

Lab 4: Block Cipher

Deadline: 22 June 2024 11:59PM

- Lab 4: Block Cipher
 - Objectives
 - PRESENT
 - PRESENT Overview
 - Implementing PRESENT
 - Implementing ECB Mode
 - Limitation of ECB Mode
 - Submission
 - eDimension Submission

Objectives

- Implement the ultra-lightweight PRESENT block cipher

PRESENT

Read on PRESENT: an Ultra-lightweight Block Cipher:

- <https://www.iacr.org/archive/ches2007/47270450/47270450.pdf> - This is a publication on PRESENT.
- [PRESENT: An Ultra-Lightweight Block Cipher | SpringerLink](#) - This is a conference presentation on PRESENT.
- Read Netpbm image format: http://en.wikipedia.org/wiki/Netpbm_format

You will be able to get all the needed information from the resources above. To aid you, some additional explanation is provided below.

PRESENT Overview

- PRESENT is an example of an SP-network and consists of 31 rounds (and 32 keys). SP refers to [substitution-permutation](#).
- Each of the 31 rounds consists of XOR to introduce a round key, a non-linear substitution (S-Box) layer and a linear bitwise permutation (pLayer).
- Last (32nd) key is used at the end for [post-whitening](#)

S-Box

- S-box: substitution-box aka performs substitution
- In PRESENT, number of input bits = number of output bits = 4
- S-box can be implemented as a fixed lookup table.
- Every word of 4 bits (a nibble) is substituted according to a fixed lookup table.
- <https://en.wikipedia.org/wiki/S-box>

pLayer

- Move bits around (permutation of bits) according to values given in a fixed table
- https://en.wikipedia.org/wiki/Permutation_box

Round Key

- There are a series (key schedule) of 32 keys to be used in PRESENT, to be generated from the 80-bit input key
- The output of each round of key schedule is called round key (key of the round, get it?)
- Schedule for each round i:
 - Rotate left 80-bit key by 61 bits
 - Pass the 4 bits [79...76] through S-box
 - XOR the bits [19...15] with LSB of round_counter (i)
- "Add" round key refers to XOR with round key

Decryption

- Run PRESENT backwards to decrypt. This requires you to be able to invert or "undo" S-box, pLayer, and run the loop in the reverse order as well.

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 linked paper. Note that `present.py` provides those test cases at the end of the file.

Implementing ECB Mode

We are going to encrypt an image (`Tux.pbm`). Check if you are able to open it.

Your `present.py` in the previous section 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) method for this purpose.

Tasks

1. Use your ECB mode block cipher to encrypt the file `Tux.ppm`.
2. 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]
```

Limitation of ECB Mode

ECB mode reveals some side-channel information about the plaintext pattern in the ciphertext. In this section, we will try to learn information about the plaintext image from its ECB ciphertext.

1. Download `letter.e`. This is a secret image, encrypted in ECB mode, with a secret key.
2. Your task is to recover the plaintext of the original image. The image is stored in PBM format and has its header information stored in the file `header.pbm`, that is known to the attacker, i.e. you.
3. Write a Python script to extract the image pattern from `letter.e`. You can use `extract.py` as a starting point.

```
python extract.py -i letter.e -o op.txt -hh header.pbm
```

Hint: You can ignore the first few characters which represent the header as it has already been given. The plaintext PBM image is black and white, i.e. the format only has two values, either 0 or 1. You can assume that there is no space between the data values.

Submission

eDimension Submission

Lab 4 submission: Upload a zip file with the following: file name format:
lab4_name_studentid

Upload a **zip file** with the following:

- `present.py`
- `ecb.py`

- `extract.py`
- `Decrypted letter.e`

Deadline: 22 June 2024 11:59 PM