

CS 651S - Status Report

Shaun Arnold, Thomas Daniels, Chris Taylor, Mike Walker

April 3, 2001

Report as of 4/3

Our group has been working on cryptography and cryptanalysis. We have decided to implement the RSA algorithm, the rotor engine, poly and mono alphabetic ciphers, and produce statistical tools for cryptanalysis. We then plan on analyzing the effectiveness of these tools on the given cryptographic algorithms where appropriate.

RSA implementation is finished, tested, and working correctly. Normal cryptanalysis has not proved helpful in cracking the ciphertext produced by our RSA implementation.

A monoalphabetic Ceasar cipher was implemented. The algorithm takes text and a number as input, which seeds the offset, to produce ciphertext. The program has been tested and appears to work correctly.

A monoalphabetic (not Caesar) has been implemented. It accepts a 26 letter permutation of the alphabet and supports encryption and decryption. It has been tested and works correctly. No errors have been found in the code.

A polyalphabetic algorithm was created. This Vignere cipher has been implemented, tested and works correctly. The program takes a key and plaintext as input to produce ciphertext.

The rotor engine has been implemented. This software version allows any valid rotor connection configuration to be set by simply entering the numbers into the appropriate array in the code. The user gives a key which sets the offsets of the rotors and there is a mode for encryption and a mode for decryption. The implementation works correctly and no code errors have been encountered.

A statistical frequency analyzer has been implemented for text files. It's purpose is to help break monoalphabetic ciphers and also to aid the gathering of statistical tendencies of a particular genre of text. The program analyzes ciphertext to produce statistical information regarding occurances of single, double and triple letter combinations. The

program works correctly and has no errors discovered.

A statistical key analysis tool has been implemented. This program analysis repetitions of letter combinations in ciphertext to produce meaningful statistics. It produces statistics of which two letter combinations repeat, where they are in the text, and how many times that combination occurred. The intent is to find keysize for polyalphabetic ciphers. The program works correctly and appears to have no errors. Modifications are pending to make command line execution similar to the previous analyzer.

Ciphertext from the monoalphabetic cipher has been submitted and readily broken. The statistical frequency analyzer in tandem with the unix tr utility and a bit of cleverness seem to overwhelm the complexity of the monoalphabetic cipher.

Ciphertext from the polyalphabetic cipher has been submitted and partially broken. The correct key has been found for two of the three ciphers originally submitted. Since the original submission of these three ciphers, another cipher has been submitted per request.

All ciphertext submitted to other members takes advantage of white space in the ciphertext as it appears in the plaintext. The exception is the final requested submission for the polyalphabetic cipher. It is possible that keylength will be determined but the actual key may not be recovered with reasonable effort for this final case.

A cracking program was written to exploit a security vulnerability concerning pseudo-random number generation in the implementation of the RSA algorithm. An RSA ciphertext message was sent via e-mail. Two assumptions were made concerning this e-mail. It was sent directly to the cracker (a group member in this case), but is assumed a standard e-mail message is easily intercepted over a network. Also, it is assumed that the message was sent relatively soon after the cipher was produced. The cracking program has searched a large portion of the search space for the random number generation, but has not yet been able to crack the message. A second, more methodical attempt at cracking the code should be completed by Monday, April 9th.