Assignment 1 - Cryptomatic

The primary goal of the “Cryptomatic” assignment was to implement the Jean-Taillléfer & Pierre-Wacozyk cryptosystem as well as to devise an algorithm for cracking the same cryptosystem using brute force. In the formal inputs declaration, it is stated that the encryption key will be a 16 bit, or 2 byte, value. With only 127 possible values for each byte, a brute force approach is very viable.

The basis of the algorithm is for every possible key, decrypt a portion of the text and determine if the decrypted text contains actual English words. To achieve this determination, a list of approximately 250,000 English words is loaded and hashed - called the dictionary. Because a sensical piece of text will contain spaces, the algorithm splits the decrypted text at each space character; each resulting string is then checked against the dictionary to see if it is an English word. If at least 50% of the strings are considered English words, the algorithm stores the key and the number of valid words it produced in a hash table. After all possible keys are exhausted, the algorithm finds the key that produced the highest number of valid words and returns it as the true key.

In the description above, it is stated that only a portion of the supplied message is used in the decryption. It is not viable to use the full message since the it could be extremely large. Instead, the algorithm uses the following rule: if the message is 200 characters or less, use the full message for the decryption; otherwise only 5% of the message will be used. These thresholds were determined empirically to provide the fastest and most reliable decryption. The final threshold used in the algorithm is the number of valid strings required to allow a key to be considered possible. Again, empirically a 50% threshold yielded the most reliable results. All three thresholds can be easily tweaked to increase the speed of the algorithm.

Another important aspect of the algorithm is the order in which key values are tested. Most things in Computer Science will start at 0, but this algorithm starts with the highest possible key value <127, 127> instead. The reasoning is that values below <32, 32> are not easily supplied using a standard keyboard, like the End of File character or the NULL character. Everything above <32, 32> are normal English characters like letters and punctuation. Since the algorithm is still checking every possible key value, this point is relatively moot; however, if the algorithm were to be tweaked to return the first “best guess” key, this would be an important distinction for saving some time.

One final important note regarding the algorithm concerns the dictionary. The dictionary selected was one found on preloaded on all UNIX systems (/usr/share/dict/words) containing roughly 250,000 entries. The words are hashed to allow constant time retrieval significantly boosting the speed of the algorithm. The dictionary used is important especially when dealing with particularly modern or particularly ancient texts since the English language has evolved so much. Especially new or old words may not appear in the dictionary allowing for true words to be flagged conversely. This was an substantial obstacle when attempting to brute force decrypt the Illiad while tweaking the threshold settings mentioned above.

As for the results, the algorithm works very well considering it always checks every key possibility. After encrypting the Gettysburg Address wit the key “#@“, the brute force algorithm was able to decrypt the text within 79 milliseconds. After encrypting “Clarissa, The History of a Young Lady”, one of the longest texts in the English language, with the same key, the algorithm

completed decryption after 5162 milliseconds. With the worst possible type-able key of <space, space>, the algorithm cracks the Illiad in 6421 milliseconds. Again, the algorithm could potentially run faster but the loss of reliability was deemed more important than saving only a few seconds.