CS-6343-Cryptography

Summer 2022

# LAB #1: Modes of Operation

This lab is meant to give you hands-on experience with some of the concepts of symmetric ciphers. You will use OpenSSL, a program and library that supports a wide range of cryptographic operations. Most OSes these days come pre-installed with OpenSSL, so you are free to use whichever environment you prefer. My personal choice for this kind of work has tended to be Kali linux since Kali comes with lots of other cool security things to play about with (being a windows user, this means a virtualbox Kali VM on my windows system).

If my own choice resonates with you, you may find a prebuilt Kali VM here: https://www.osboxes.org/kali-linux/#kali-linux-2017-03-vbox

… and of course virtualbox itself here:

https://www.virtualbox.org/wiki/Downloads

You are free to do the project in an environment of your choice. You don't have to follow my choices.

1. You will first create random keys and initialization vectors that you will use throughout the lab.

   For example, to create an 8 byte random number (hex), you can use the command:

   ```
   openssl rand —hex 8
   ```

   Create a random 8-byte key and 8-byte IV that you will use for DES operations and a random 16-byte key and 16-byte IV that you will use for AES operations.

2. To encrypt the file MyFile and produce the output in the file ciphertext.bin, you use the command:

   ```
   openssl enc nameofcipher -e -in MyFile -out cipher.bin -K key -iv intializationvector
   ```

   The –e stands for encryption. To decrypt you would use –d and specify the cipher text as input. An example of the nameofcipher is: -aes-128-cbc for 128-bit aes being used in cbc mode or –des-ecb for DES being used in cbc mode.

   Will you need the iv for all schemes? Go review your course slides.

   In this exercise you will compare the behavior of the CBC and ECB modes of operation. You have been provided with an image file, Secret.bmp. The file is for a message that

Marty Byrde has left behind for Wendy Davis, as they launder money for the Navarro cartel. The message must not be seen by anyone besides these two, so Marty has decided to encrypt it before saving it on the local drive.

(a) Before doing anything with the image file, view it using an image viewer of your choice.

(b) Encrypt the image file using: (i) –des-ecb, (ii) –aes-128-ecb. You now have two different encrypted files (Lets say you name them Secretdesecb.bmp and Secretaes128ecb.bmp). Try to view these two encrypted files using the image editor you used previously.

What do you see and why?

(c) To correct the issues you should have met above, you will use the help of a hex editor. Install a hex editor of your choice and use it to open the original unencrypted image file. Now that you can directly view the bytes in the file, copy the first 54 bytes of Secret.bmp and paste them to replace the first 54 bytes of Secretdesecb.bmp and the first 54 bytes of Secretaes128ecb.bmp.

Use the image viewer to view each of the two new files you have created.

(i) What explains the difference in behavior to what you observed in (b)?

(ii) Are you impressed by the encryption? Explain why (why not).

(d) Repeat steps (b) through (c) using the schemes: (i) –des-cbc, (ii) –aes-128-cbc

Explain the difference in observations between the case when you used the ecb options and when you used the cbc options.

(e) The problems seen above could also happen with data that is not an image (e.g., text data). Your task here is to show experimental evidence to support this claim. You will come up with (a carefully crafted?) file that is not an image and do an encryption and show and discuss your results with regard to the vulnerability seen above with ecb.

3. In this exercise, you will compare the effects of data corruption on ecb, cbc, ofb and cfb. While encryptions schemes in practice will have mechanisms to address corruption, this is still a critical exercise for our understanding of the dynamics of the above modes of operation. You will create a small text file that is several blocks long, encrypt the file using DES or an AES configuration of your choice, flip a single bit in the cipher text and then decrypt the corrupted cipher text to observe the impact of the corruption on each

of the above 4 modes of operation. Discuss your observations and how they relate to the operation mechanisms of the different cipher options.

NOTE: The project report will comprise descriptions of your observations at each stage, supported by screen shots showing what you observed.

NOTE:

Before you begin the experiment, please do the following.

If you have a recent Kali installation, your default shell should likely be *zsh*. Add the following text at the top of the *~/.zshrc* file.

cowsay "Welcome *First Name Last Name*! Today is $(date '+%A %B %d %Y %r')"

Install *cowsay* before restarting the terminal and make sure the name and date displayed show up in all the screenshots you will submit in blackboard. *First Name* should be your first name and *Last Name* should be your last name. If your shell is bash, the edit should be made to *~/.bashrc* instead.