CS 727/827: Computer Security
Lab 1: Cryptography

Dongpeng Xu

Due March 11

1 Goal

In this lab, students will first learn the idea of information hiding and practice steganography in a bmp image. They will explore the basic usage of OpenSSL, such as command line utilities about SHA-256, RSA. Students will implement the standard TEA algorithm and then use it for encryption and decryption.

2 Description

1. Download `alice.bmp` from `http://cs.unh.edu/~dxu/cs727/lab1/alice.bmp`.
2. Open the image by a hex editor, write your name (ASCII code) inside the image, and save the new image as `alice-new.bmp`.
3. Compare the new image with the original image. Can you see any differences?
4. Generate digest for the original image and the new image using SHA-256. Compare the two digests. Are they different? Can you observe the avalanche effect?
5. Generate an RSA public and private key pair (2048 bit) using OpenSSL.
6. Use your private key to sign the SHA-256 digest of the new image.
7. Implement the encryption and decryption of standard TEA algorithm as two programs: `tea-enc.c` and `tea-dec.c`. The encryption program should be invoked from command-line by `tea-enc key plaintext` and the output file name is "ciphertext". The decryption program is invoked by `tea-dec key ciphertext` and the output file name is "plaintext".
8. Use your name as the encryption key to encrypt `alice-new.bmp` and then decrypt it.
9. (BONUS) Implement CBC mode TEA encryption and decryption, and use it to encrypt and decrypt `alice-new.bmp`.

3 Submission

Please pack the following files into a compressed file (.zip) and submit to Canvas.

1. A lab report (.pdf) including the description and screenshot of every step.
2. `alice-new.bmp`, your public key, the signed SHA256 digest of `alice-new.bmp`.
3. `tea-enc.c`, `tea-dec.c`, your key, the ciphertext.
4. (BONUS) `tea-enc-cbc.c`, `tea-dec-cbc.c`, your key, the ciphertext.

4 Resource

- BMP file format: `https://en.wikipedia.org/wiki/BMP_file_format`
- Hex editor: Emacs hexl-mode
  `https://www.gnu.org/software/emacs/manual/html_node/emacs/Editing-Binary-Files.html`
  Or other tools you like.
- Steghide: A steganography tool
  `http://steghide.sourceforge.net`
- OpenSSL Tutorial: `https://wiki.openssl.org/index.php/Command_Line_Utilities`
  Commands: `openssl genrsa ...`, `openssl dgst ...`
- TEA: `https://en.wikipedia.org/wiki/Tiny_Encryption_Algorithm`
- OpenSSL, Emacs, GCC have been installed in Agate server.