CS 5460: Computer Security I
Fall 2019

Assignment 2
Total Marks: 130

In the first two tasks of this assignment, you will get hands-on experience of using a real-life cryptography tool, where you can use one of the following tools that use public key cryptosystem:

1. Devglan
2. Sela

Task 1 [Confidentiality of Message]
Marks: 10

In this task, you will encrypt a message so that only the intended recipient can read it. We generated a pair of 1024 bit public / private key using Devglan (public key is shared with you as a part of the assignment). Encrypt the following message using our public key and Devglan cryptography tool. Choose "RSA" from "Select Cipher Type" dropdown menu before encryption.

Message: *I am sending a confidential message. I have encrypted the message so that only the intended recipient can read it.*

Deliverables: Ciphertext in **cipher_task1** file

Task 2 [Authenticity of Recipient]
Marks: 10

Use an online tool (preferably one of the above) to create a pair of public / private key, and digitally sign the following message using your private key.

Message: *I am signing this message for a class assignment. The digital signature is used to assert that I myself have signed the message.*

Deliverables:

1. The public key in **key_task2** file
2. The signed message in **signed_task2** file

In the file **process_task2**:

3. Link to online tool used for generating the key pair
4. List of options that you selected during the signing process (e.g., number of bits, etc.)

Task 3 [Cryptanalysis]
Marks: 30

Assume, you are a cryptanalyst who have access to a plaintext and corresponding ciphertext message, where ciphertext message is generated using Columnar Transposition and a key of length less than or equal to 7 (e.g., length of key: 'sure' is 4). Now, you need to identify the key used in Columnar Transposition.

In this task, you need to write a computer programming code that takes plaintext and corresponding ciphertext as input, and provides the key for Columnar Transposition as the output (length of key would be less than or equal to 7).

Example:

Inputs:
Plaintext: "you need to find the key"
Ciphertext: "ntneudihyyeodkoefte"

Output:
3421 (sequence from the key: 'sure'). It will be sufficient if your program shows 3421 as the output in this case.

Task 4 [RSA Cryptosystem]
Marks: 80

Implement RSA Cryptosystem that you have learnt in the class, which should include:

- Generation of Public and Private key
- Encryption
- Decryption

Your program needs to have graphical user interface (GUI), one for each of the above operations. See below for further information.

- The first GUI is for key generation, where clicking on a button will generate a pair of public and private keys, followed by storing them in a local directory. The file names could be public.key and private.key. *An optional but useful feature*: Let users store the key-files in a specific directory at your local machine.
- The second GUI needs to have a *text area* to input the plaintext message, a *file browser button* to get the public key stored in local machine (e.g., in public.key file), and an *Encrypt* button to compute the encryption operation. In order to show the ciphertext, the GUI needs to have a text area (uneditable by user).
- The layout of the third GUI is similar to the second one. This GUI is used for performing decryption operation.

Additional Guidelines:

For key generation and encryption/decryption operations, make sure that you would not exceed the length of integer or long-integer permitted/allowed/handled by your programming language/compiler/OS.

Do not use a built-in public key cryptosystem provided as a library function by your compiler/ programming language framework. Rather, we want you to build the code using basic arithmetic operations required for RSA algorithm.

The encryption / decryption operations should be able to handle long inputs, and comply with the generic equation of public key cryptosystem.

Submission and Demonstration

- You will need to submit the noted deliverables for Task 1 and Task 2, and a working version of your code for Task 3 and Task 4 through email to GTA of this course (Manazir Ahsan, email: manazir.ahsan@aggiemail.usu.edu) before **11:59 PM on Friday, October 04**. If needed, add additional instructions for running the code in a 'Read Me' file.

- Your programs for Task 3 and Task 4 need to have required input fields and mechanism to show the output.

- One submission is required from each group (all group members need to be cc'd in the submission email). See Late Submission Policy in course syllabus.

- The subject-line of the email: **CS 5460: Assignment 2 Submission: <Group Name>**

- Each group, with all members present, will need to demonstrate the code on **Monday, October 07**. Attendance during demonstration is required. A student will not receive any marks for the assignment if he/she fails to attend the demonstration.