Design and Strength of a Feasible Electronic Ciphermachine from the 1970s

Jaap van Tuyll
Retired cryptanalyst
The Netherlands
jaapvantuyll@gmail.com

Abstract

This paper explores the design and strength of a feasible electronic ciphermachine that might have been invented in the 1970's. The design uses linear feedback shift registers, to get a key generator that satisfies some necessary requirements for a secure cryptographic algorithm. The analysis of the strength however, shows that the algorithm can be attacked successfully.

1 Introduction

Around 1970 a number of new off-line commercial cryptographic machines for the encryption of messages in telexcode, appeared on the market. Some examples are the H460 produced by Crypto AG, the TC803 produced by Gretag, the DC105 and DC26 produced by Datotek, the T1000CA produced by Siemens and the TST9669 produced by Telesecurity Timmann. These machines have in common, that they make use of electronic components, such as logical gates and shift registers. Thereby they mark a new generation of cryptographic machines, when compared with the older electromechanical machines such as the CX52. Not much is known in the open literature about the design of these machines. Therefore it might be interesting to explore and evaluate one of the possible ideas to design such a cryptographic machine, using electronic components.

2 General design criteria

Around 1970 there were several properties of a key generator known, to which a designer should pay attention. All of these have to be addressed by the designer of a cryptographic machine to make sure that the design will not a priori be a weak one.

One aspect is the need of a large key space for the initialisation keys. If the number of different initialisation keys is relatively small, it is possible for a cryptanalyst to test all the different keys (an exhaustive search) in a relatively short time.

A key generator without input is by its nature periodic. If it contains n memory elements of one bit, then the number of different states of the key generator is at most 2^n. Therefore the period can also not be larger than this number. The key generator should be designed in such a way, that the periods that are attained, are in the order of magnitude of this number. A very short period produces the possibility of reading in depth of a message shifted against itself. A somewhat longer period gives a cryptanalyst the possibility of testing all the positions of the period.

Different messages should use different parts of the period of the key generator to prevent reading in depth of the messages. This could be achieved by changing the complete initialisation key for the machine, but for many reasons this is not a practical solution. Mostly this is achieved, by changing a small part of the initialisation key (the message key) from message to message.

Then the key numbers that are used should have a flat distribution. If this distribution is skewed, it might be possible for a cryptanalyst to guess (part of) the plaintext.

Lastly there is the special off-line problem:

For an off-line machine using the telex alphabet the following problem has to be solved. If the keystream is randomly generated, it contains all 32 five bit combinations, so the result of the encryption of a plain character can be any of the 32 combinations. However, only the 26 printable combinations are acceptable as a cipher character. This problem can be solved in several ways. One of them is the following one: Generate key numbers between 0 and 31, use as encryption a modulo 32 addition of plain and key numbers, and in case

Proceedings of the 1st Conference on Historical Cryptology, pages 153– 158,
Uppsala, Sweden, 18-20 June, 2018

the result is a non-letter, repeat the encryption with the same key, until an acceptable result has been achieved.

2.1 Properties of shift registers

A linear feedback shift register (LFSR) is an important building block for electronic cryptographic machines. The reason is that an LFSR can produce a pseudo-random sequence of bits. This is a sequence with properties, resembling those of a real random sequence. Such a sequence is also called a maximum-length sequence, because the length of the period is maximal for a linear shift register (i.e. $2^n - 1$ for a shift register of length n). If this number is a prime, then each irreducible polynomial of degree n is associated with a shift register producing a maximum-length sequence. Moreover it is the polynomial with the lowest degree associated to that shift register, the so-called minimal polynomial.

The length of the period is a property that makes maximum-length sequences important for cryptographic purposes. Another property which is important is that the distribution of bit-combinations upto length n is flat.

For practical purposes it is convenient to use feedbacks that only have two connections. The polynomial then becomes a trinomial. In that case one XOR gate is sufficient to determine the feedback.

An important property of a binary LFSR is the following: Every bit in the output sequence of a binary LFSR is a linear combination of the bits of the initial content. A consequence is that as soon as n independent bits of the output sequence are known, it is possible to solve the initial content of a shift register of length n by solving a set of n linear equations.

More information on shift registers can be found in (Golomb, 1967).

3 Design

The goal is an off-line cryptographic machine that encrypts plaintexts, containing letters and the space character. The ciphertext should consist of letters only, so that a ciphertext could be printed before transmisson.

3.1 The plaintext alphabet

Because the space character is wanted in the plain alphabet, one letter (called the word separator) must be left out of the plaintext alphabet. (If not, the plain alphabet would have more elements than the cipher alphabet which is unacceptable, because that prevents unique decipherment). The space will then be replaced by the word separator during the encipherment. The consequence is, that the word separator itself cannot be used in the text of a plain message. If the resulting letter after decryption is the word separator, a space will be printed. That implies that it becomes invisible which is undesirable. A reasonable choice for a word separator could be the X and if the X itself is needed in a plaintext, then a trick must be used, e.g. representing an X by KS.

3.2 The key generator

To encrypt five-bit telex characters it is necessary to produce five-bit key characters. Therefore it seems plausible to use five binary LFSRs. Each register can then produce one of the required keybits. For convenience they can all have the same length n and the same feedback trinomial polynomial

$$X^n + X^f + 1$$

This must be an irreducible polynomial producing a maximum-length sequence of period $2^n - 1$. The required keybits can then be read off from the sections k of the registers. Because the shift registers all produce pseudo-random sequences, the produced key characters also are pseudo-random. See figure 1 for a picture.

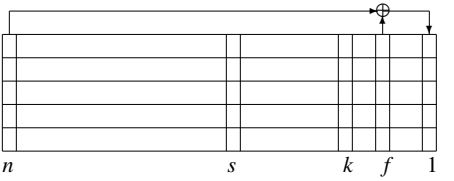

Figure 1: Shift registers

3.3 Initialising the machine

The registers could be initialised by $n-1$ key characters. One of the sections of each register (e.g. the first) should be set to 1 to ensure that no register starts in the all-zero state. The five bits of a key letter could then be read into the registers 1 through 5. As the initialisation of the machine needs to be different for every message, two keys should be used: a long-term key of $n-m-1$ letters

and a message key, unique for one message, of m letters. In practice it might be necessary to send the message key in the clear with the message, so the message key should not be used unmodified. This could be achieved by using (part of) the long-term key to change the bits of the message key, before they are entered into the machine.

The key space of this machine has the size $2^{5(n-1)}$.

3.4 Character encipherment

To encipher a plain letter, a key number K could be determined by reading off the contents of sections k of all the registers.

$$K = \sum_{i=1}^{5} R_{i,k} * 2^{i-1}$$

(where $R_{i,j}$ is the content of section j of register i).

The plain letter could be converted to a number P using its ITA2 bit value. For the word separator representing the space, the value $P = 4$ could be used.

The formula for the encipherment could be:

$$C = P - K \ mod \ 32$$

where the number C is converted through the ITA2 table into a cipher letter. If C is equal to $0, 2, 8, 27, 31$ or the number corresponding to the letter used as the word separator, it cannot be used in a ciphertext, because the corresponding character cannot be printed. (Such a character is considered illegal for this machine). In such a case the encipherment could be repeated with the same keynumber, until an acceptable result has been reached. It is easy to verify that this procedure always stops.

3.5 Character decipherment

Character decipherment is the inverse operation of the character encipherment. To decipher a cipher letter, a key number K is determined as described above, by reading off the contents of sections k of all the registers.

$$K = \sum_{i=1}^{5} R_{i,k} * 2^{i-1}$$

The cipher letter is converted to a number C using its ITA2 bit value.

The formula for the decipherment is:

$$P = C + K \ mod \ 32$$

where the number P is converted through the ITA2 table into a plain letter. If P is equal to $0, 2, 8, 27, 31$ or the number corresponding to the letter used as the word separator, it cannot be used in a plaintext, because the corresponding character cannot be printed. In such a case the decipherment is repeated with the same keynumber, until an acceptable result has been reached. If the plain number equals 4, the plain character is a space.

3.6 Stepping of the registers

After the encryption of a letter has ended successfully, the registers should step a number of times. To ensure a high period for the keystream generated by the machine, at most one register can have a constant step, so e.g. register 1 steps a constant number of steps. Then the higher numbered registers should make a variable number of steps, e.g. the number of steps could be determined by the XOR of the content of the sections s of the lower numbered registers.

$$S_i = \sum_{j=1}^{i-1} R_{j,s} \ mod \ 2$$

where S_i determines the number of steps of register i for $i > 1$.

In this way it is guaranteed, that whenever one register completes its period, all the other registers do not. This ensures that the period of the keystream of the machine is in the order of magnitude of $(2^n)^5$.

3.7 Summary of the design

The design of the machine meets the necessary criteria for a secure algorithm that were mentioned earlier:

The keyspace has a large size.

The keystream has a large period.

The keycharacters generated have a flat distribution.

Different messages use different parts of the keystream.

4 Strength

When attacking the feasible cryptographic machine described above, it is reasonable to assume that the details of the design are known to an attacker. This approach is fully in line with the second Kerckhoffs' principle that was stated by Auguste Kerckhoffs in 1883 in (Kerckhoffs, 1883).

It is clear that the variable stepping of the higher numbered registers implies that the start of an attempt to break this feasible machine, can only be an attack on the first register, the only one that has a constant step. After solving the first register the stepping pattern of the second one is known and

then the second register could be attacked and so on.

4.1 Fast correlation attack

The fast correlation attack is a procedure to determine the initial content of a shift register if only a part of the output sequence containing some errors is known.

This attack has been presented for the first time at Eurocrypt 1988 in Davos by Willi Meier and Othmar Staffelbach and has been published in (Meier and Staffelbach, 1989), although the attack was already known many years before.

The method works as follows:

The enciphering method modifies the bits of the register that will be attacked. This modification is not random, but biased to either 0 or 1. The part of the shift register sequence for which this biased information is available is initialised with values coming from the bias.

The feedback polynomial gives a relation that is satisfied by all the bits in the output sequence of the shift register. Moreover all the multiples and in particular all the 2^n powers of this polynomial (which are all also trinomials) give relations that are satisfied.

As an example if $X^n + X^f + 1$ is the feedback polynomial, then if b_i is the i-th bit of the output sequence, $b_{i+n} + b_{i+n-f} + b_i = 0 \ mod\ 2$ is satisfied for all $i > 0$. But also $b_{i+2n} + b_{i+2(n-f)} + b_i = 0 \ mod\ 2$ and so on. Moreover one can add $b_{i+n+f} + b_{i+n} + b_{i+f} = 0 \ mod\ 2$ to $b_{i+n} + b_{i+n-f} + b_i = 0 \ mod\ 2$ and in this way get the new relation $b_{i+n+f} + b_{i+n-f} + b_{i+f} + b_i = 0 \ mod\ 2$. In this way many trinomial and tetranomial relations can be found which are satisfied by the bits of the produced output sequence.

Step 1 is: Count for every position the number of relations involving that position that are satisfied and the number of relations that are not satisfied. If the number of satisfied relations is much higher, then the bit at that position is probably reliable and nothing is changed. In the other case the bit is probably unreliable and it is declared unknown.

Step 2 is: Compute for all the unknown bits a new value on the basis of the reliable bits only. For many unreliable bits this will succeed and then a new sequence has been determined.

The counting and correcting steps are repeated until enough highly reliable bits have been found. Then the initial content is solved by solving the set of equations, generated by those reliable bits. Under certain conditions on the values of the bias and the length of the used part of the shift register sequence this method converges.

4.2 Correlation attack on register 1

For every letter of the ciphertext the 32 possible decryptions can be divided into two groups, one corresponding with an even key number and the other one corresponding with an odd key number. Depending on the letter frequencies in the language of the plaintext, the two groups have a different probability of occurrence. The consequence is that the probability of the last bit of the key number being 0, is different from the probability of the last bit of the key number being 1. In other words for every cipher letter there is a bias to either 0 or 1 for the last bit (K_1) of the key number K.

Compare

$$P(K_1 = 0 \mid cipherletter = C)$$
with
$$P(K_1 = 1 \mid cipherletter = C)$$

An example: For cipher letter H the set of decryptions with an even keynumber is:

{B,G,H,L,M,O,P,Q,T,V,W,Y,Z}

The letters L, W and Z occur twice as a decryption with different keynumbers.

For odd keynumbers the set is:

{A,C,D,E,F,G,I,J,K,N,R,S,U,X}

Here F and U occur twice as a decryption with different keynumbers.

In most languages the second set is much more frequent than the first one, especially because the X in the second set is the word separator and thus represents the space character. So for cipher letter H the bias is for a 1 as the last keybit.

This bias makes it possible to determine the initial shift register sequence. First the bias is used to make an approximation of the shift register sequence produced by the first register of the machine.

If the language statistics are favourable enough and the message length is sufficiently large, the fast correlation attack reconstructs the initial content of the first register of the machine.

4.3 Correlation attack on register 2

A similar approach can be used for the higher numbered registers. Once register one is known

the stepping of register two is also known. Moreover for every cipher letter one keybit is known. In this case the decryptions of a given cipher letter are divided into four groups. In each group the key numbers have a fixed value modulo 4.

An example: For cipher letter H the four groups are:

key = 0 mod 4 {H,L,P,Q,T,W,Y,Z}
key = 2 mod 4 {B,G,L,M,O,V,W,Z}
key = 1 mod 4 {C,D,F,J,K,N,R,U}
key = 3 mod 4 {A,E,F,G,I,S,U,X}

Then depending on the value of the keybit produced by register one, either the groups with key numbers equal to 0 and 2 modulo 4 are compared or the groups with key numbers equal to 1 and 3 modulo 4 are compared.

Compare

$$P(K_2 = 0 \mid cipherletter = C \wedge K_1 = b)$$
with
$$P(K_2 = 1 \mid cipherletter = C \wedge K_1 = b)$$

where b is the known keybit from the first register.

For every ciphertext letter the bias between the groups that are compared is used to initialise an approximation for the second register. Once again if the conditions are favourable the fast correlation attack finds the initial content of register two.

4.4 Registers 3, 4 and 5

In principle the same method can be used for the registers 3, 4 and 5. However, in many languages the statistics are unfavourable for a successful attack on register 3.

One way to improve the statistics is by using bigrams of the ciphertext instead of single ciphertext letters. That improves the bias because in that case the frequencies of plain letter bigrams are used to calculate the bias.

Compare

$$P(K_3 = 0 \mid cipherpair = CC' \wedge K_1 = b_1 \wedge K_2 = b_2 \wedge K_1' = b_1' \wedge K_2' = b_2')$$
with
$$P(K_3 = 1 \mid cipherpair = CC' \wedge K_1 = b_1 \wedge K_2 = b_2 \wedge K_1' = b_1' \wedge K_2' = b_2')$$

where b and b' are the keys for the consecutive cipherletters C and C'.

4.5 Probable word method

There is however another method: the probable word method. If for every cipher letter the key bits of the registers 1 and 2 are known only 8 possible decryptions remain possible.

If the cipher letter is H and the keybits from the first and second register both are 1 then the plainletter must be one of the set {A,E,F,G,I,S,U,X}.

Now it is possible to test for every position in the ciphertext, whether a word that probably will be present in the plaintext, fits the ciphertext.

So you might get the following situation: (CT is the ciphertext letter and k_1 and k_2 are the known keybits from the registers 1 and 2).

CT	D	P	I	F	E	R	U	O	B	K
k_1	0	0	1	1	0	1	0	1	0	0
k_2	1	1	1	0	0	0	0	1	0	0
	S	V	V	**A**	E	G	U	N	B	K
	A	B	**B**	H	**S**	V	X	R	**O**	N
	U	T	P	Z	A	B	X	C	**O**	N
	C	L	R	O	U	X	S	**D**	G	**R**
	X	Z	D	M	E	I	I	F	**O**	C
	K	O	O	G	X	**S**	E	J	M	D
	I	**M**	M	V	A	O	S	K	G	F
	E	G	G	B	I	M	**A**	T	V	J

The boldface letters show that here the word **AMBASSADOR** is possible.

If the word has a sufficient length only correct positions are found. (Almost) every plaintext letter that is found in this way fixes a keybit in the registers 3, 4 and 5. In the example above that is the case for all ciphertext letters, except the ciphertext letter B, for which there are three possible decryptions as O.

As soon as enough independent keybits have been found, the initial content of the registers 3, 4 and 5 can be found by solving a set of linear equations for each register.

5 Conclusion

The cryptographic algorithm that has been described could very well have been invented in the 1970's, when new ideas in cryptography were generated by the new electronic components that became available. The paper shows that although the algorithm meets a number of design criteria, it still could have been broken using an idea that was published in 1989.

References

Solomon W. Golomb. 1967. *Shift Register Sequences*. Holden–Day.

Auguste Kerckhoffs. 1883. La cryptografie militaire. *Journal des Sciences Militaires*, IX:5–38, jan.

Willi Meier and Othmar Staffelbach. 1989. Fast Correlation Attacks on certain Stream Ciphers. *Journal of Cryptology*, 1(3):159–176.