Lab 6: Block Cipher

50.020 Security

Hand-out: November 3
Hand-in: November 10, 11am

1 Objective

By the end of this lab, you should be able to:

- Implement an ultra-lightweight block cipher

2 Part 0: Do this before you come to class

- Read on PRESENT: an Ultra-lightweight Block Cipher (Section 3):
  - http://yannickseurin.free.fr/pubs/Bogdanov_et_al07_CHES.pdf
- Read Netpbm image format:
  - http://en.wikipedia.org/wiki/Netpbm_format
- ASCII Table for your reference:
  - http://www.unix-manuals.com/refs/misc/ascii-table.html

3 Part I: Implementing PRESENT

- Use Python script `present.py` as a template to implement PRESENT block cipher with 80-bit key and 64-bit block length.
- You need to implement both the encryption and decryption portion.
- Check your result using Appendix 1 of the above paper. Note that `present.py` provides those test cases at the end of the file.
- **Checkoff 1:**
  - Submit your present.py to eDimension.
  - Showed the results of the test cases on Appendix 1.
  - Explain PRESENT algorithm or implementation of your code.
  - Explain where the non-linear part of PRESENT is in the code.
  - Explain how PRESENT algorithm is in comparison to DES and AES.

4 Part II: Implementing ECB Mode

- We are going to encrypt an image (`Tux.ppm`). Check whether you can see the image by typing:

  ```
  $ display Tux.ppm
  ```

  Install ImageMagick if you cannot run display.

  ```
  $ sudo apt-get install imagemagick
  ```

- The above implementation is for a plaintext of 64-bit length. In this section, you need to extend your code so that it can work with plaintext larger than 64-bit. Use Electronic Codebook Mode (ECB) for this purpose.
- Use your ECB mode block cipher to encrypt the file `Tux.ppm`. Decrypt the file and see whether you can still see the same image. Use `ecb.py` and run the file as the following.

  ```
  $ python ecb.py -i [input filename] -o [output filename] -k [key filename]
  -m [mode]
  ```

- **Checkoff 2:**
  - Submit your code to eDimension.
  - Run your encryption and decryption, then show your decrypted image using an image viewer (e.g. `display`).
  - Explain your ECB code.

5 Part III: The Limitation of ECB Mode

- ECB mode allows us to see the plaintext pattern in the encrypted file. In this section, we will try to decipher an encrypted image done by ECB mode.
- Download `letter.e`. This is an encrypted image with a secret message. Your task is to decipher the message written on the image. The image is stored in PBM format and has its header information stored in the file `header.pbm`.
- Write a Python script to extract the image pattern from `letter.e`. You can use `extract.py` as a starting point. *Hint: You can ignore the first few characters which is the header as it has already been given. PBM format only has two values, either 0 or 1. You can assume that there is no space betweeen the data values.*

  ```
  $ python extract.py -i [input filename] -o [output filename] -hh [known header file]
  ```

- **Checkoff 3:**
  - Show the extracted secret message.
  - Explain how you extract the pattern.
  - Submit your Python code and extracted image to eDimension.
  - Propose another Mode of Operation that overcome the limitation of ECB. Explain it.