Codes and Cryptography - Cryptography

1 Initial Ideas

1.1 Linear Cryptanalysis

Given that DES was used the first thing I reasearched was Linear Cryptanalysis. After reading [1] I had a very simplified conceptual understanding and so moved on to reading [2] and [3]. However, it very quickly became apparent that implementing linear cryptanalysis would likely require 2^{43} ciphertext, plaintext pairs [3], making it an approach that would be both hard to implement and unlikely to yeild a result on my laptop within the given time frame.

1.2 Dictionary Attack

The use of ECB combined with the structure of all 'what3words' addresses (3 words seperated by two ".") meant two things:

- That the number of candidate expressions was significantly reduced
- That the two blocks could be decoded independently, using the information in one block to inform the decryption of the other

To exploit the first of these points a dictionary containing a large quantity of the english language could be used, iterating over all possible combinations of words. The second point meant that having decoded the first block the possible starting characters of the second block were significantly narrowed down(the prefix of the second word having already been identified).

In addition to this, in experimenting with the provided executable, it became clear that only inputs that were a multiple of 8 would be accepted and meaning that no padding was used. This meant that the input plaintext must have been 16 bytes long. [4] also states that the minimum word length for a word in what3words is 4, therefore, there must always be two words of length 5 and one of length 4 in the plaintext.

2 Chosen Approach

Of the two I chose to implement the dictionary based attack. The first component of which was finding a suitable dictionary, for this I used words_alpha.txt from [5]. Using

this I could store and access every word by it's corresponding length. In addition to this I also computed every possible word prefix, for words of length 4 or 5, and stored these by size as well. With this I was then able to iterate over every possible first word, followed by a '.', combined with every prefix of length 8 - (length(first word) + 1). When a match did occur, I was then able to use this prefix to get the subset of all words that started with this prefix. The latter half of one of these words must have formed the first section of the second block. Iterating over these, followed by another dot and combined with every word of length 8 - (length(second word) - length(prefix) + 1) I was able to identify the second block. Combining the two decrypted blocks gave the full plaintext. For this this to work it relied on the assumption that length(first word) + 1 + length(second word) > 8. Given that every word is of length either 4 or 5 this always holds.

2.1 Number of Tests

In total there are 7,186 4 letter words and 15,918 5 letter words in the 'words_alpaha.txt' file. There are then also 26, 413 and 3,526 unique 1, 2 and 3 letter prefixes over all of these 4 and 5 letter words. In the worst case that the first word is 5 letters long this would mean checking all 4 letter words first, and for each of these every 3 letter prefix, $7186 * 3526 = 25,337,836$ cases. After this every 5 letter word would also have to be checked, combined with every 2 letter prefix, $15,918 * 413 = 6,574,134$ cases. For the second block this would require iterating over all 4 and 5 letter words that share a 2 letter prefix that was identified in the first block. Again assuming the worst case that the second word is 5 letters long and that the number of words with each prefix is roughly equal there are $7186/413 \approx 18$ four letter words with the prefix found and $15,918/3,2536 \approx 5$ five letter words with the prefix found, to check. This makes $18 * 7186 = 129,348$ cases followed by another $5 * 7186 = 35930$ cases, in total $286,524 + 35,930 = 322,454$ cases to check for the second block. Overall this is $25,337,836 + 6,574,134 + 322,454 = 32,234,424$ cases checked, significantly less than 2^{43}.

2.2 Time Taken

Using multiprocessing, modifying the exe and running on my 2017 surface book with Intel i7-6600U CPU with 8gb of RAM it took 50 minutes to complete.

2.3 The Result

The plaintext was found to be 'tile.bills.print' giving the location 7346 Melrose St, Philadelphia, PA 19136, USA, the back of the Four Seasons Total Landscaping parking lot, not the Four Seasons Hotel Philadelphia.

References

[1] "The Amazing King - Linear Cryptanalysis Tutorial", Theamazingking.com, 2021. [Online]. Available: http://theamazingking.com/crypto-linear.php. [Accessed: 15- Jan- 2021].

[2] Heys, Howard. (2001). A Tutorial on Linear and Differential Cryptanalysis. Cryptologia. 26. 10.1080/0161-110291890885.

[3] Junod, Pascal. (2011). Linear Cryptanalysis of DES.

[4] "What are the shortest and longest words used?", Support.what3words.com, 2021. [Online]. Available: https://support.what3words.com/en/articles/2212810-what-are-the-shortest-and-longest-words-used. [Accessed: 14- Jan- 2021].

[5] "dwyl/english-words", GitHub, 2021. [Online]. Available: https://github.com/dwyl/english-words. [Accessed: 19- Jan- 2021].