Assignment: Final Project Report
Class: CS 7370 – Advanced Computing and Networking
Team: Matthew Stroble and Hannah Reinbolt
Date: 4/27/2022

Automatic Encryption Breaker

Code:
https://github.com/CySpiegel/COMP-7370-FinalProject.git

Problem:
When finding a file that contains encrypted data, it may be difficult to discern what cypher or encryption method was used to encrypt. It can take hours of forensic work to find the correct encryption type, and then even longer to decrypt the data.

Project Summary/Justification:
The Automatic Encryption Breaker, or AEB, will evaluate the contents of a file and use frequency analysis to determine exactly what type of encryption is used. After the encryption type is identified, it will attempt to use brute force or frequency analysis to unencrypt the data. This can be done in seconds, which could save hours of valuable forensic work and properly automate tasks so the human mind can work on other more complicated decision making. This tool may also find the solution to an encrypted file that the user is not able to see; thus providing a solution when it would not have been found before. This project focuses on detecting and cracking shift and substitution ciphers with the English language. Shift cipher can be defined as Caesar cipher and rot ciphers, as the cipher simply modular rotates a message by key N. (N is within the key space) This tool includes spaces and periods in the key space for a total key space of 28. (key space can be easily increased) Substitution ciphers are a bit different, as they can scramble the letters into different positions, it's not a simple shift. Frequency analysis is used to determine what letters go where.

Design Summary:
The AEB tool is built in python with two pip libraries installed, pyperclip and english-words. AEB is intended to be run in the command line of a linux environment but can theoretically be run on Windows too since it utilizes python 3. A dynamic menu system is connected to the frequency analysis and decrypter tools to help the user quickly analyze a file. Most files tested with this program were full text books from the Gutenberg Project. Frequency analysis works best where there are many characters to analyze. The more cipher text there is to analyze the more accurate the fingerprinting. The current encryption fingerprinter can distinguish between a shift cipher and substitution cipher. The encryptions methods both use the same symbol set of characters for simplicity but it should be noted that it is not a strict requirement. The symbol set used for both encryption types will encrypt A-Z with spaces and periods. This was done to give both ciphers the maximum chance of unreadability in there cipher text form. The AEB will require a library of encryption profiles that it will use to attempt to identify an encrypted file's contents. These will be common traits of certain encryption types. These profiles will be modular allowing easy updates; making the tool more versatile for the future. The AEB tool includes a

decryption library, where it automatically runs through brute force techniques to solve that encryption type. The library also includes a complex frequency analysis solver that can be more helpful with complex ciphers like substitution. The solutions are output to a file for the user and tested against a dictionary to find out if any solutions are close to English words. The program recommends a "best-guess" solution to the user if it is close to English. Otherwise it will be up to the user to identify the correct solution. Correct solutions can be saved to a file upon user request.

Fingerprinting or Frequency Analysis Modeling:

Initially we researched a lot into frequency curve analysis to see how far away we are from the English text but this did not prove to be useful in identifying the differences between encryption types. The fingerprinting of cipher text requires a large amount of analysis of English Characters. This means there must exist a plane text frequency analysis table for all letters A-Z with space and period character. This frequency table was built over the analysis of 209 full English textbooks from the Gutenberg Project to get the correct frequency table for an unencrypted book. The fingerprinting module will build a second frequency analysis table off the cipher text being analyzed. To make use of the frequency tables that have been created both tables are arranged from highest frequency character to lowest frequency character. This rearranged frequency table of character will give us a partial key to decrypt both text correctly but will not allow us to distinguish between different types of encryption. Instead the fingerprinting module will use the rearranged encrypted frequency map from high to low as a sort key to check characters. For the shift cipher we walk down the frequency map and count the distance in shift from the first character in the encrypted frequency map to the second character in the frequency map and build a table of key offsets. For each key we build a strength score to see how many times we can walk down the frequency table and computing the distance of the key. These keys are then put into a table with the number of times the distance is the same distance from one character to the next in the frequency map. The end result when sorted will be a table of keys with the highest to lowest strength score for a given key of the shift cipher. The table will show that for any text analyzed will give the key and score next to it like (11, 10), with 11 being the key and the number of times a comparison from the encrypted frequency table entries showed to be a distance of 10 in the shift that exists between them. This means that with high degree of accuracy that the current text being analyzed is a shift cipher and its key being 10. What we found interesting was when we applied the same technique to the substitution cipher. Out of all the various random encrypted keys we used with the substitution cipher we noticed that using the same distance measurement that substitution strength indicator always read the following (8, 3). This means that for any encrypted text using the substitution if the highest strength indicator was 3 it must have been a substitution encryption the fingerprinting module was analyzing. For shift ciphers the strength indicator was always above 10. We hypothesize there must exist for any encrypted text that a unique strength indicator must exist using this method for any 1:1 encryption method. Further development is needed to allow for the distinguishment of a 1:N encryption method like Hill Cipher. We suspect that the Hill Cipher will have a higher strength indicator than substitution but no larger than shift.

Decryption:

Decryption is automatically done in the AEB program using two main methods. Brute force (or round robin approach) and frequency analysis. Brute force is mainly used with the shift cipher, as all possible shifts are applied to the message and saved. Each shift is sent to a special function that separates each

word (space delimited) and checks each word against an English dictionary. (english-word) Each correct word is counted as a point towards that solution and at the end of the decryption, the result with the most points is awarded as “best guess”. This is especially good if the user has to decrypt large files and it would take a lot of time to look through all solutions manually.

How we break substitution cipher done through heavy frequency analysis. As stated before the method used by the fingerprinting module, we build a frequency table of the cipher text and order it from highest to lowest frequency. For substitution cipher we stick with the same encryption symbol set A-Z with space and period characters. This means that any special symbols not in the frequency table are unencrypted and in their original position. Cracking substitution cipher is not something that can be done through brute force, it must be done in stages and done methodically. We build our decryption key from the frequency analysis table that was arrange from highest frequency to the lowest frequency. We know that in the analysis from the 209 plane text books the space character is by far the most used character in text. This will always be the first character in the frequency table. We apply the current decryption key to the cipher text. This will give us about a 67% success rate in cracking the substitution cipher just off pure frequency analysis alone. The next step is looking for key patters if any is found. This allows us to key in on a particular pattern that may appear in the text for example, http://www.somesite.com” this allows us to key in on specifically the http and www characters and update the decryption key. This will give us an additional 5 character that are almost guaranteed to be correct, however this is not always present and if this pattern is not found it will skip and move onto the next stage. If found we update the decryption key and attempt to decrypt the book. Next is looking for the patter of the word “Ebook”. We do this by treating characters and numbers and “Ebook” looks like this 01223. This is word pattern analysis. We know that according to the frequency map that E will become known after the first initial frequency analysis because after the space character it will always be the character E. In truth the word Ebook will look like E0112 since for the first few characters in the frequency table and in our decryption key we never update those characters. The rest of the steps is looking for word pattern analysis and checking with high probability of what the remaining words might be using this method. In this step the substitution cracker will go through and continue word pattern analysis for every word in the text. When the program reaches the end it will apply the learned decryption key to the original cypher text for the final crack that results in a 96% accuracy in recovering the initial plane text. It should be noted that anything over 75% accuracy is human readable and you can infer the meaning of the text without needing to decrypt the remaining text.

Dynamic Menu System:
The menu system is dynamically designed to allow the developer the freedom to create custom menus with custom options upon demand. This was necessary because of the complexity of menus needed. The menu is also able to be easily modified for future additions. This menu system is special because it also contains the main program logic. There are 4 steps to the current logic.

Step 1: start the program and request a file. This step will ask the user for a file and check to make sure the file is valid. If not then it will ask for a valid file.

Step 2: the program fingerprints the file and determines what cipher it is using. Here the fingerprinting functionality is called upon the file. The whole file is opened and worked on in memory to increase

speed. Frequency analysis models are used, as explained above, to determine what cipher is used. The result will return as “shift” or “substitution”.

Step 3: choose decryption and start decryption. Here the program asks what decryption method to apply to the file. Shift and substitution are available but also an option to let the program decide based on the results of the fingerprinting. If the guess option is selected then the menu system will choose the correct decryption type and begin the process. Substitution uses frequency analysis and only has one solution; this can also be correctly used for shift cipher. Shift cipher also has the round-robin approach and this result will have 28 results for each shift plus a best guess. The decryption process only takes a second on most files but may take longer on bigger files.

Step 4: decryption results. Here the results are sent to a file for the user to view. The results are not output to screen because of possible length.

Throughout the whole system are options to quit the program. The program itself will never quit unless the user selects the quit option or the program hits an unknown fail error.

Conclusion:

The Automatic Encryption Breaker tool performs as expected and successfully identifies and decrypts shift and substitution encrypted data for the user in a timely manner. This tool was built dynamically and can be easily added to for future improvements and additional supported ciphers/encryptions. The frequency analysis identification model used to identify ciphers can in theory support many-to-one ciphers like the hill cipher, and other ciphers. Future work would include experimenting with adding additional ciphers to the frequency analysis identification model and additional decryption tools for automatic decryption. One fall back is that small messages do not work well with this system. This system is built for bigger messages and files rather than small one sentence messages. This is because the frequency analysis identification model needs more information to work properly and see patterns. Overall this tool can prove useful to professionals and can save many hours in forensic work by automating tasks that in the past were manual. In addition this tool can correctly decrypt data that may be overlooked by professionals, increasing the success of data retrieval.

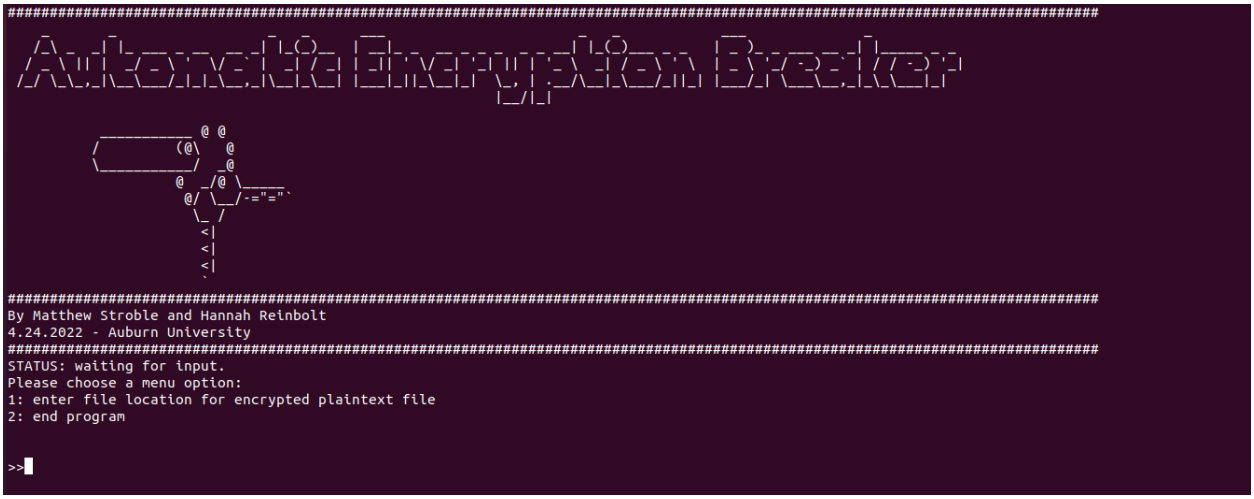
Member Responsibilities/Tasks:

Below are the tasks that each member worked on. The entire project was worked on by both members as evenly as possible.

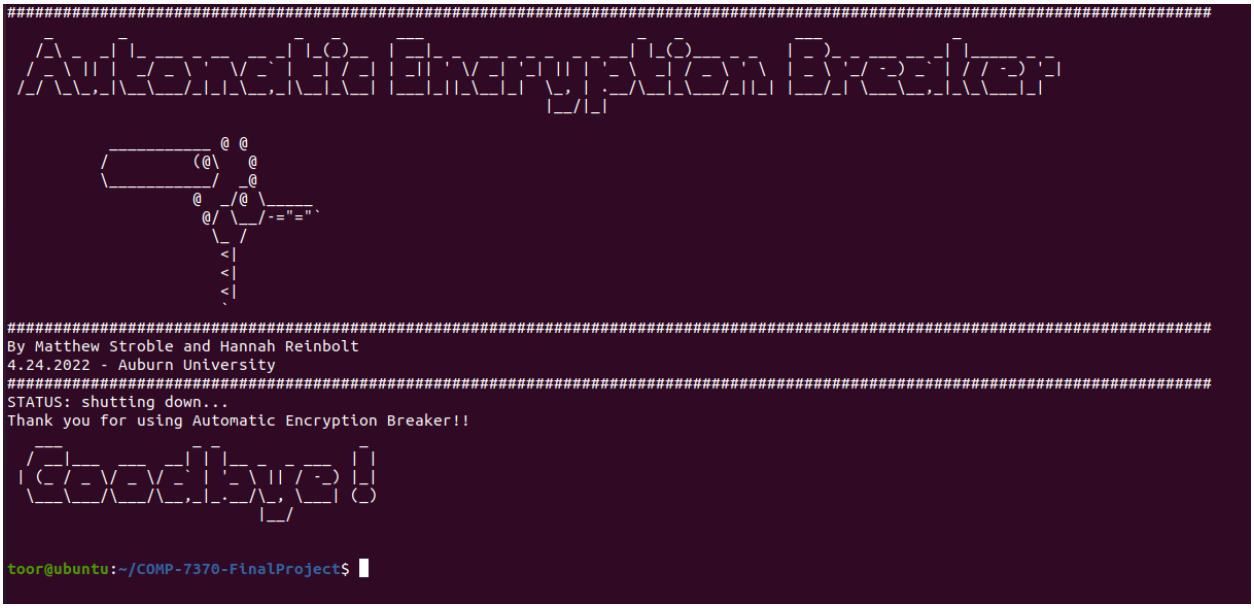
Task	Team Member
Fingerprinting with frequency analysis	Matthew
Frequency analysis decrypter	Matthew
Shift cipher encrypter and decrypter	Matthew and Hannah
Dynamic menu system	Hannah
Best guess analysis	Hannah
Reports and Presentations	Matthew and Hannah

Tables and Snapshots:

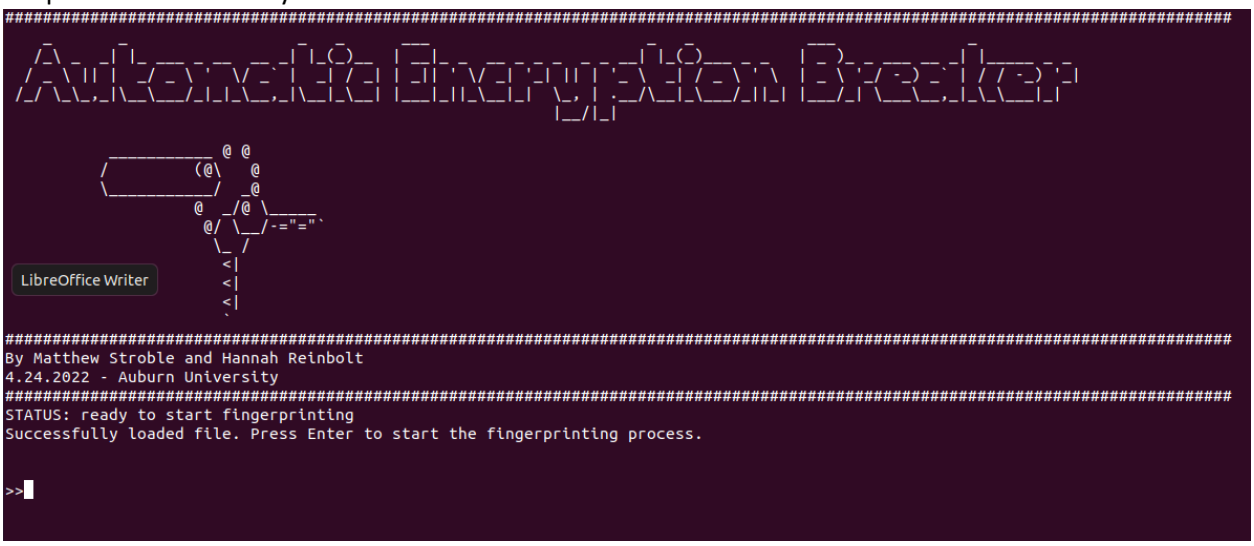
Snapshot 1: Main Menu



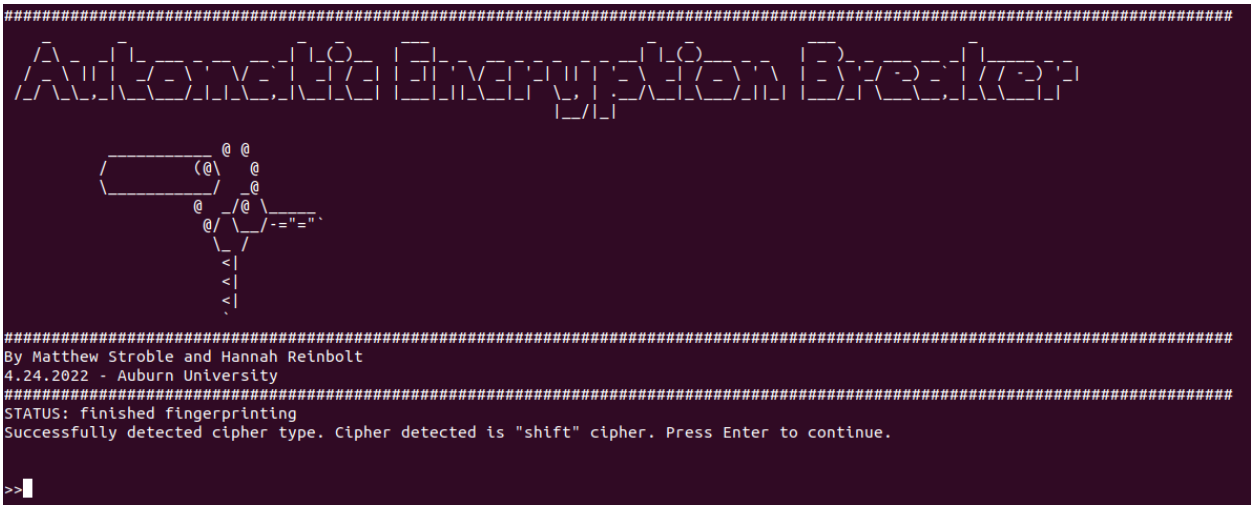
Snapshot 2: End Program



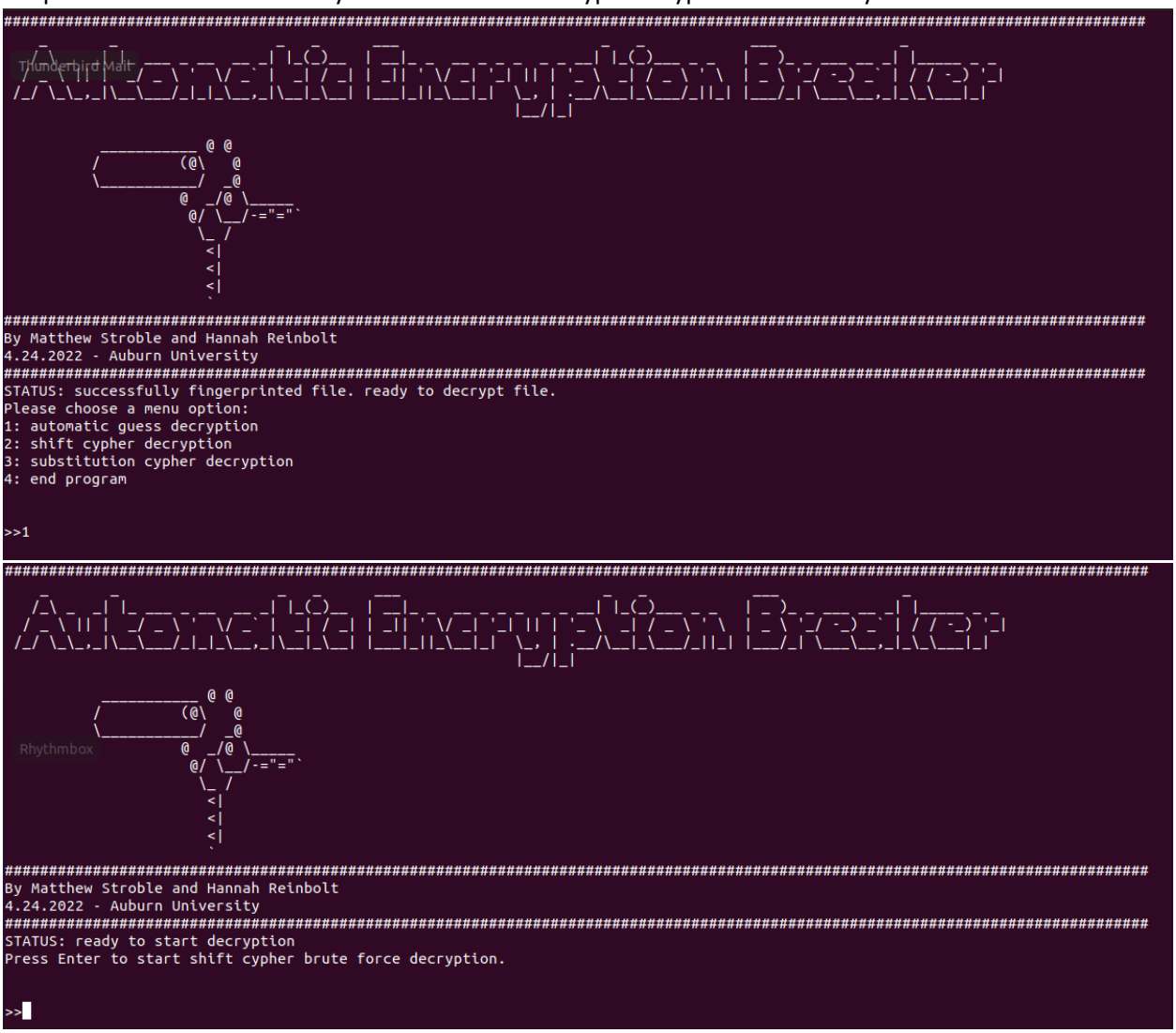
Snapshot 3: Successfully loaded file



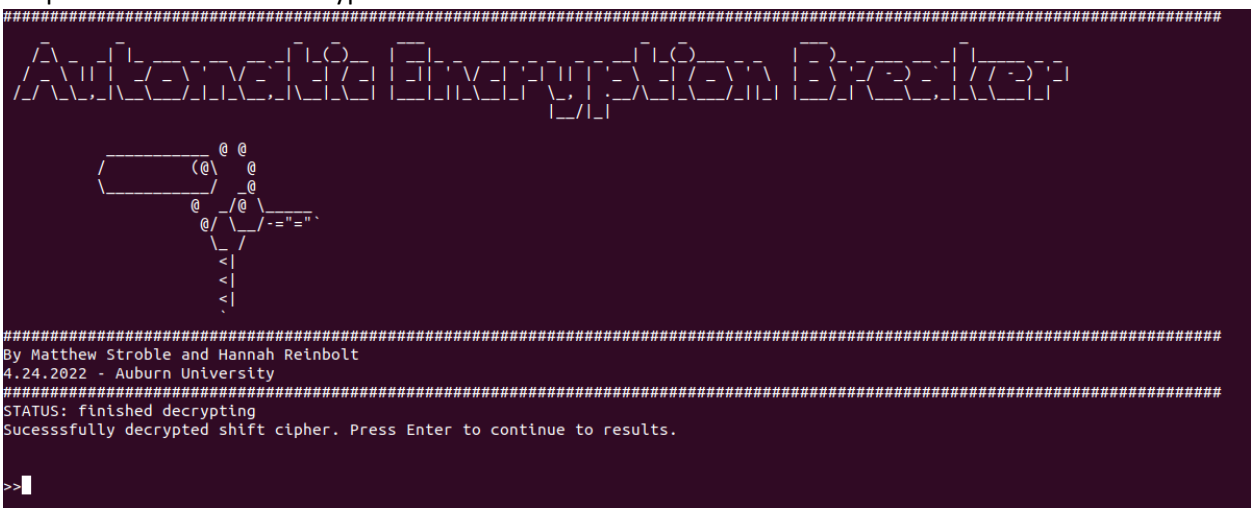
Snapshot 4: Successfully Fingerprinted a File



Snapshot 5: Menu Successfully Choose Correct Decryption Type Automatically



Snapshot 6: Successful Decryption



Snapshot 7: Shift Cypher Results Menu

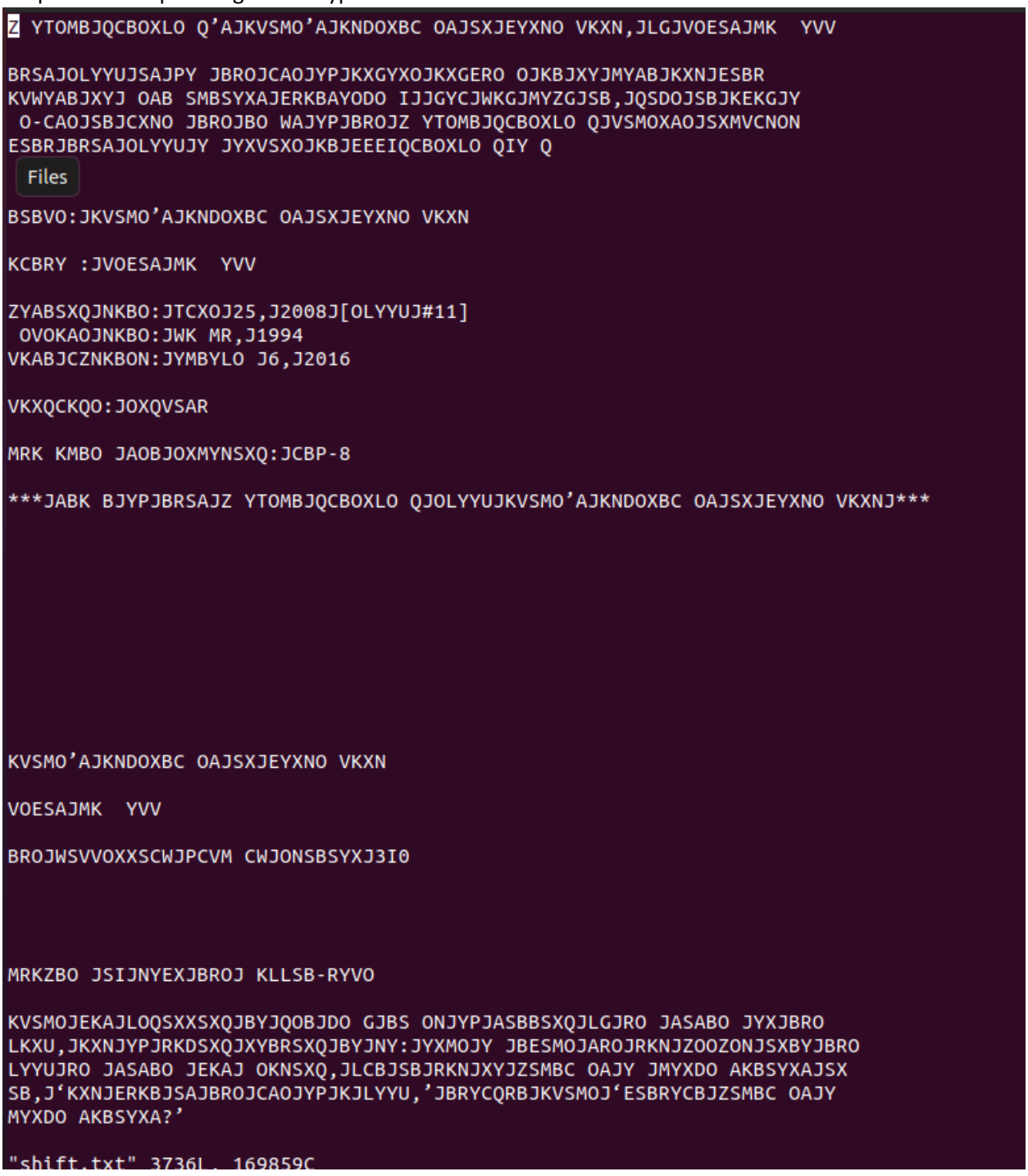
Snapshot 8: Top of All Solutions File After Shift Decrypt (only part of first shown here)

```
ROT:0 <feff>Z YTOMBJQCBOXLO Q'AJKVSMO'AJKNDOXBC OAJSXJEYXNO VKXN,JLGJVOESAJMK  YVV

BRSAJOLYYUJSAJPY JBROJCAOJYPJKXGYXOJKXGERO OJKBJXYJMYABJKXNJESBR
KVWYABJXYJ OAB SMBSYXAJERKBAYODO IJJGYCJWKGJMYZGJSB,JQSDOJSBJKEKGJY
 O-CAOJSBJCXNO JBROJBO WAJYPJBROJZ YTOMBJQCBOXLO QJVSMOXAOJSXMVCNON
ESBRJBRSAJOLYYUJY JYXVSXOJKBJEEEIQCBOXLO QIY Q
 Files
BSBVO:JKVSMO'AJKNDOXBC OAJSXJEYXNO VKXN

KCBRY :JVOESAJMK  YVV

ZYABSXQJNKBO:JTCXOJ25,J2008J[OLYYUJ#11]
 OVOKAOJNKBO:JWK MR,J1994
VKABJCZNKBON:JYMBYLO J6,J2016

VKXQCKQO:JOXQVSAR

MRK KMBO JAOBJOXMYNSXQ:JCBP-8

***JABK BJYPJBRSAJZ YTOMBJQCBOXLO QJOLYYUJKVSMO'AJKNDOXBC OAJSXJEYXNO VKXNJ***









KVSMO'AJKNDOXBC OAJSXJEYXNO VKXN

VOESAJMK  YVV

BROJWSVVOXXSCWJPCVM CWJONSBSYXJ3I0




MRKZBO JSIJNYEXJBROJ KLLSB-RYVO

KVSMOJEKAJLOQSXXSXQJBYJQOBJDO GJBS ONJYPJASBBSXQJLGJRO JASABO JYXJBRO
LKXU,JKXNJYPJRKDSXQJXYBRSXQJBYJNY:JYXMOJY JBESMOJAROJRKNJZOOZONJSXBYJBRO
LYYUJRO JASABO JEKAJ OKNSXQ,JLCBJSBJRKNJXYJZSMBC OAJY JMYXDO AKBSYXAJSX
SB,J'KXNJERKBJSAJBROJCAOJYPJKJLYYU,'JBRYCQRBJKVSMOJ'ESBRYCBJZSMBC OAJY
MYXDO AKBSYXA?'
```

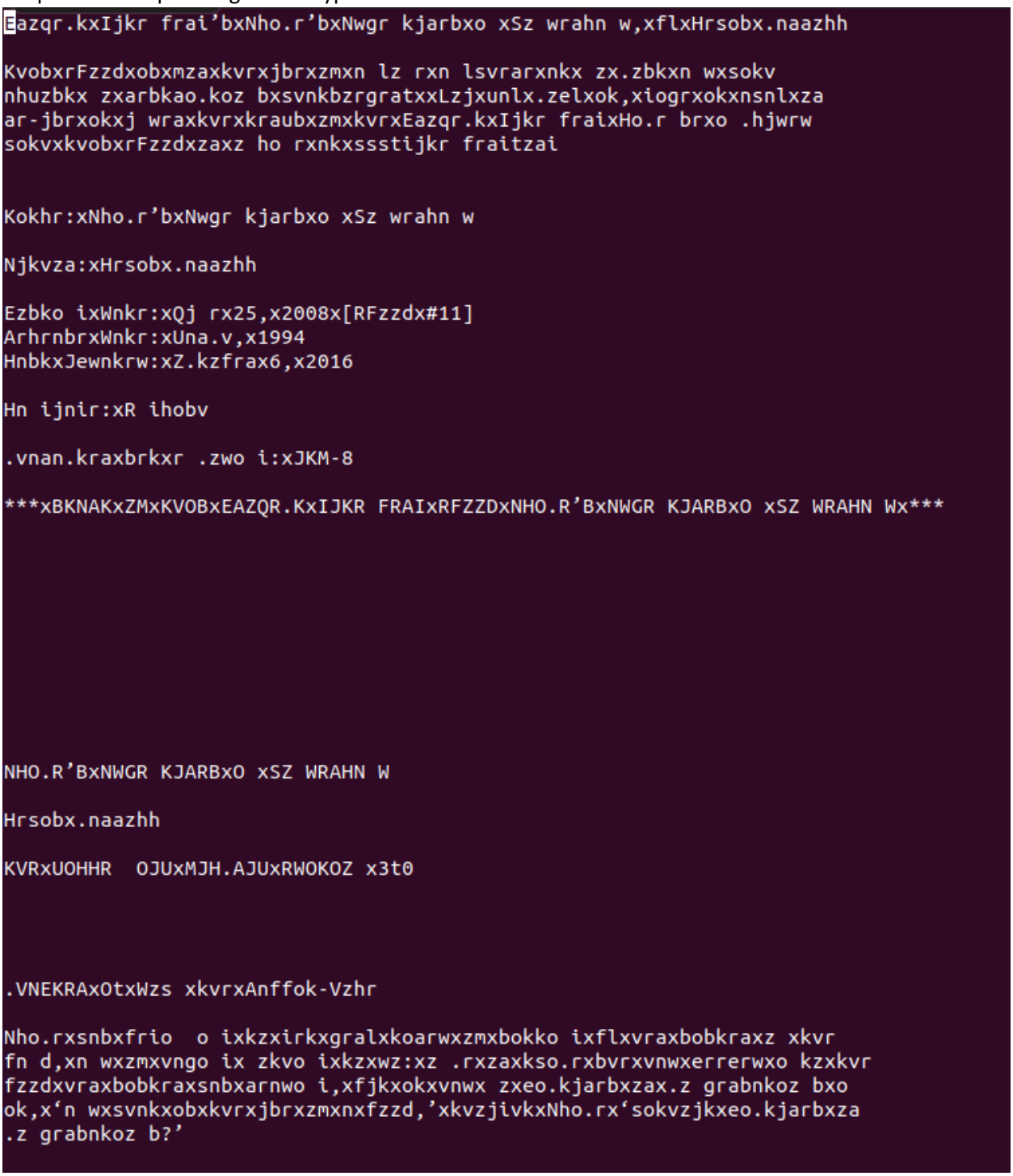
Snapshot 9: Top of best Guess File after Shift Decrypt

```
KEY: 10 <feff>project gutenberg’s alice’s adventures in wonderland, by lewis carroll

this ebook is for the use of anyone anywhere at no cost and with
almost no restrictions whatsoever.  you may copy it, give it away or
re-use it under the terms of the project gutenberg license included
with this ebook or online at www.gutenberg.org


title: alice’s adventures in wonderland

author: lewis carroll

posting date: june 25, 2008 [ebook #11]
release date: march, 1994
last updated: october 6, 2016

language: english

character set encoding: utf-8

*** start of this project gutenberg ebook alice’s adventures in wonderland ***









alice’s adventures in wonderland

lewis carroll

the millennium fulcrum edition 3.0




chapter i. down the rabbit-hole

alice was beginning to get very tired of sitting by her sister on the
bank, and of having nothing to do: once or twice she had peeped into the
book her sister was reading, but it had no pictures or conversations in
it, ‘and what is the use of a book,’ thought alice ‘without pictures or
conversations?’
```

Snapshot 10: Top of Original Encrypted Shift Book Before AEB was Run



Snapshot 11: Top of Original Encrypted Substitution Book Before AEB was Run



```
Eazqr.kxIjkr frai'bxNho.r'bxNwgr kjarbxo xSz wrahn w,xflxHrsobx.naazhh

KvobxrFzzdxobxmzaxkvrxjbrxzmxn lz rxn lsvrarxnkx zx.zbkxn wxsokv
nhuzbkx zxarbkao.koz bxsvnkbzrgratxxLzjxunlx.zelxok,xiogrxokxnsnlxza
ar-jbrxokxj wraxkvrxkraubxzmxkvrxEazqr.kxIjkr fraixHo.r brxo .hjwrw
sokvxkvobxrFzzdxzaxz ho rxnkxssstijkr fraitzai


Kokhr:xNho.r'bxNwgr kjarbxo xSz wrahn w

Njkvza:xHrsobx.naazhh

Ezbko ixWnkr:xQj rx25,x2008x[RFzzdx#11]
ArhrnbrxWnkr:xUna.v,x1994
HnbkxJewnkrw:xZ.kzfrax6,x2016

Hn ijnir:xR ihobv

.vnan.kraxbrkxr .zwo i:xJKM-8

***xBKNAKxZMxKVOBxEAZQR.KxIJKR FRAIxRFZZDxNHO.R'BxNWGR KJARBxO xSZ WRAHN Wx***









NHO.R'BxNWGR KJARBxO xSZ WRAHN W

Hrsobx.naazhh

KVRxUOHHR  OJUxMJH.AJUxRWOKOZ x3t0




.VNEKRAxOtxWzs xkvrxAnffok-Vzhr

Nho.rxsnbxfrio  o ixkzxirkxgralxkoarwxzmxbokko ixflxvraxbobkraxz xkvr
fn d,xn wxzmxvngo ix zkvo ixkzxwz:xz .rxzaxkso.rxbvrxvnwxerrerwxo kzxkvr
fzzdxvraxbobkraxsnbxarnwo i,xfjkxokxvnwx zxeo.kjarbxzax.z grabnkoz bxo
ok,x'n wxsvnkxobxkvrxjbrxzmxnxfzzd,'xkvzjivkxNho.rx'sokvzjkxeo.kjarbxza
.z grabnkoz b?'
```

Snapshot 12: Top of Best Guess File after Substitution Book was Decrypted with AEB Tool

```
Procect Guteneerg's Alice's Adventures in Wonderland, ey Lewis carroll

This eEook is for the use of anyone anywhere at no cost and with
almost no restrictions whatsoever.  You may copy it, give it away or
re-use it under the terms of the Procect Guteneerg License included
with this eEook or online at www.guteneerg.org
```

Files

```
Title: Alice's Adventures in Wonderland

Author: Lewis carroll

Posting Date: Cune 25, 2008 [EEook #11]
Release Date: March, 1994
Last Updated: Octoeer 6, 2016

Language: English

character set encoding: UTF-8

*** START OF THIS PROCEcT GUTEnEERG EEOOK ALIcE'S ADVEnTURES In WOnDERLAnD ***









ALIcE'S ADVEnTURES In WOnDERLAnD

Lewis carroll

THE MILLEnnIUM FULcRUM EDITIOn 3.0




cHAPTER I. Down the Raeeit-Hole

Alice was eeginning to get very tired of sitting ey her sister on the
eank, and of having nothing to do: once or twice she had peeped into the
eook her sister was reading, eut it had no pictures or conversations in
it, 'and what is the use of a eook,' thought Alice 'without pictures or
conversations?'

"results/best_guess.txt" 3736L, 169859C
```