ENCRYPTION SCHEMES
FOR
COMPUTER CONFIDENTIALITY

Vera Pless

May 1975
Encryption Schemes for Computer Confidentiality

I. Introduction

With the ever-increasing amount of data stored on computers, the need for security in transmission and storage becomes greater and greater [2]. We here consider some new stream enciphering schemes based on J-K flip-flops. The data is considered to be a stream of binary bits. There are two main types of encipherment schemes; one is a block scheme which divides the data into blocks and then enciphers and deciphers a block at a time, the other is a stream scheme which enciphers and deciphers bit by bit. The stream enciphering scheme has the advantage that both the enciphering and the deciphering occur in real time. Since the aim of this paper is to present some new stream enciphering schemes, we shall describe briefly a general stream enciphering scheme.

If we let \( X \) denote the data set, i.e. \( X = (x_1, x_2, \ldots, ) \) and \( K \) denote a key which is a determined set of bits, \( K = (k_1, k_2, \ldots, ) \), then the enciphered message \( Y = (y_1, y_2, \ldots, ) \) is equal to \( X + K = (x_1 + k_1, x_2 + k_2, \ldots, ) \) where \( x_i + k_i \) is computed mod 2. Deciphering is very simply accomplished by adding the key \( K \) to the enciphered message \( Y \) obtaining \( X \) as \( Y + K \). So we see that the important item in an enciphering scheme is the key \( K \). It is assumed that an unauthorized person knows \( Y \) and a portion of clear text (that is a number of bits of
X) and so can determine the same number of bits of K. The problem is to prevent the unauthorized user from being able to determine all of K (hence all of X).

Clearly this cannot happen when K is a random sequence of bits. Such a key, however, has the disadvantage that it can only be used once. To overcome this disadvantage and preserve the features of randomness, people have generated pseudo-random sequences or sequences of very long period. It is possible to generate a sequence of period \(2^r - 1\) with an r-stage linear shift register [3]. Hence the particular shift register plus an initializing vector which is placed in the shift register at the start forms the key in this particular case. However, as was shown in [5], the linearity of the system enables one to solve for both the shift register and the initializing vectors with about \(2^r\) bits of clear text. For a general introduction to cryptography see [8].
II. Non-Linear Schemes Using J-K Flip-Flops

The enciphering schemes we propose below preserve the pseudo-randomness properties of the shift register while removing the weakness due to linearity. This is accomplished by combining shift registers with J-K flip-flops. So we will first define a linear n-stage shift register with feedback.

The figure above is a diagram of an n-stage linear shift register with feedback, for brevity we will just call this a shift register for the rest of this paper. Each of the squares labelled $x_1$, $x_2$, ..., $x_n$ contains either a 0 or a 1.

At periodic intervals, the contents of $x_i$, $i > 1$, are transferred into $x_{i-1}$ and the contents of $x_1$ go out. The new content of $x_n = \sum_{i=1}^{n-1} c_i x_i$ where the $c_i$ are all specified, each is 0 or 1, and the addition is modulo 2. The word linear comes from this expression. If an initializing vector of $n$ 0's and 1's is put into positions $x_1$, ..., $x_n$,
then the shift register generates a sequence of 0's and 1's. The longest period of this sequence is called the period of the shift register. It is not hard to show [3] that the longest period which an r-stage shift register can achieve is \((2^r - 1)\). Further, if the characteristic polynomial (which determines the \(c_j\) of the r-stage shift register divides \((x^{(2^r - 1)} - 1)\) over GF(2), but no \((x^s - 1)\) for \(s < (2^r - 1)\), then its shift register has period \((2^r - 1)\). These sequences of length \((2^r - 1)\) are called maximum-length shift register sequences. Even though no finite sequence is truly random, certain properties are associated with random sequences. In [3] it is shown that maximal length shift register sequences satisfy three natural randomness properties. In our cycling schemes we will be using maximum length shift register sequences of large period.

Another device we must explain is a J-K flip-flop. This is a 2 input, 2 output (where one output is the complement of the other) device which operates according to the following rules. We consider an ordered pair to represent the inputs \((j, k)\). An input \((0, 0)\) leaves the output unchanged, a \((1, 1)\) input changes the output, a \((0, 1)\) input produces a 0 output, and a \((1, 0)\) input produces a 1 output. An important fact for the successful operation of the encryption schemes proposal is given as follows. Let \(R_N\) denote the Nth output and \((j, k)\) denote the Nth input. Then

\[
R_{N+1} = (j + k + 1)R_N + j.
\]
This can be demonstrated by direct computation [7].

Hence two consecutive outputs are needed to compute one of \( j \) or \( k \), which one cannot be specified, but if two consecutive outputs are known, one of these is known also.

Before we propose some encryption schemes we consider the following arrangement in order to analyze its strengths and weaknesses.

![Diagram](image)

In this arrangement, there are two shift registers denoted by \( \text{S.R.1} \) and \( \text{S.R.2} \) whose outputs constitute the \( j \) and \( k \) inputs to a J-K flip-flop. We here consider \( \text{S.R.1} \) and \( \text{S.R.2} \) to generate maximum length shift register sequences. This whole arrangement is considered as generating a key sequence \( K \) which it can do once initializing vectors are input into \( \text{S.R.1} \) and \( \text{S.R.2} \). Clearly changing either \( \text{S.R.1} \) or \( \text{S.R.2} \) results in a new key.

Remark 1: Notice as a consequence of \( \text{A} \) that two outputs of \( \text{A} \) are needed in order to determine either \( j \) or \( k \). Specifically, an output of
01 (first 0, then 1) specifies j as 1, 00 implies j = 0, 10 (first 1, 
then 0) specifies k as 1 and 11 implies k = 0. From this it follows 
that a set of s consecutive j's can only be determined by a set of s + 
1 outputs of A whose first s elements are 0 and that the set of s j's 
are all 0. Similarly, a set of s consecutive k's can only be 
determined by a set of (s + 1) outputs of A whose first s elements are 
1 and the set of s k's must be all 0.

First we discuss the strengths of this scheme.

1) Even if the periods of S.R.1 and S.R.2 are of moderate sizes, it is 
possible to choose them so that the period of A is much larger. This 
is expressed precisely in the following theorem.

Theorem: If S.R.1 has period p₁ \( \neq 1 \), S.R.2 has period p₂ = 1, 
g.c.d. (p₁, p₂) = 1, and p₁ and p₂ are odd, then A has period p₁p₂.

Proof: Denote by s the period of A. Note that s cannot be 1. After 
the initialized conditions have been overcome, the output of A must 
repeat at p₁p₂ since g.c.d. (p₁, p₂) = 1. Hence s \| p₁p₂. Since 
g.c.d. (p₁, p₂) = 1, both p₁ and p₂ divide s so that s is p₁p₂.

The periods of irreducible polynomials of degree 12 through 20, 
where the degree gives the stage of the shift register, is given in
Table I. From this it can be seen that there are \((144) \times (630)\) ways to choose a 12-stage shift register of period 4,095, and a 13-stage shift register of period 8,191. These periods satisfy the conditions of the theorem so that \(A\) would have period greater than \(10^7\). Similarly there are 27,594 \(\times\) 24,000 choices for a 19-stage shift register of period 524,287 and a 20-stage shift register of period 1,048,575 where \(A\) for this situation would have period greater than \(5 \times 10^{11}\).

2) \(A\) is very easy to implement since both shift registers and J-K flip-flops are easy to implement.

3) System \(A\) has good features in case an error occurs in the transmission of \(K\). If the error is in a bit which has emerged from the flip-flop then it is a single error which does not affect any other bits. If an error occurs in the internal state of the flip-flop then it affects all bits as long as \((j,k)\) is either \((0,0)\) or \((1,1)\). However, the error is corrected as soon as \((j,k)\) is either \((0,1)\) or \((1,0)\) so that we either have a completely incorrect stream which could be easily detected or a completely correct stream.

Remark 2: Assume S.R.i \((i = 1, 2)\) has \(r_i\) stages. The largest sequence of consecutive digits which the output of \(A\) can determine for S.R.i has \((r_i - 1)\) consecutive zeroes.

Proof: By the randomness properties of maximum-length shift registers,
the largest sequence of zeroes generated by S.R.1 or S.R.2 has length 
(r₁⁻¹) or (r₂⁻¹) respectively [3]. The proof of Remark 2 then 
follows from Remark 1.

The following is the most serious weakness. From the previous 
discussion of shift registers we know that a particular r-stage shift 
register can be determined by knowing 2r bits of clear text; r bits for 
the initializing vectors and r bits to solve the r linear equations. By 
remark 2, the largest sequence of consecutive bits for S.R.1 which can 
be determined by the output of A has length (r₁⁻¹). In the following 
very unlikely situation S.R.1 and S.R.2 can be determined by 2r₁ + 2r₂ 
bits of clear text. We assume the values of r₁ and r₂ are known. 
Suppose the output of A has (r₁⁻¹) zero bits. Then S.R.1 has a 
sequence of (r₁⁻¹) zeroes and so must have a one at each end of this 
sequence yielding r₁ + 1 known bits. If the next (or preceding) (r₂⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻⁻ßen

Another weakness is that the randomness properties of the maximum 
length shift register sequences are not preserved in A. Namely if a 
one is output the likelihood is less than one half that the next output
will be a one. Similarly for a zero output. This is undesirable because it is more susceptible to a statistical attack than a random sequence. For these reasons we consider a modification of arrangement A, namely arrangement B.

![Diagram]

This is an arrangement A except for the alternator (denoted by A) after the flip-flop. The alternator eliminates alternate bits.

Remark 3. If S.R.1 has odd period $p_1$, S.R.2 has odd period $p_2$, and g.c.d. $(p_1, p_2) = 1$, then arrangement B has period $p_1 p_2$.

Proof: By the theorem the output of the flip-flop has period $p_1 p_2$ which is an odd number. Hence the sequence formed of every other bit of the output sequence has the same period $p_1 p_2$.

Note that the alternator restores some of the randomness properties of the maximum length shift registers since two is relatively prime to $p_1 p_2$ when $p_1 p_2$ is odd.
Since the output of arrangement B cannot determine any digits of either S.R.1 or S.R.2 by Remark 1, we could attempt to reconstruct the key by guessing the \((2r_1 + 2r_2)/2 = (r_1 + r_2)\) missing alternate bits. This requires \(2^{r_f}\) guesses and so represents a great deal more computation needed to reconstruct the key than for arrangement A. Arrangement B has the disadvantage that it emits one digit for every 2 cycles of the clock. The shift register must operate at twice the input stream rate.

III. Some proposed encryption schemes.

The first proposed scheme is arrangement C below.

The 3 A's again denote alternators.

Assume S.R.1 has \(r_1\) stages. Then to reconstruct the key one must guess \(2^{(2^{r_2} + 2^{r_4})}\) alternate bits at least and this is too much to
calculate for \( r_1, r_2, r_3, \) and \( r_4 \) of even moderate sizes. Arrangement C is straightforward to simulate on a computer. It also has the disadvantage that it does not run in real time, however it is easy to implement.

The second proposed scheme D combines all the unbreakable properties of arrangement C with the advantage that it does run in real time.
This is also unbreakable by linear calculation, and runs in real
time. The final device with boxes labelled from 1 until 4 is a
recycling counter and transmits the contents of $i + 1 \pmod{4}$ right
after the contents of box i is transmitted. If the output of S.R.1 is
an odd number $p_i$ and the $p_i$'s are relatively prime in pairs, then the
output of D is $\prod_{i=1}^{8} p_i$. Since four is relatively prime to $\prod_{i=1}^{8} p_i$ when the
$p_i$ are odd the output maintains some of the randomness properties of
the original shift register sequences. It is an open question whether
the output is a pseudo-random sequence in the sense of [3].

The following is one possible way to choose S.R.i, $(i = 1, \ldots, 8)$
in arrangement D. From Table I we see that we can choose the 8
polynomials as follows.

<table>
<thead>
<tr>
<th>$C_1$ S.R.1</th>
<th>$C_2$</th>
<th>$C_3$</th>
<th>$C_4$</th>
<th>$C_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>r_i = stage</td>
<td>period = $(2^i - 1)$</td>
<td># of choices for S.R.1</td>
<td>factorization of $C_3$</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>31</td>
<td>6</td>
<td>31</td>
</tr>
<tr>
<td>2</td>
<td>19</td>
<td>524,287</td>
<td>27,594</td>
<td>524,287</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>127</td>
<td>18</td>
<td>127</td>
</tr>
<tr>
<td>4</td>
<td>17</td>
<td>131,071</td>
<td>7,710</td>
<td>131,071</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>511</td>
<td>48</td>
<td>7x73</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>-----</td>
<td>----</td>
<td>------</td>
</tr>
<tr>
<td>6</td>
<td>16</td>
<td>65,535</td>
<td>2,048</td>
<td>3517257</td>
</tr>
<tr>
<td>7</td>
<td>11</td>
<td>2,047</td>
<td>176</td>
<td>23x89</td>
</tr>
<tr>
<td>8</td>
<td>13</td>
<td>8,191</td>
<td>630</td>
<td>8,191</td>
</tr>
</tbody>
</table>

Note from $C_5$ that the periods of the eight shift registers are odd and relatively prime in pairs so that the final period is their product which is greater than $10^{28}$. The number of different choices of these periods for the eight shift registers is given by the product of the numbers in column $C_4$ which is greater than two times $10^{20}$. This is a number so large that even if a precise circuit and all the keys are given to an unauthorized person there is no possibility of successfully breaking a message by simply trying all the keys. Note that is is necessary to store less than 40,000 polynomials to obtain these more than $10^{20}$ choices.

Table I below is given to illustrate the large number of irreducible polynomials which are available to generate maximal length shift sequences and where to find some of them.

A gives the degree of the polynomial = stage of shift register
B = period of a maximal length shift register of degree A.

C = number of irreducible polynomials of degree A and period B. This is given by the formula \( \phi(2^A - 1) / A \) where \( \phi \) is the Euler-\( \phi \) function [3].

D = P means all C polynomials of degree A and period B can be found from the tables in the back of Peterson and Weldon [6].

E = factorization of B [computed on Macsyma[4]].

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>31</td>
<td>6</td>
<td>P</td>
<td>prime</td>
</tr>
<tr>
<td>6</td>
<td>63</td>
<td>6</td>
<td>P</td>
<td>( 3^2 \cdot 7 )</td>
</tr>
<tr>
<td>7</td>
<td>127</td>
<td>18</td>
<td>P</td>
<td>prime</td>
</tr>
<tr>
<td>8</td>
<td>255</td>
<td>16</td>
<td>P</td>
<td>( 3 \cdot 5 \cdot 17 )</td>
</tr>
<tr>
<td>9</td>
<td>511</td>
<td>48</td>
<td>P</td>
<td>( 7 \cdot 73 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1023</td>
<td>60</td>
<td>P</td>
<td>3·11·31</td>
</tr>
<tr>
<td>11</td>
<td>2047</td>
<td>176</td>
<td>P</td>
<td>23·89</td>
</tr>
<tr>
<td>12</td>
<td>4,095</td>
<td>144</td>
<td>P</td>
<td>3^2·5·7·13</td>
</tr>
<tr>
<td>13</td>
<td>8,191</td>
<td>630</td>
<td>P</td>
<td>prime</td>
</tr>
<tr>
<td>14</td>
<td>16,383</td>
<td>756</td>
<td>P</td>
<td>3·43·127</td>
</tr>
<tr>
<td>15</td>
<td>32,767</td>
<td>1,800</td>
<td>P</td>
<td>7·31·151</td>
</tr>
<tr>
<td>16</td>
<td>65,535</td>
<td>2,048</td>
<td>P</td>
<td>3·5·17·257</td>
</tr>
<tr>
<td>17</td>
<td>131,071</td>
<td>7,710</td>
<td>prime</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>262,143</td>
<td>7,776</td>
<td>3^3·7·19·73</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>524,287</td>
<td>27,594</td>
<td>prime</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>1,048,575</td>
<td>24,000</td>
<td>3·5^3·11·31·41</td>
<td></td>
</tr>
</tbody>
</table>

We can see from Table I that there is a very large number of
keys of relatively prime lengths available. These yield an output of period so long that it is difficult to break. Just this short table is enormously rich in keys and periods and yet a scheme using them has about 100 components.

Another variation on these encycling schemes would be to use an n-counter, which is a set of interconnected J-K flip-flops, instead of a shift register. The mathematical theory of these n-counters is presently being developed ([1] and [7]). The algebraic formulas for determining the output sequence is given in [7, p.9] There are both linear and non-linear n-counters and the non-linear ones would be more difficult to determine than a linear shift register. Using n-counters rather than shift registers would make the proposed schemes even more resistant to statistical attack. However, since the mathematical theory is so new, how to choose an n-counter with a very large period is still an open question.
IV. Conclusion

In conclusion, we have proposed some stream enciphering schemes which use standard components and are easy to implement. These schemes appear to be difficult to break and we have made estimates in some instances of how difficult this is. These estimates have shown that these schemes require more computations than can economically be performed. We believe that these schemes would perform very well as data encryption schemes for computer confidentiality.

Acknowledgements

I would like to acknowledge stimulating discussions with Edward Fredkin on these topics.
Bibliography


