ECE 592 / CSC 591: Cryptographic Engineering and Hardware Security

Assignment 1: Implementing AES

Instructor: Dr. Aydin Aysu
Email: aaysu@ncsu.edu
TA: Digvijay Anand (danand3@ncsu.edu)

1 Introduction and Goals

The purpose of this assignment is to provide a hands-on introduction to the implementation of the Advanced Encryption Standard (AES). AES will appear again and again during multiple components throughout this course. It is, therefore, crucial to have a deep understanding of AES building blocks. And the best way to get a deep understanding of something is to actually code it!

The assignment will walk you through, step-by-step, with a running example. Please follow the document closely and work on the assignment in a sequential manner. But please first read the entire document in one sitting to understand the assignment, what is expected, and to assess how much time would you need to spend on it. **Don't start without reading the entire document first!!!** Use office hours effectively and do not start on the last day...

In this assignment, you are given the task of building a complete AES-128. Luckily, you are building this AES for an embedded device that has a fixed, master secret key that will never change during the life time of the device. Hence, you don't need to implement the key expansion function. Instead, you will be given the keys to be used at each round of the AES.

The secret key for the device you are given is:
'`000102030405060708090a0b0c0d0e0f`' in hexadecimal format. The round keys are therefore:

round[0] key `000102030405060708090a0b0c0d0e0f`

round[1] key `d6aa74fdd2af72fadaa678f1d6ab76fe`
round[2] key `b692cf0b643dbdf1be9bc5006830b3fe`
round[3] key `b6ff744ed2c2c9bf6c590cbf0469bf41`
round[4] key `47f7f7bc95353e03f96c32bcfd058dfd`
round[5] key `3caaa3e8a99f9deb50f3af57adf622aa`
round[6] key `5e390f7df7a69296a7553dc10aa31f6b`
round[7] key `14f9701ae35fe28c440adf4d4ea9c026`
round[8] key `47438735a41c65b9e016baf4aebf7ad2`
round[9] key `549932d1f08557681093ed9cbe2c974e`
round[10] key `13111d7fe3944a17f307a78b4d2b30c5`

Note that the initial transformation of AES uses the master secret key and the next 10 rounds have unique round keys already pre-derived from the master key.

You will verify the functionality of your AES unit using the input plaintext: `'00112233445566778899aabbccddeeff'`.

The initial transformation of AES is XORing input with the master key. Write a code to compute this value. **Question 1:** What is the value (round[1].start) you obtain? This is the value you start your first round of AES.

Now, code and apply the S-box functionality in software and apply the S-box transformation on all bytes. **Question 2:** What is the value (round[1].s_box) you obtain? (Hint the second byte from the left is 'ca').

Next, arrange your round[1].s_box variable as a 4-by-4 matrix each containing a byte (two hexadecimal numbers) as discussed in the lectures to represent the "state". **Question 3:** What is the matrix you obtain? Draw a 4-by-4 table and fill in all the cells.

Next, code and apply the ShiftRows on the table and return the result back to a bitstring. **Question 4:** What is the value (round[1].s_row) you obtain? (Hint: the second byte from the left is '53'.)

Next, code and apply the mix column transformation. **Question 5:** What is the value (round[1].m_col) you obtain? (Hint: the second byte from the left is '72'.)

Next, code and apply the add round key functionality. **Question 6:** What is

the value (round[2].start) you obtain? (Hint: It should be '`89d810e8855ace682d1843d8cb128fe4`'.)

You now completed the first round of AES!!! Do this 8 more times. **Question 7:** What is the value (round[10].start) you obtain? (Hint: the second byte from the left should be '`6e`'.)

Code and apply the final round of AES. Note that, the final round does *NOT* use MixColumns. **Question 8:** What is the output ciphertext you obtain? (Hint: The second byte from the left should be 'c4'.)

Question 9: Report how much time you spent on this assignment in hours.

BONUS: You can copy-paste code from the internet but you need to specify which parts you retrieved and which parts you wrote in your report. If you write the entire code yourself, you get an extra 25 points (out of 100) added on your score.

BONUS: If you implement decryption and show that you can get the plaintext back from the ciphertext you get an extra 25 points (out of 100) added on your score. This can increment on top of the other bonus.

1.1 Graded Items

You will turn in a soft copy report of your results as well as any code to the TA on Moodle. Make sure to include answers to every question in the report.

Lab reports must be in *readable* English and not raw dumps of log files. Your lab reports must be typeset and must not exceed 6 pages. You will be required to use LaTeXto generate the reports. You can use overleaf to generate the LaTeXfile. Please submit your lab report (in .pdf format) and all of your code in a .zip file on Moodle as lab1_YOUR_UNITY_ID_HERE.zip

You can use any software language you prefer to write the programs required to complete for this assignment. If you have any questions about the assignment, please first refer to the TA of the course (Ferhat Yaman, fyaman@ncsu.edu).