

Lab 2

Allen Jiang
alljiang@utexas.edu
UT Austin, USA

ACM Reference Format:

Allen Jiang. 2023. Lab 2. In *Proceedings of ACM Conference (Conference'17)*. ACM, New York, NY, USA, 1 page. <https://doi.org/10.1145/nnnnnnn.nnnnnnn>

1 Introduction

In this lab, we implement AES-256 and RSA encryption and decryption in C. The goal of this lab is to familiarize ourselves with implementing cryptography algorithms.

We are using CTR block cipher mode for AES-256 encryption and decryption. The S-box is provided for us, along with a generated key and IV.

2 Our Architecture

The implementation of AES-256 followed the NIST standard for AES-256. In the `aes.c` file, there are a few functions that are used to implement AES-256:

- `calculate_round_keys()`: generates all keys needed given a key and the s-box.
- `encode()`: encodes an input given plaintext, a key, round keys, and the s-box.
- `decode()`: decodes an input given ciphertext, a key, round keys, and the s-box. While this function is not used in this lab due to the use of CTR mode, it is still implemented and tested.

3 Experimental Results

The output of the `aes.c` program passes the given asserts:

- `assert(memcmp(enc_buf, ciphertext[0], 32) == 0);`
- `assert(memcmp(decrypted_text, plaintext[0], 32) == 0);`

The raw output of the `aes.c` program is:

```
Encrypted text:
60 1e c3 13
77 57 89 a5
b7 a7 f5 04
bb f3 d2 28

f4 43 e3 ca
4d 62 b5 9a
ca 84 e9 90
ca ca f5 c5

Decrypted text:
6b c1 be e2
2e 40 9f 96
e9 3d 7e 11
73 93 17 2a

ae 2d 8a 57
1e 03 ac 9c
9e b7 6f ac
45 af 8e 51
```

4 Conclusions

The output of the `aes.c` program matches the expected output. This indicates that the AES implementation is correct.

One possible addition to this AES implementation is to integrate integrity checks. This can be done by computing the SHA-256 hash of the plaintext and including it as a header of the plaintext before encrypting the entire message. The receiver can then compute the SHA-256 hash of the decrypted plaintext without the header and compare it to the included hash header. If the hashes match, the integrity of the message is verified.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Conference'17, July 2017, Washington, DC, USA

© 2023 Association for Computing Machinery.

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...\$15.00

<https://doi.org/10.1145/nnnnnnn.nnnnnnn>