14-741/18-631: Homework 1

All sections Due: Thursday September 23, 2021 by 10:00am EST

Name:

Andrew ID:

Total (100 pts max):

Guidelines (Please read before starting!)

- Be neat and concise in your explanations.
- You must use at most one page for your explanation for each Problem (code you wrote may be on additional pages). Start each problem on a new page. You will need to map the sections of your PDF to problems in Gradescope.
- To access the CTF problems, create the SSH proxy as per the guide on Canvas.
- For CTF problems, **you must use the following format in your explanation**:
  - CTF Username
  - Flag
  - Explain the vulnerability in the program, and explain conceptually how that vulnerability can be exploited to get the flag.
  - How did you exploit the vulnerability? List the steps taken and the reasoning behind each step. The TA grading should be able to replicate the exploit following the steps. Feel free to make references to your code! **Note that *"use XYZ online solver"* is not sufficient - you must explain how the online solver derived the answer for full credit.**
  - Append your source code in the same writeup. Your source code should be readable from the writeup PDF itself. Note that this does not count towards the page count above.

  Omitting any of the above sections would result in points being deducted.
- Some questions are marked as **Optional Team Work**. For those, you can work with another student. The maximum team size is 2. In your write up, please write that "completed as a team", followed by the andrew ids of you and your teammate. **Individual CTF username and flag need to be put in the write up for CTF questions.** Please refer to the problem write up for requirements of whether to submit individual or one joint write up (this may defer from problem to problem).
- **References:** List resources outside of class material that helped you solve a problem. This includes online video tutorials, other CTF problems on other platforms, etc. Remember that source code available online (e.g. stackoverflow) also needs to be cited. A quick guide on citing source code can be found here: `https://integrity.mit.edu/handbook/writing-code`. **Omitting the references section may result in an Academic Integrity Violation(s).**
- It is highly recommended that you use Python for your assignment. You may use other languages that you are familiar with, but the teaching team will not be able to support or debug language specific errors.
- Please check your English. You won't be penalized for using incorrect grammar, but you will get penalized if we can't understand what you are writing.

- Proofs (including mathematical proofs) get full credit. Statements without proof or argumentation get no credit.
- There is an old saying from one of my math teachers in college: "In math, anything partially right is totally wrong." While we are not as loathe to give partial credit, please check your derivations.
- Write a report using your favorite editor. Note that **only PDF submissions will be graded.**
- Submit to Gradescope a PDF file containing your explanations and your code files before 1:29pm Eastern Standard Time on the due date. You can find the timing for EST here: `https://time.is/EST`. Late submissions incur penalties as described on the syllabus (first you use up grace credits, then you lose points).
- If you choose to use a late day, you do not have to inform the instructors. We will calculate the number of late days used at the end of the semester based on the time of submission on Gradescope.
- Post any clarifications or questions regarding this homework to Piazza.
- **General allowed team work** Beyond designated team questions, you are encouraged to shared resources (e.g., TA's help, online resources you found helpful); you are encouraged to set up virtual study sessions with your teammate(s) to check each other's progress and discuss homework assignment solutions.
- **This is not a group assignment. Beyond your teammate, feel free to discuss the assignment in general terms with other people, but the answers must be your own.** Our academic integrity policy strictly follows the current INI Student Handbook `http://www.ini.cmu.edu/current_students/handbook/`, section IV-C.
- Good luck!

1 Breaking Vigenere (15 points)

The Vigenere cipher was considered unbreakable for centuries. Now it is easily cracked online! Log into the 14741/18631 CTF Server. Do not use the web shell. Copy/pasting is hard. Follow the directions provided on Canvas to SSH into the server.

To connect to a network service, you can use netcat (nc). For example, "nc 192.168.2.83 80" would connect to a service running on port 80. Note that we highly recommend you use Python - other languages are not officially supported in this course. Here's some simple Python code to get started:

```
#!/usr/bin/python           # This is client.py file

import socket               # Import socket module

s = socket.socket()         # Create a socket object
host = "192.168.2.83"       # Remote machine name
port = 12345                # Remote port

s.connect((host, port))
print s.recv(1024)
s.close() # Close the socket when done
```

You can also use *pwntools*, and the equivalent boilerplate code is provided below.

```
#!/usr/bin/env python
from pwn import *

host, port = "192.168.2.83", 12345

conn = remote(host, port)
content = conn.recv(1024)

print content

conn.close()
```

Solve the Vigenere problem. You can use an online solver for this problem, but please explain how the answer was derived. For an extra challenge, write the solver yourself! Submit a writeup containing your CTF username following the guidelines on the first page.

2 One-time pads (20 points)

A one-time pad is unbreakable crypto, but what happens if you reuse the pad? Solve the keyreuse problem on the 14741 CTF Server. The flag is the decrypted 32 decimal character message stored on the server. Submit a writeup containing your CTF username following the guidelines on the first page.

3 ECB (25 points)

Why is it bad to use ECB mode with block cryptography? Solve the ECB problem on the 14741/18631 CTF Server. Submit a writeup containing your CTF username following the guidelines on the first page.

4 Padding attacks (40 points) (Optional Team Work)

You can choose to form a team of 2 students or work individually. If you are working in a team, only one needs author the write up, and the other simply writes "*see [teammate andrewid]*". Each team member is expected to write their own code and individual CTF username and flag need to be put in the write up.

Preliminaries

PKCS#7 Padding AES is a block cipher based algorithm, which operates on 128 bit blocks (16 byte blocks) with a 128 bit key. AES requires padding to make plaintext in multiple of 16 bytes (padding is required even if the plaintext length is already multiple of 16 for avoiding ambiguity).

The PKCS#7 padding scheme is a simple padding algorithm: Calculate $b = 16 - (\mathsf{length}(plain) \bmod 16)$ then append b number of byte b to the plaintext. Below are examples of valid padding in hex representation:

"XXXXXXXXABC" requires 5 bytes padding: 414243**0505050505**
"XXXXXXXXABCDEF" requires 2 bytes padding: 41424344454**60202**
"XXXXXXXXABCDEFGH" requires 16 bytes padding: 4142434445464748**10101010101010101010101010101010**

Cipher Block Chaining (CBC) Mode As a block cipher algorithm, AES operates on blocks. The input string will be divided into multiple 16 byte blocks prior to encrypting or decrypting. Mathematical formulas for CBC are:

$$\begin{aligned} C_i &= \mathsf{Enc}(P_i \oplus C_{i-1}) \\ P_i &= \mathsf{Dec}(C_i) \oplus C_{i-1} \\ C_i &= i_{th} \text{ block of ciphertext } (C_0 \text{ is IV, first ciphertext block is } C_1) \\ P_i &= i_{th} \text{ block of plaintext (first plaintext block is } P_1) \end{aligned}$$

Decryption Attack In the decryption formula, attacker controls C_{i-1} and wants to learn $\mathsf{Dec}(C_i)$. So the mathematical formula is: $\mathsf{Dec}(C_i) = C_{i-1} \oplus P_i$. Attacker doesn't know P_i but the attacker knows whether P_i has valid or invalid padding using an Oracle.

Encryption Attack In this attack, the attacker controls both C_i and C_{i-1}. Once attacker learns $\mathsf{Dec}(C_i)$ using decryption attack, the attacker can make any text P_i with the formula: $P_i = \mathsf{Dec}(C_i) \oplus C_{i-1}$.

Solve the PKCS7 problem on the 14741 CTF Server. Submit a writeup containing your CTF username following the guidelines on the first page.

Local Setup

This problem will require you test out various techniques and figure out the way to decrypt and encrypt a piece of data without knowing the key. Often the correct technique isn't the one that comes to mind first. Try testing your technique locally before testing it on the server endpoint.

- create empty directory
- download the source file from the PICO website
- run the commands below to setup a local netcat deployment (On Ubuntu)

```
echo "testFlag{E1I00T}" > flag
echo "4e591d42ef3533b42dd0fc3bebefac32" > key
echo '{"username": "guest", "expires": "2000-01-07", "is_admin": "false"}'
> cookie
```

```
socat tcp-listen:23555,reuseaddr,fork EXEC:'python pcks7.py' &
```

This last command opens a local problem instance at localhost:23555 (nc localhost 23555), which you can connect with in the same way as the CTF server. It will be a good idea to try out the problem with the values above. Once your solution works locally, try running it against the PICO server's endpoint. Alternatively you may use pwntool's process function like:

```
oracle = process('./pcks7.py')
```

You can find more information on this function at: `https://docs.pwntools.com/en/stable/tubes/processes.html`