Introduction

Scam the server is a challenge that is based on a length extension attack. The flag of this challenge is the port number for the next challenge.

The attack script is a python script that uses the hlextend module from
https://github.com/stephenbradshaw/hlextend
It is used to generate the length extended message and the new MAC for the message.

Idea behind the solution

First, the original message needs to be decoded.

VDU= can be decoded to T5

Since the server expects `T5flag` for it to return the flag, the message needs to be extended to `T5 || padding || flag` .

Afterwards, the MAC needs to be updated so that the server is able to verify the message.

To do so, we will feed the original_mac = fb92d54b2136c756ee80b2d2d8fd925ceccd09f4 and the other required data to the hlextend module and get the new mac.

Finally, the extended message, `T5 || padding || flag` , needs to be encoded to base64 before sending it, along with the new_mac, to the server.

How to use the attack script

The attack script is a python script that is used to generate the length extended message and the new MAC for the message.

1. To use the script, make sure the folder containing hlextend is in the same directory as the script.
2. Then simply run the script and the new_message + new_mac will be printed to the terminal.
   ```
   python3 scam_the_server_attack.py
   ```
3. Using the generated new_message and new_mac, send the new_message + new_mac to the server in this format `{"message": new_message, "MAC": new_mac}`

Alternate solution

HashPump can be used to generate the new_message and new_MAC.

To use HashPump, you will need to install it first.
Installation instructions can be found here:
https://github.com/bwall/HashPump

For MAC users, you can use homebrew to install HashPump by typing:

```
brew install hashpump
```

Then to use HashPump, simply type `hashpump` in the terminal and a CLI will appear requesting for the necessary information:

1. Input Signature
2. Input Data
3. Input Key Length
4. Input Data to Add

HashPump will generate the new_message and new_mac for you, but you will still need to encode the new_message in base64 before sending it to the server in this format:

```
{"message": new_message, "MAC": new_mac}
```