Crypto Report - 3

Team Anonymous:
CS14B023, Rahul Kejriwal
CS14B039, Bikash Gogoi

Q15. Have you thought about efficient implementations? If so, demonstrate and document tricks in software and cipher design choices that have makes your cipher efficient to implement on the target platform. (you can either choose to optimize to take minimum memory or minimum operations)

A15. We are using bitwise operations instead of arithmetic operations wherever it is possible. Bitwise operations are used very extensively as they are faster in execution. We also use gcc compiler optimizations to optimize the code.

We use separate lookup tables for substitution and permutation to reduce computation times.

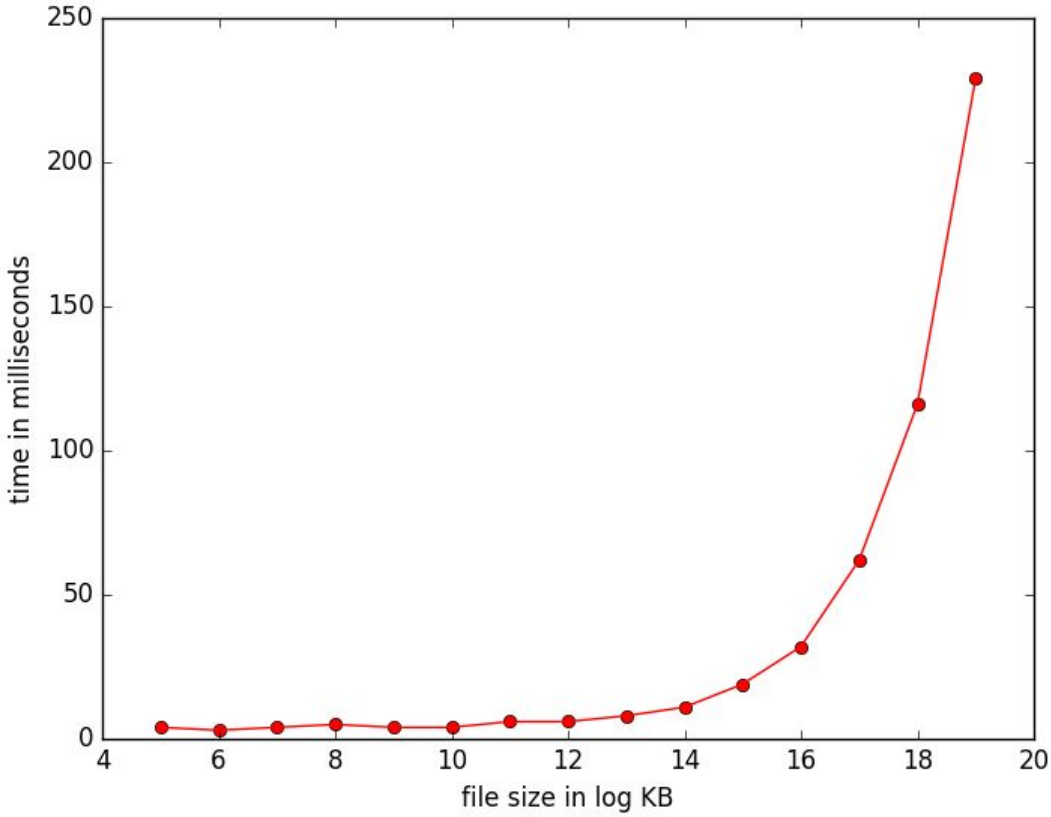
We thought of having lookup tables for doing substitution and permutation together which would greatly reduce the no. of operations (specially bit operations) that were being done. However, this caused a great increase in size of lookup tables (64kB from earlier 4kB). Thus, the tables would not be fully cached and speed might decrease. Thus, we abandoned this idea.

Q16. Revisit all questions in Section 1 and Section 2. If you decide to make changes in any of the answers, mention them here and justify why you are making the changes.

A16. We did not make any changes to our cryptosystem pertaining to the older questions.

Q17. For files of sizes 2n, where n=5 to 20, plot the file size vs the encryption time on the target platform (Intel 64-bit x86).**

A17.



Q18. Demonstrate the working of your cipher. Encrypt file alice.txt and save the result in enc_alice.txt. Decrypt enc_alice.txt and save the decrypted result in dec_alice.txt.

A18. PFA the files alice.txt, enc_alice.txt and dec_alice.txt. We used a random text generator to generate the random text in alice.txt.