CPE-321: Introduction of Computer Security
Assignment: Symmetric Key Cryptography Implementation

Objectives

The objectives for this lab assignment are as follows:

- To explore symmetric key cryptography security with different modes
    - Electronic Codebook Mode (ECB)
    - Cipher Block Chaining Mode (CBC)
- Explore the limits of block ciphers in their use
- Performance Study of Public/Symmetric key Algorithms

Background

AES-128 provided by the cryptographic library will be utilized to implement different modes of operations. AES is a block cipher, and expects a 128-bit plaintext message and produces a 128-bit ciphertext – nothing more, nothing less. Putting a block cipher, like AES, into a "mode of operation" is the way to enable for encrypting data larger than 128-bits using a single key. However, a problem arises in certain modes of operation if the files are not evenly divisible by 128-bit. To account for plaintexts that are not an integral size of AES's block size, implement PKCS#7 padding. Details of this scheme can be found here: http://tools.ietf.org/html/rfc5652#section-6.3 but perhaps a more easily understood description is here: http://en.wikipedia.org/wiki/Padding_(cryptography)#PKCS7.

Tasks:

Task 1: Modes of Operation. In this task, you will explore the differences in security attained by the ECB and CBC modes of encryption. Using the AES-128

primitive provided by your cryptographic library, implement the ECB and CBC modes of operations (don't cheat and use built-in methods.) Your program should take a (plaintext) file, generate a random key (and random IV, in the case of CBC), and write the encryption of plaintext in a new file. ~~Encrypt one of the BMP files from the OTP assignment using your ECB and CBC implementations, creating two different ciphertexts. (Be sure to the preserve and re-append the plaintext BMP headers.)~~

Task 2: Limits of confidentiality. In this task, you will explore some of the limits of block ciphers in their use within a secure system. Start by using the PKCS#7 padding and CBC code from Task 1 to write two "oracle" functions that emulate a web server that wants to use cryptography to protect access to a site administration page. First, at the start of your program generate a random AES key and IV, which will be used in both of functions, keeping it constant for the execution of your program (don't generate a new key or IV for every encryption and decryption). The first function, called submit(), should take an arbitrary string provided by the user, and prepend the string:

userid=456; userdata=

and append the string:

;session-id=31337

For example, if the user provides the string:

You're the man now, dog

submit() would create the string:

userid=456;userdata=You're the man now, dog;session-id=31337

In addition, submit() should: (1) URL encode any ';' and '=' characters that appear in the user provided string; (2) pad the final string (using PKCS#7), and (3) encrypt the padded string using AES-128-CBC. Submit() should return the

resulting ciphertext. The second function, called verify(), should: (1) decrypt the string; (2) parse the string for the pattern ";admin=true;" and, (3) return true or false based on whether that string exists. If you've written submit() correctly, it should be impossible for a user to provide input to submit() that will result in verify() returning true. Now the fun part: use your knowledge of the way CBC mode works to modify the ciphertext returned by submit() to get verify() to return true. Hint: Flipping one bit in ciphertext block ci will result in a scrambled plaintext block mi, but will flip the same bit in plaintext block mi+1.

Task 3: Performance Comparison. In this task, we will quantify the performance differences between public and symmetric key algorithms. Fortunately, OpenSSL provides a simple interface for doing so.

Openssl speed RSA

Openssl speed AES

can perform and measure their respective public and symmetric key operations using different parameters. One of the returned results for both operations is a measure of throughput: operations per time (e.g., signatures per second). Run these operations and report your findings. Include two graphs: one that plots the block size vs. throughput for the various AES modes of operations and one that plots the RSA key size vs. throughput for each RSA operation.

Questions:

1. For task 1, viewing the resulting ciphertexts, what do you observe? Are you able to derive any useful information about either of the encrypted images? What are the causes for what you observe?
2. For task 2, why is this attack possible? What would this scheme need in order to prevent such attacks?

3. For task 3, how do the results compare? Make sure to include the plots in your report.

Submission: Write a brief report describing what you did and what you observed. Include any code that you wrote, as well as answers to any questions. Please include any explanations of the surprising or interesting observations you made. Write at a
level that demonstrates your technical understanding, and do not shorthand ideas under the assumption that the reader already "knows what you mean". Think of writing as if the audience was a smart colleague who may not have taken this class.

Describe what you did in sufficient detail that it can be reproduced. Please do not use screenshots of the VM to show the commands and code you used, but rather paste (or carefully retype) these into your report from your terminal. Submit your completed write up to Canvas in PDF format. Any generated images should be included in your report.

Modes of Operations

The modes of operation explored in this assignment (ECB and CBC) are all NIST standards. For diagrams see

Electronic Codebook Mode (ECB)

Cipher Block Chaining Mode (CBC)