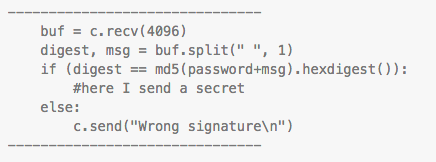
Hash-Length-Extension(Hle)

Problem: We have found one authorized message “c974b779d095f5772a36e2139276ffdc testing connection”.

Can you find another authorized message and send it to nc 192.168.3.5 41300.

Given:



Hint:

Intro:

This is a clear hash length extension problem. Luckily there are tools to help solve this so it does not have to be brute forced. To learn more about hash length extension visit this [link](https://en.wikipedia.org/wiki/Length_extension_attack). There are a few tools out there to help with this type of attack. The one that will be used for this write up is called hashpump and can be found [here](https://github.com/bwall/HashPump).

Steps:

1) Lets discuss what we know and don’t know from the given code and the authorized message we have.

First, we have a few important variables. digest and message which are what you want to manipulate and the password that is hidden. Since we don’t know the password we need to generate a digest with the part of the msg that we know.

We know that part of the message is ‘testing connection’. We also have the hash given from their password and the message we know. We now need to generate a new hash with the known message that will then match the digest.

2) Start making a script for the hash length extension. The following photo shows our known information and how to connect to the server to send and receive data.



Now we want to create a loop to use our test function to send our extended hashes that are generated by hashpump to the server and check if it is the correct answer. We want to create a for loop that calls the hashpump executable with certain options. This command will generate the hash and send it to the server. The following screenshot is how to set up the command line for hashpump.



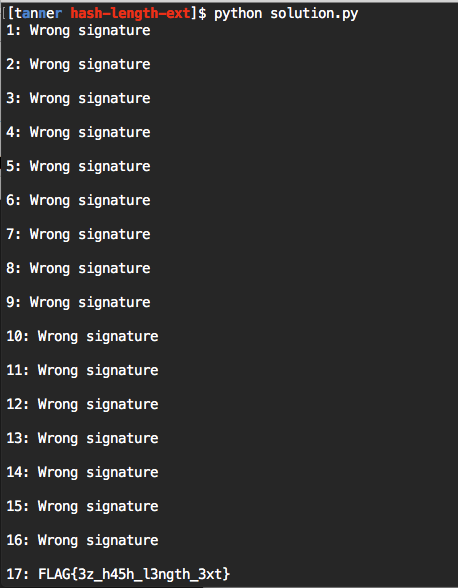
In this screenshot you can see how the full command will look like when calling hashpump:

./hashpump –s known\_sig –d \”+known\_text+\” –k str(i) –a “AAAA”

Then the next line uses the command call to call the hashpump file and get its output into a variable. Then split the hex string at ‘\\x80’(space) of the returned string from the hashpump command. Then strip out the ‘\\x’ hex beginning of each string and decode in hex. Now lastly, put the complete hash length extended string together and send it to the server.

Then we continue until we are not given wrong as the response from the server.

The solution then gives the following ouput:



Sample Script:

import socket

import commands

host = "192.168.3.5"

port = 41300

known\_sig = "c974b779d095f5772a36e2139276ffdc"

known\_text = "testing connection"

def test(sign, text):

sock = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM);

sock.connect((host, port))

data = sign + " " + text

sock.send(data)

return sock.recv(4096)

for i in range(1,32):

cmd = "/usr/local/Cellar/hashpump/1.2.0/bin/hashpump"

cmd += " -s " + known\_sig

cmd += " -d \"" + known\_text + "\""

cmd += " -k " + str(i)

cmd += " -a " + "AAAA"

sign, text = commands.getoutput(cmd).split('\n')

a, b = text.split('\\x80')

b = b[:-4].replace('\\x','').decode('hex')

data = a + '\x80' + b + 'AAAA'

res = test(sign, data)

print str(i) + ': ' + res

if not 'Wrong' in res:

break