

RSA and AES

Lab 2

Due 9/26/03

Objective:

The objectives of this lab are 1) to improve your knowledge of the RSA cryptosystem, 2) to get familiar with a public domain implementation of RSA, 3) to compare the performance of RSA and AES, and 4) to design a small application involving RSA and AES.

Instructions:

After review, we highly recommend that you download RSAREf2 at <http://www.funet.fi/pub/crypt/cryptography/asymmetric/rsa/rsaref2.tar.gz> and use that instead of the GnuPG code, since it will make your lives much simpler (for this and future homeworks). You can of course continue to use GnuPG code, but it'll make your life much more difficult.

1. Download the source code.
2. When you download it, rename it to rsaref2.tar (since its not really gzipped).
3. untar the code:

```
tar xf rsaref2.tar
```

4. read the readme.txt in in doc/ and then build rsaref2 accordingly.
5. For part 2, your job is to use the RSA functions to generate keys and encrypt the specified files. So that means you should write your OWN program that uses this public implementation and its functions to do what you need to do.

Your Task:

Part I:

1. Which key in RSA would give you a cipher text that is identical to the plaintext? What is the minimum recommended key length for a secure RSA transmission?
2. It is well known that RSA is much slower than DES or AES. Then what are the advantages in using RSA over AES/DES?
3. How can RSA be used to provide authentication and confidentiality between two parties? (don't take into account possible attacks on RSA)
4. Suppose your public RSA key is $(e, n) = (24523, 47880953)$. Encrypt the following plaintext: 030120. Show your work (using the modular exponentiation algorithm) and the resulting encrypted text.
5. What types of numbers are 41, 257, & 65537 and what are they used for in some RSA implementation (hint: binary)? The more specific you are, the more credit you get.

Part II: In this part you will do a small experiment to compare the performance of AES and RSA.

1. First generate text files with the following sizes, 8 bytes, 32 bytes, 128 bytes, 512 bytes, 2 KB (1KB = 1024 bytes), 8 KB, 32 KB, 128 KB, 512 KB, 1MB, 4MB, 16MB, 32MB, 64MB. (Note: 1 text character = 8 bytes)
2. Encrypt these files using RSA and AES, and record the time take for encryption each file separately for RSA and AES. (You can choose your own keys for AES and RSA, but it is critical that you use a large n and use reasonable / realistic keys for RSA, otherwise the results of this experiment will not be correct and some points may be detected)

Hint: use time.h in your c/c++ code to keep track of time, the test code from AES assignment uses function from time.h to keep track of encryption and decryption time.

3. Generate a table using the recorded timing information and create a graph from the table.
4. What do you observe from your graph and why?

For this part of the homework, you are required to hand in the main function/program for file encryption (including the selection of the keys for AES and RSA), the table, the graph, and an analysis/explanation of the graph comparing AES and RSA performance.

Part III:

Design an algorithm using RSA and AES to solve the following problem:

XYZ enterprise wants to develop a file encryption and authentication utility for their employees. This utility should allow users to encrypt/decrypt their files, allow other users belonging to the same group to decrypt files if allowed by the user who encrypted the file, and should allow any user belonging to “administrative” group to decrypt any user file. For example: Let’s say “Joe” is a user in XYZ who belongs to “accounting” group. Joe should be able to encrypt a file “fin_status” and specify if other users belonging to the same group could decrypt “fin_status”. And (very important) if Joe is fired, XYZ should be able to recover any files encrypted by Joe without interrogating (both physical and mental) him for encryption key.

You must state your algorithm clearly to receive credits. Note you don’t have to implement anything yet. Just design a solution and describe how it will work in detail.

Submission:

Please send all that you have to hand in for this homework as a zipped file to Konstantin Rozinov at konstantin@rozinov.com by the due date (midnight) and use the following subject line: “CS391/CS681: LAB#2”. Place a hard copy in the ISIS drop box by 6 pm on the day after the due day. **Please make sure the graders are able to COMPILE ANY CODE you have (give instructions or a makefile) under LINUX so that your work can be graded.** If you have questions, email Shihan (shihanyu@hotmail.com), Konstantin (konstantin@rozinov.com), or see Lok in person.