CSE 565 Fall 2018
Homework 3
Due on October 4, 2018, in class at 9:30am

1. This exercise will allow you to obtain experience with working with different cryptographic tools. For this purpose, you are to write a program in a programming language of your choice using a cryptographic library (or libraries) of your choice. The program should compile and run on one of CSE student machines.

   The program will use different cryptographic functions and measure the speed of each. For that purpose, it should create or read a small file of size 1KB and a large file of size 1MB. These can be randomly generated data or existing files of any type. The program is to implement the following functionalities:

   (a) Create a 128-bit AES key, encrypt and decrypt each of the two files using AES in the CBC mode.
   (b) Create a 128-bit AES key, encrypt and decrypt each of the two files using AES in the CTR mode.
   (c) Repeat part (b) with a 256-bit key. two files using AES in the CTR mode.
   (d) Compute a hash of each of the files using hash functions SHA-256, SHA-512, and SHA3-256.
   (e) Create a 2048-bit RSA key, encrypt and decrypt the files above with PKCS #1 v2 padding (at least v2.0, but v2.2 is preferred; it may also be called OAEP).
   (f) Repeat part (e) with a 3072-bit key.
   (g) Create a 2048-bit DSA key, sign the two files and verify the corresponding signatures. If creating a key takes two parameters, use 224 bits for the exponent sizes. If the hash function algorithm needs to specified separately, use SHA-256.
   (h) Repeat part (g) with a 3072-bit DSA key (if the second parameter is required, use 256).

   Include simple checking for correctness (e.g., that computed ciphertexts decrypt to the original data).

   For each encryption algorithm, measure (i) the time it take to generate a new key, (ii) the time it takes to encrypt and (iii) the time it takes to decrypt each of the two files. Compute encryption and decryption speeds per byte for both of the files. Similarly, for the signature scheme, measure the key generation time and the time to produce and verify a signature for the two files (the total time and per-byte time). Finally, for the hash functions, measure the total time to compute the hash of both files and compute per-byte timings.

   In your homework, (i) include the results of your measurements and computations as instructed above, (ii) provide all source code, (iii) specify on which CSE machine your code runs, and (iv) provide your comments regarding the following performance aspects and any justification of the observed results that you can offer:

   - how per byte speed changes for different algorithms between small and large files;
   - how encryption and decryption times differ for a given encryption algorithm;
   - how key generation, encryption, and decryption times differ with the increase in the key size;

- how hashing time differs between the algorithms and with increase of the key size;
- how performance of symmetric key encryption (AES), hash functions, and public-key encryption (RSA) compare to each other.

2. Consider the set of rights {read, write, execute, append, list, modify, own}.

   (a) Using the syntax for describing commands that alter the access control matrix in DAC, write a command *delete_all_rights*$(s_1, s_2, o)$. This command causes subject $s_1$ to delete all rights the subject $s_2$ has over object $o$ only if $s_1$ has *modify* rights over $o$ and $s_2$ does not have *own* rights over $o$.

   (b) Now suppose that all rights above except the *own* right can have a copy flag set (for right $r$, $r^*$ will mean that the copy flag is set). Write a command *copy_all_rights*$(s_1, s_2, o)$ that copies all rights that $s_1$ has over $o$ to $s_2$ in such a way that only those rights with an associated copy flag are copied and the new copy doesn't have the copy flag.

   You can omit copying unmodified matrix cells to the new matrix.