ANALYSIS OF THE ENCRYPTION ALGORITHM USED IN THE WORDPERFECT WORD PROCESSING PROGRAM

John Bennett

ADDRESS: Technology Systems, Inc. 81 Hartwell Avenue, Lexington MA 02173 USA.

ABSTRACT: WordPerfect V4.2, produced by WordPerfect Corporation, is a popular word processing program for the IBM PC series of microcomputers. The program includes an encryption option and, in their handbook, the manufacturers claim that "if you forget the password, there is absolutely no way to retrieve the document." The encryption is shown to be a simple affine Vigenère cipher with a significant weakness.

KEYWORDS: Cryptanalysis, WordPerfect, Vigenère

INTRODUCTION

An affiliated company of the author's uses the network version of WordPerfect Version 4.2 by WordPerfect Corporation. When files are used on a network they are available to a wide range of users and the file encryption facility of the WordPerfect program appeared attractive as a method of controlling access. Since a number of highly sensitive documents might be entrusted to the system the security provided by the encryption algorithm was investigated.

THE FILE FORMAT

On selection of the encryption option in WordPerfect (referred to as 'locking' in the documentation) it is necessary to enter a password (or key) of up to 75 characters which is not echoed on the screen. These characters may include spaces, numbers and other special, printable, characters such as the $. As a safety measure and to guard against mistyping, it is necessary to enter the same key a second time. Unfortunately, this tends to encourage users to enter short keys and, for the type of cipher chosen, the longer the key the better. One other weakness worth pointing out is that the user will be prompted to save the document again on exiting from the program, and, should he or she do so, the encrypted file will be overwritten by the original plaintext version. This only happens the first time the file is encrypted, subsequent editing of the document is properly protected.

The easiest situation in which to attempt a cryptanalytic attack is when the attacker can choose any plaintext and key and then see the results of the encryption. This is the situation that every user of WordPerfect is in. In order to make sense of the encrypted output, however,

the attacker will need to be able to read the ciphertext file. A short program in any computer language will permit the ciphertext to be displayed in a suitable form, such as hexadecimal; or the creative use of such utility programs as DEBUG, which is provided as part of the IBM PC DOS operating system package, will achieve the same end.

Having set up the environment in which both plaintext and ciphertext may be compared, the next step is to generate some samples. For the first attempt a number of single letter keys were selected and used to encrypt the following plaintext:

```
A A A A A A A A A A (nl)
B B B B B B B B B B (nl)
C C C C C C C C C C
```

The (nl), or newline, character is generated by the 'carriage return' or 'enter' key and is converted by WordPerfect to the line feed character with the ASCII value 0A hex. With the two carriage returns the length of the plaintext file is 32 characters while the ciphertext is 38 characters long. Changing the key does not alter this so it may be assumed that six characters are used by WordPerfect for a special purpose. The first six characters (in hexadecimal notation) of the ciphertext file for three different keys are given in Table 1.

FE	FF	61	61	41	00	Key = 'A' = 41
FE	FF	61	61	42	00	Key = 'B' = 42
FE	FF	61	61	43	00	Key = 'C' = 43

Table 1.

The sequence FE FF 61 61 is constant for all keys, whatever their length, and it may therefore be assumed that it is used by the program to determine whether the file is plaintext or ciphertext. The character in position 5 is the same as the single letter key and clearly does not provide much security. Table 2 gives the values of characters 5 and 6 for longer keys.

							Characters 5 & 6
Key	=	'AA'	=	41	41		61 80
Key	=	'AAA'	=	41	41	41	71 C0

Table 2.

Inspection of the result shows that characters 5 and 6 form a simple checksum of the key by taking the first character of the key and multiplying by 100 (base 16) to form a 16 bit word. If there is a second key character the initial result is circularly rotated right one binary position and then exclusive-ORed with the 16 bit word formed, as above, from the second character. This is repeated for all characters in the key. Using the key 'AAA' as an example:

First Key letter:	'A' = 41, multiplied by 100	= 41 00 =	01000001	00000000
	Rotate right one position	=	00100000	10000000
Second Key letter:	'A' = 41, multiplied by 100	= 41 00 =	01000000	00000000
	Exclusive OR	=	01100001	10000000
	Rotate right one position	=	00110000	11000000
Third Key letter:	'A' = 41, multiplied by 100	= 41 00 =	01000001	00000000
	Exclusive OR	=	01110001	11000000

The final result is equivalent to 71 C0 in hexadecimal.

The checksum is used by WordPerfect as a quick way of determining whether a valid password has been entered prior to decrypting the file. An error message is given to the user if an incorrect password is entered. Clearly the checksum itself cannot be used to determine the key except in the trivial case of very short key lengths. It should also be noted that the checksum is not a one to one transformation of the key. Keys 'DD' and 'HB', for example, both map to the checksum 66 00. Entering a password at random that passes the checksum test is fairly unlikely for longer keys and does not compromise the ciphertext. WordPerfect would, in this case, attempt to decrypt the ciphertext using the incorrect key and produce a yet further encrypted file. Depending on the action being taken this might result in the loss of the original file data but would not compromise its contents.

Plaintext:	41	41	41	41	41	41	41	41	41	41	0A
	42	42	42	42	42	42	42	42	42	42	0A
	43	43	43	43	43	43	43	43	43	43	
Key = 'A'	02	03	04	05	06	07	08	09	0A	0B	47
	0E	0D	0C	13	12	11	10	17	16	15	5C
	1A	1B	18	19	1E	1F	1C	1D	22	23	
Key = 'B'	01	00	07	06	05	04	0B	0A	09	08	44
	0D	0E	0F	10	11	12	13	14	15	16	5F
	19	18	1B	1A	1D	1C	1F	1E	21	20	
Key = 'AA'	03	04	05	06	07	08	09	0A	0B	0C	46
	0D	0C	13	12	11	10	17	16	15	14	53
	1B	18	19	1E	1F	1C	1D	22	23	20	
Key = 'AB'	03	07	05	05	07	0B	09	09	0B	0F	46
	0E	0C	10	12	12	10	14	16	16	14	50
	1B	1B	19	1D	1F	1F	1D	21	23	23	

Table 3.

THE ANALYSIS

Having determined the purpose of the additional six characters, it can be assumed that the other 32 characters in the test file are the ciphertext in a one to one mapping with the plaintext. Table 3 gives the plaintext and ciphertext for four keys in hexadecimal form.

From observation of the ciphertext the 'newline' character sticks out from the rest of the characters, indicating that a transposition cipher is not used. The ciphertext for the key 'A' shows that the encryption algorithm is not a simple exclusive-OR of the key with the plaintext. However, it is almost that simple as the first few characters for key 'A' indicate. If the ciphertext is exclusive-ORed with an ascending sequence 02 03 04 05 ... FC FD FE FF 00 ... and then exclusive-ORed with the key, the original text is restored. This would have been seen more easily if the original plaintext had been a repetition of a single character rather than the plaintext chosen. Table 4 gives the result of the exclusive-OR of the ciphertext for key 'A' with the ascending sequence.

00	00	00	00	00	00	00	00	00	00	4B
03	03	03	03	03	03	03	03	03	03	4B
02	02	02	02	02	02	02	02	02	02	

Table 4.

Examination of the ciphertext for key 'AA' shows that the ascending sequence starts at 03. Similar checks on keys 'AAA' and 'AAAA' reveal that the ascending sequence starts with the number equal to keylength + 1.

If the ascending sequence is removed from the ciphertext for key 'AB' by the exclusive-OR operation the result for the first ten characters is:

00 03 00 03 00 03 00 03 00 03

If the first character is exclusive-ORed with the first character of the key and the second character exclusive-ORed with the second character of the key, the first two characters of the plaintext are restored (00 xor 'A' = 'A', 03 xor 'B' = 'A'). By repeating this process for each of the subsequent groups of two characters (or, in general, for keylength groups) the whole of the plaintext will be decrypted. This type of encryption is the Vigenère cipher.

DECRYPTING

Having determined the cipher used, how can the file be decrypted? Konheim [1] describes methods for determining the keylength (k) and then splitting the ciphertext into k smaller subtexts created by sampling the ciphertext at intervals of k characters. Each of these subtexts is encrypted by a single character which can be found relatively simply, assuming the original plaintext is normal English, by searching for the most commonly occurring character and assuming that it is the letter 'E'. If the original plaintext is short and the keylength long, then the resulting subtexts may be too short to permit the key to be obtained. The WordPerfect encryption algorithm has, however, a fatal weakness which permits the recovery of the key for even quite short plaintexts. The weakness is that the commonly occurring 'space' character is not treated specially. Since this character occurs approximately once in every five characters for a wide variety of languages and other text sources it is trivial to simply try all possible

keylengths, testing for the space character in each subtext, and then verifying the resulting key against the checksum stored in the file. Using the methods outlined above a short Pascal program was written using the popular Turbo Pascal by Borland International. When run on an IBM AT computer, a key of 20 characters in length can be found in less than 20 seconds when decrypting a document of about 1500 words.

Copies of the program may be obtained by sending a disk to the author (IBM PC/AT format only) with a note explaining your interest.

CONCLUSION

The encryption system used in the WordPerfect program (version 4.2) is unsatisfactory for protecting sensitive documents. The manufacturer's literature should indicate more accurately the level of protection afforded.

REFERENCES

1. Konheim, A. G. 1981. *Cryptography: A Primer.* New York: John Wiley and Sons.

BIOGRAPHICAL SKETCH

John Bennett has an honors degree in electrical and electronic engineering from the University of Bath. He is currently working as Vice President of Marketing for Technology Systems, Inc., a software consulting company. His interests include microcomputers, cryptology, and private flying.