CNT5412 – Network Security **Spring 2025**

Lab 4: Deadline Friday 3/7

Instructor: Viet Tung Hoang

1. (**Padding-oracle attacks**) (125 points) In this lab, you are asked to implement the padding-oracle attack. The target server is `linprog2.cs.fsu.edu` (IP address is 128.186.120.158), and the port number is 31537. Due to some administrative restriction, to have a socket connection to this server, you need to run your code at the same machine. (You can log into this server using your linprog account.)

   The server maintains a secret string M which your program needs to recover. Currently, this string is (without the quote and case sensitive) "*C0ngr4tul4ti0n5!!! Y0u 5ur3 kn0w ur crypt0!!! :)*", and thus the byte length is 48. However, when I test your programs, I may change it to a different string, of a different length. You should *not* assume that the byte length of the secret is a multiple of 16.

   Interacting with the server. The server provides two types of requests. If you want an encryption of $P\|M$, for some chosen prefix P, then you need to send a request of the form "-e P". The server will send you a ciphertext C of $P\|M$ under the TLS 1.2 encryption on a random IV L. (This encryption scheme is theoretically secure, but if the implementation leaks the reason for decryption failure then you can mount the padding-oracle attack on it.) If you want to send a ciphertext core C with IV L for validation, you need to send a request of the form "-v C L", and the answer is "Valid", "Invalid: Wrong Pad", or "Invalid: Wrong Tag". Your goal is to recover the message M, and output it to `stdout`.

   For an example of how to interact with the server, see the Python script client.py in the course website. When I run `python3 client.py` in my terminal, the script waits for me to enter requests to the server. If I type `-e abcdef0123456789` (meaning my prefix is the **hexadecimal** string `abcdef0123456789`), I get back

   ```
   b‘Encryption:  80\n49 8d 8e 57 ...6a 3d de \n’
   IV: b‘a5244e79b9f94b4f5634a8b00e06e46c’
   -E abcdef0123456789
   ```

   Here the cipherext core C (in hex encoding) is `49 8d 8e 57 ...6a 3d de` that is 80-byte long, and the IV L (in hex encoding) is `a5244e79b9f94b4f5634a8b00e06e46c` If you want to encrypt with the prefix as the empty string, just use "`-E`". Likewise, if I type `-v aaaaaa 40beed9c1b5bfcbc997bc025a42b4c0a`, I get back `‘Invalid: Wrong Pad’`. This means that after the server CBC-decrypts the ciphertext core `aaaaaa` with IV `40beed9c1b5bfcbc997bc025a42b4c0a`, it finds the padding incorrect.

   Another testing server. To help you to test your program on fragmentary secrets, I also set up another server at the same IP address, but the port is now 31536, and the secret is "*Hello World*".

   Deliverables. Upload to Canvas a zip file containing your source code, which includes a README.txt that informs me how I should run the program.