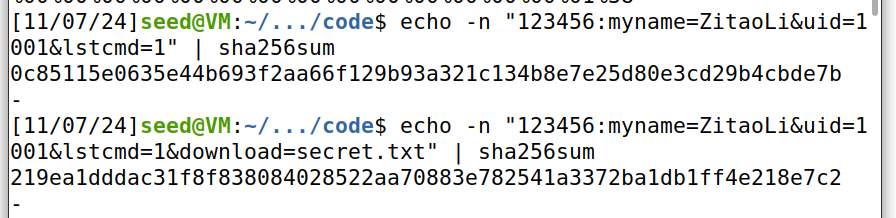
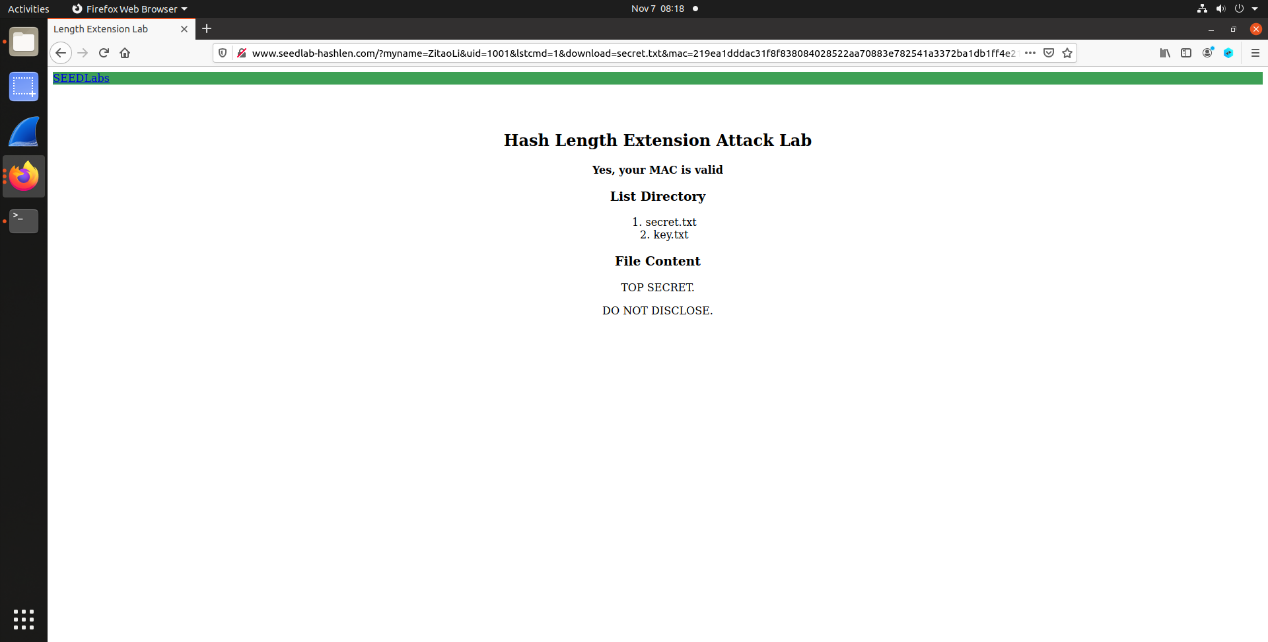
|  |
| --- |
| 哈尔滨工业大学（深圳） |
| **《密码学基础》实验报告** |
|  |
| Hash长度扩展攻击实验  学 院: 计算机科学与技术   |  |  | | --- | --- | | 姓 名: | 李子韬 | | 学 号: | 220110609 | | 专 业: | 计算机科学与技术 | | 日 期: | 2024-11-07 | |

1. 【任务1】请发送一个`download`命令到服务区，myname的信息修改为你自己的姓名拼音，并且记录你得到的响应内容（截图显示）。

生成mac



有download：

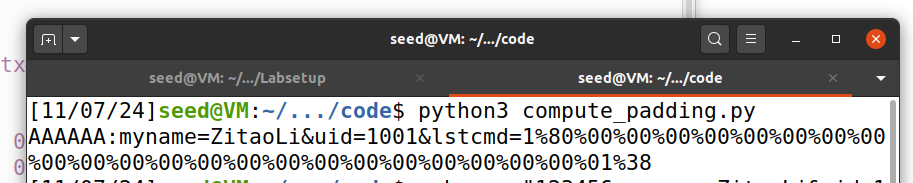


1. 【任务2】为消息<key>:myname=<name>&uid=<uid>&lstcmd=1创建对应padding，其中`<key>`和`<uid>`的实际内容应该从`LabHome/key.txt`文件中得到，myname依然用你自己的姓名。

*结果类似这样，红色部分可以参考代码换成AAAAAA，不影响填充的内容：*

123456:myname=SEEDManual&uid=1001&lstcmd=1

%80%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%01%50

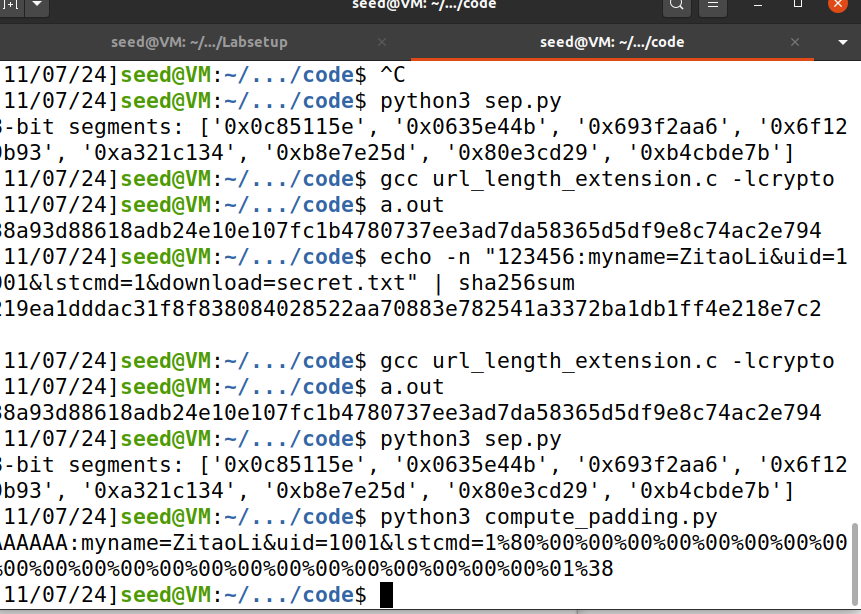


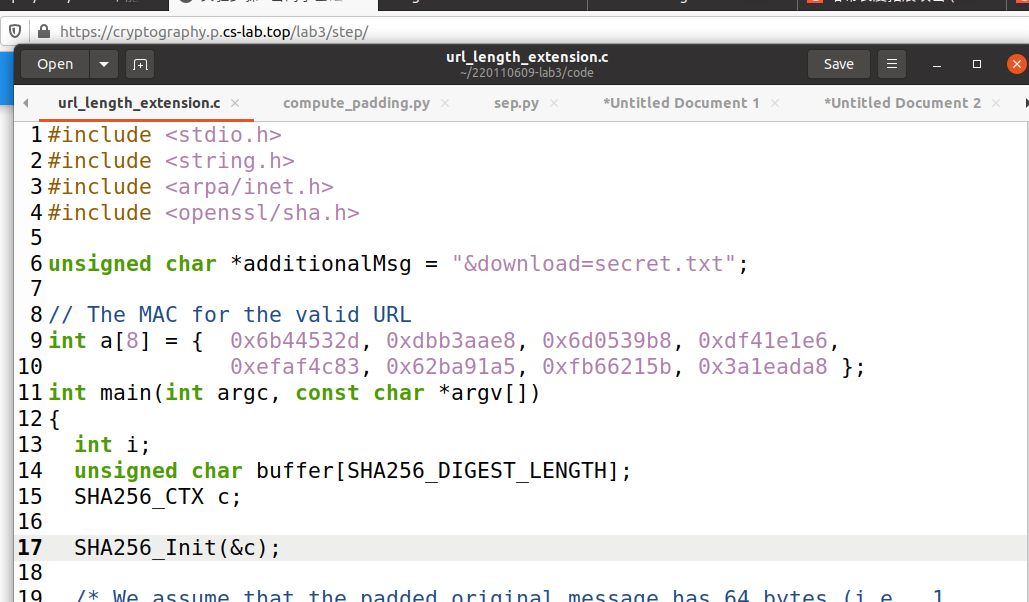
1. 【任务3.1】为下面的请求生成一个有效的MAC，其中`<key>`和`<uid>`的实际内容应该从`LabHome/key.txt`文件中得到，name就是自己的姓名拼音。

*http://www.seedlab-hashlen.com/?myname=<name>&uid=<uid>*

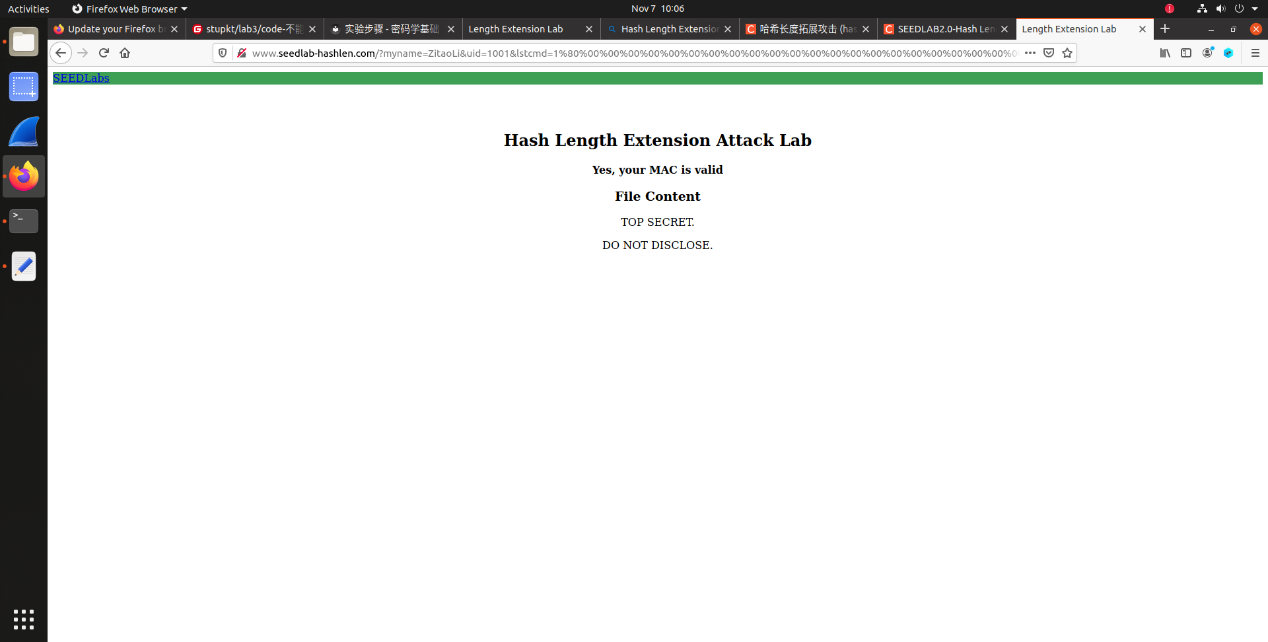
*&lstcmd=1&mac=<mac>*

利用1生成的第一个mac，分割填入newmac生成程序





带上padding：

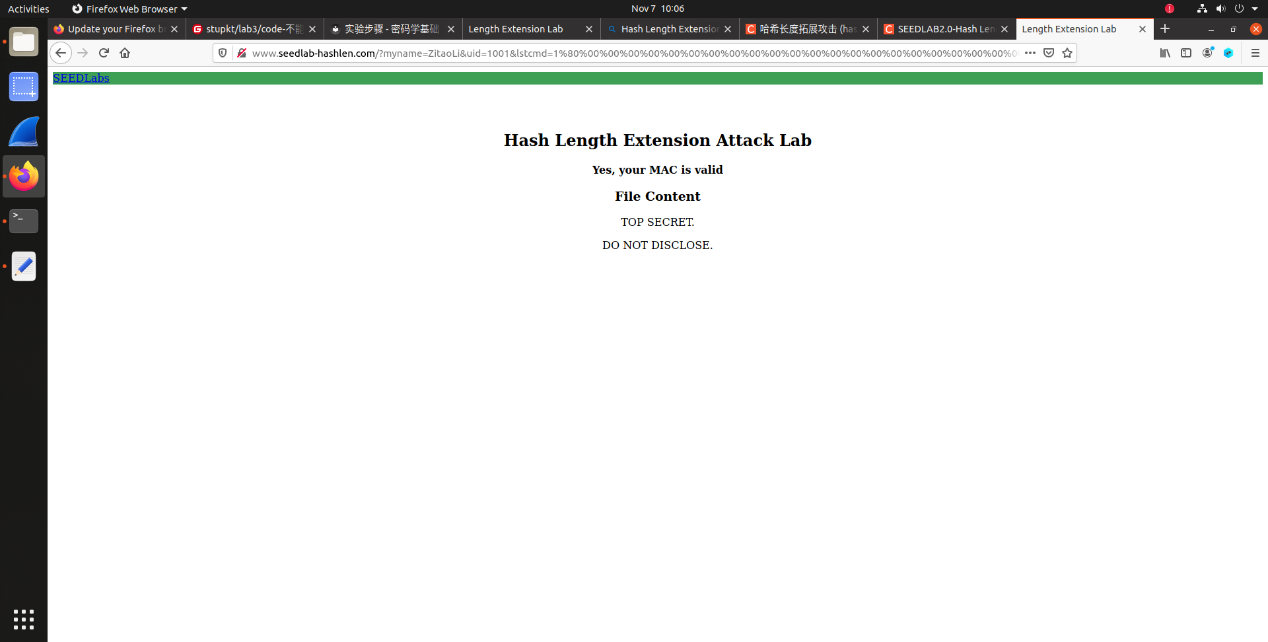


可以看见，Hash长度扩展攻击成功

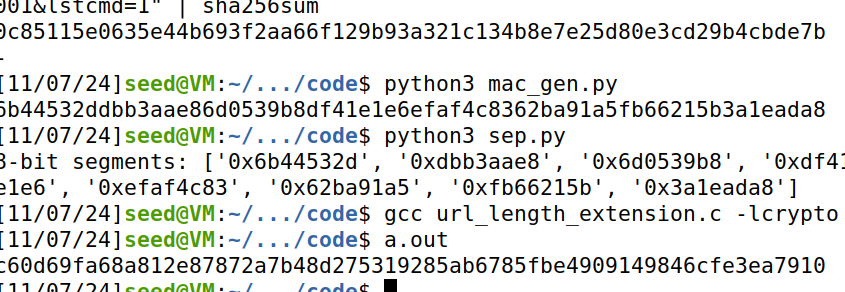
1. 【任务3.2】发送构造好的新请求到服务器，padding是上面获取到的信息，记录收到的服务器响应并截图。

http://www.seedlab-hashlen.com/?myname=<name>&uid=<uid>

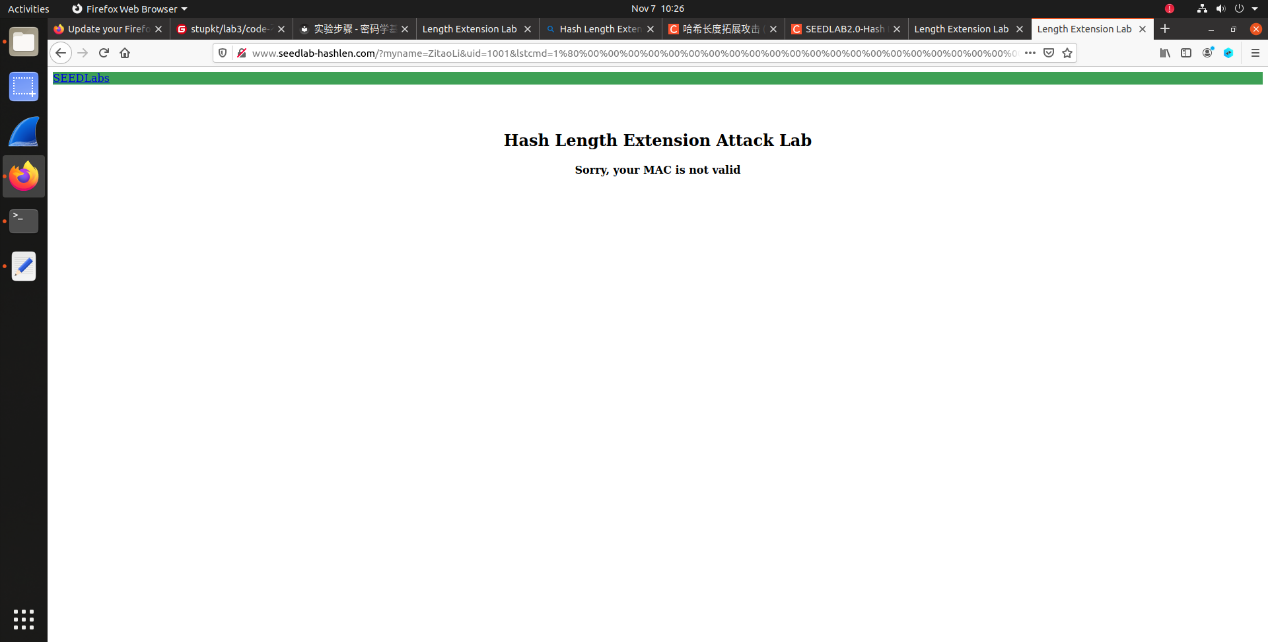
&lstcmd=1<padding>&download=secret.txt&mac=<new-mac>



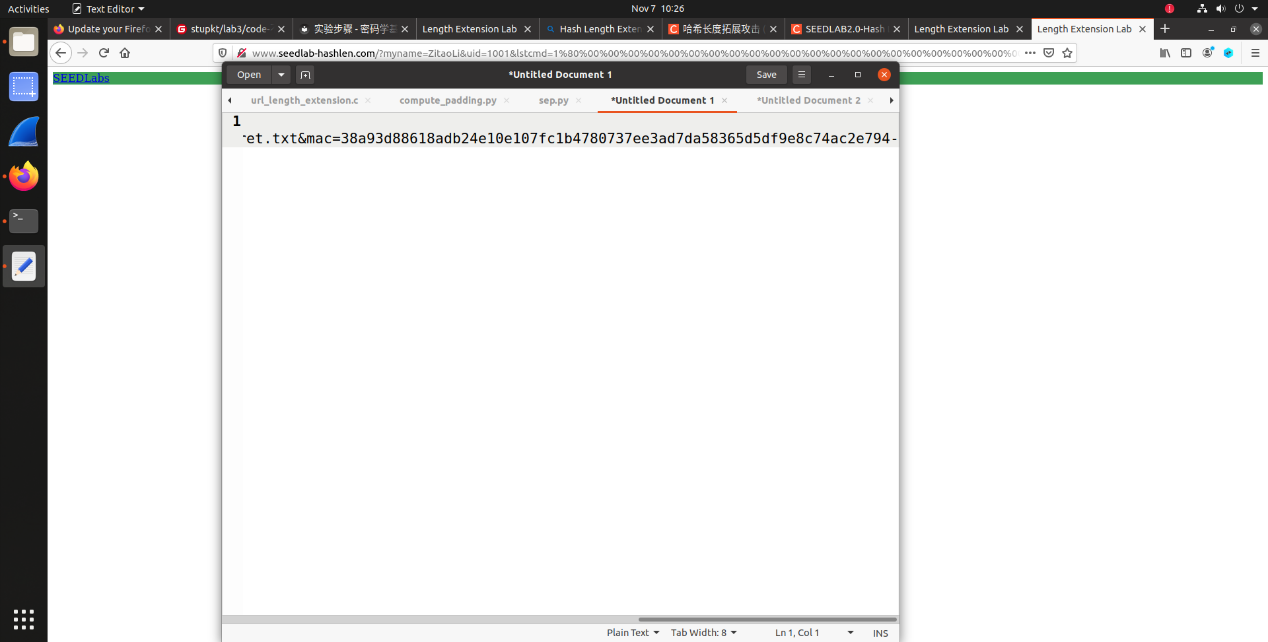
1. 【任务4】用HMAC算法修改代码后，记录使用长度攻击的结果，根据收到的服务器响应进行截图。



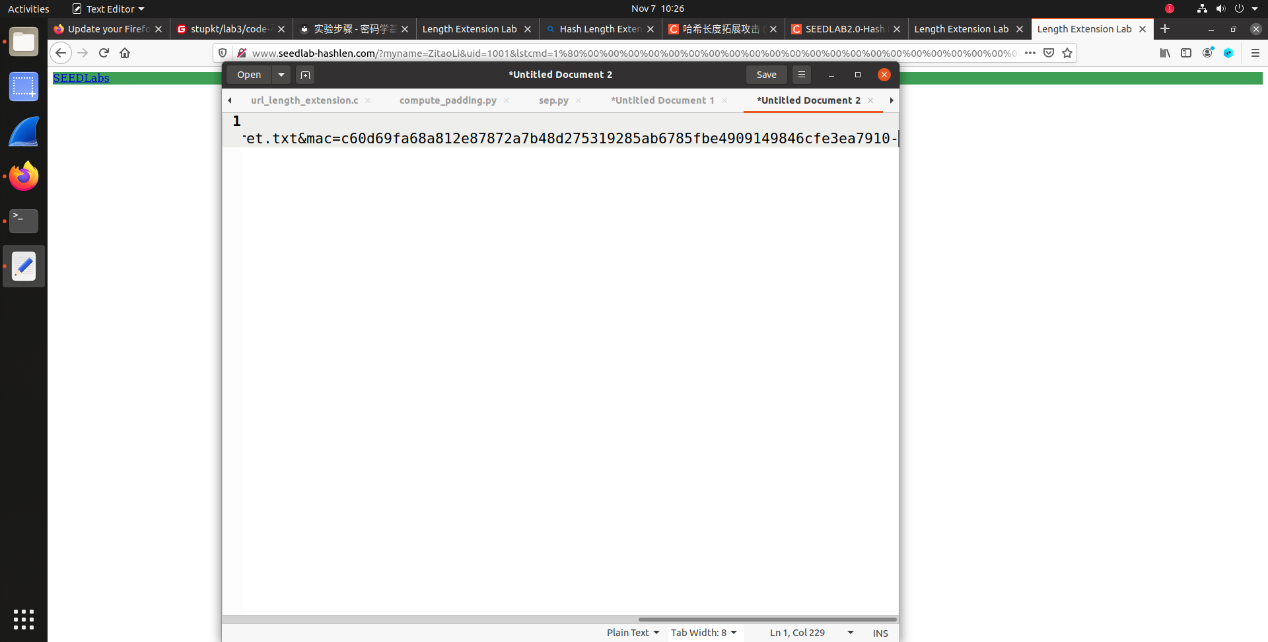
上图是改动为HMAC算法之后的密钥生成，再次使用长度扩展攻击



可以看见，攻击失败了，其中和第三个任务url只差一个mac，如下：



（上图是任务三，下图是任务四）



思考hash长度扩展攻击失败的原因：

哈希长度扩展攻击的原理是：如果我们知道了原始数据和其对应的哈希值（例如MD5、SHA1等），并且知道哈希算法的工作方式，我们就可以在原始数据后追加新数据并计算出一个新的有效哈希值（MAC）。这种攻击依赖于哈希算法的可扩展性，即哈希算法的内部结构允许攻击者在原有数据上扩展并生成有效的哈希值。

然而，HMAC通过引入密钥的机制解决了这个问题。具体来说，HMAC将密钥和消息数据一同进行哈希计算，并且使用了两次哈希操作（一次对密钥和数据进行组合哈希，另一次对结果进行哈希）。这种设计有效地防止了哈希长度扩展攻击。