Lab 2
Encryption

1 Introduction and Goals

The goal of this lab is to get you familiar with implementing cryptography algorithms. You will create a RSA and a AES implementation in C.

1.1 Graded Items

For both problems, you will be rewarded extra credit if you go above and beyond the requirements (e.g. supporting any size plaintext, particular fast implementations, adding integrity checking, etc.)

1. Problem 1: Screenshot of the output of the AES test program.
2. Problem 2: Screenshot of the output of the RSA test program.

Lab reports must be in readable English and not raw dumps of log-files. Your lab reports must be typed and must not exceed 6 pages. You are encouraged to use the report template provided on Canvas. Please submit your lab report and all of your code in a zip/tar file on on Canvas as lab2_EID.tar.gz or lab2_EID.zip. Include both files as well as any necessary .h files in your final zip file.

2 Problem 1: AES - Due Friday, September 8

In this problem, you will write a software algorithm that implements AES. We will provide you with the S-box and you will be responsible for implementing encrypting and decrypting of 128-bit messages with your accelerator.
The `part1/aes.c` file provided has the starter code that provides you with the generated key.

1. Submit your screenshot of your AES implementation running as **Figure 1** in your report.

2. Describe how would you integrate integrity checking on top of your AES implementation in your report.

3 Problem 2: RSA - Due Friday, September 15

In this problem, you will write a software algorithm that implements RSA. We will provide you with a generated 128-bit key and you will be responsible for implementing encrypting and decrypting of 128-bit plaintext messages with your accelerator.
The `part2/rsa.c` file provided has the starter code that provides you with the modulus, public exponent, and private exponent of the generated key.

1. Submit your RSA implementation screenshot as **Figure 2** in your report.

2. Describe how would you integrate integrity checking on top of your RSA implementation in your report.