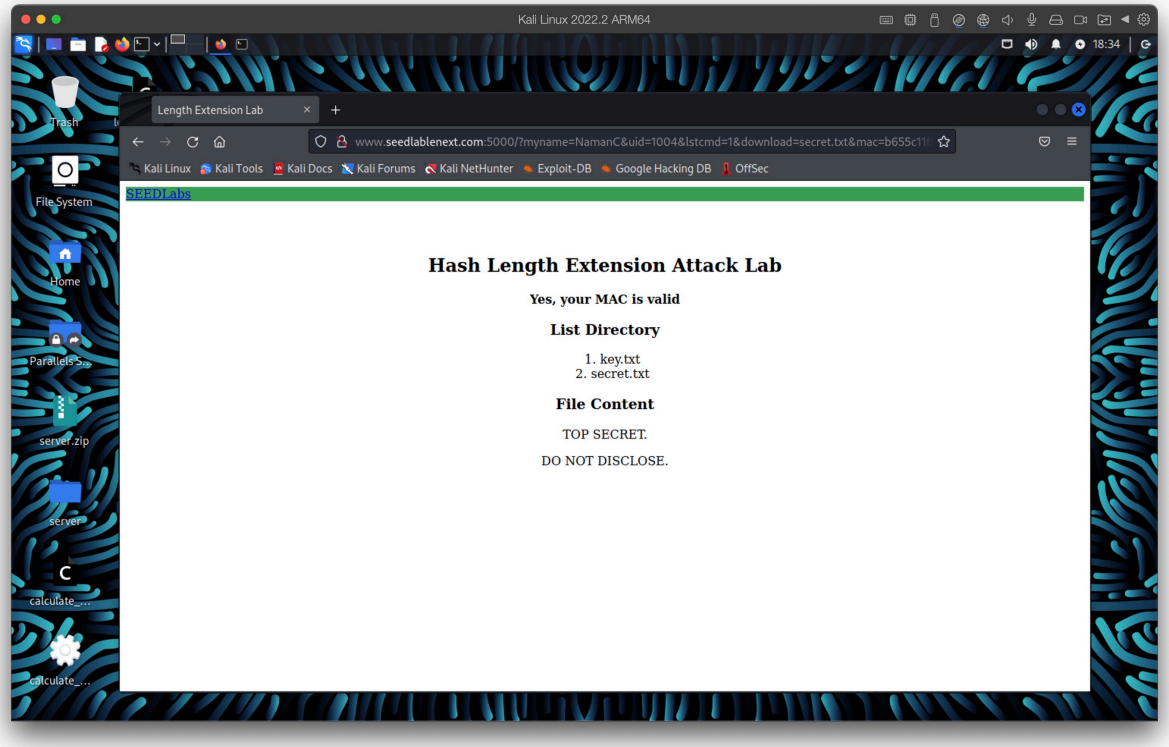
Website ouput



Task 4:Length Extension Attack



Website output



Task 5:Attack Mitigation using HMAC

Using Python to calculate HMAC

```
>>> import hmac
>>> import hashlib
>>> key='98zjxc'
>>> message='lstcmd=1'
>>> hmac.new(bytearray(key.encode('utf-8')),msg=message.encode('utf-8','surrogateescape'),digest
mod=hashlib.sha256).hexdigest()
'b2bfbb4f616a5bebc13642dae5606d02aa693cc091f251dd2f1e7d03bd9acd10'
>>>
```