A talk on The Oppel-1 Block Cipher University of Science and Technology Beijing, China

Dr. Arshad Ali

CESAT, Islamabad, Pakistan
Arshad.Ali.2008@live.rhul.ac.uk

June 28, 2018

References

Arshad Ali, Oppel-1: A New Block Cipher. In Proceedings of the 14th IBCAST 2017, pages 441–447

Sequence

Cyberspace?

Definition

According to NISTIR 7298 Revision 2[SOURCE: CNSSI-4009]

“A global domain within the information environment consisting of the interdependent network of information systems infrastructures including the Internet, telecommunications networks, computer systems, and embedded processes and controllers”.

Introduction

- In "Information Systems", the information is safeguarded by using cryptographic primitives.
- Block Ciphers and Stream Ciphers are two important types of Cryptographic Primitives.
- Oppel-1 is a block cipher published in the Proceedings of IBCAST 2017

Introduction

- The cipher known as one-time-pad (OTP)
    - is the unique type of cryptographic system that can provide perfect secrecy.
- Since this cipher uses
    - non-repetitive,
    - true random sequence

  as keystream,
- therefore, there is an implementation problem associated with OTP-cipher,

  which is referred to as key distribution problem.
- This problem limits the wide spread use of OTP cipher in practical applications.

Introduction

- As a result of this situation,
    - ciphers which provide time-based security known as computational security
    - are commonly used to provide various security services
    - for information traversing through various information systems infrastructures of the Cyberspace.

Reasons for New Designs?

- The design and analysis of non-OTP ciphers is very active research area now days.
- There are number of reasons which prompt cryptographers to design new non-OTP ciphers.
- Two of these reasons are discussed here:

Reason-1

- Since security of non-OTP cryptographic schemes relies heavily on prevailing computational and mathematical techniques
- Therefore, advancement in these areas of knowledge adversely affects the security of non-OTP ciphers and makes them prone to aging affects
- Recent advances in quantum computing can be considered as an example of this phenomena

Reason-2

- New technological advancements in the field of electronic communication systems result in the emergence of new cryptographic applications,
    - e.g. miniaturised mobile communication devices
- Since these new cryptographic applications have their own implementation requirements and constraints
- Therefore, this restricts the use of off-the-shelf available cryptographic primitives

Block vs Stream Ciphers

- Block ciphers (e.g. DES, AES, KASUMI, Skipjack, PRESENT, DESL, KATAN and KTANTAN)
    - are used to provide many cryptographic services
    - for information that is to be stored, processed or transmitted through electronic communication networks
- These types of ciphers operate on blocks of fixed lengths using time-invariant encryption transformations
- In contrast stream ciphers encrypt individual units of data using time-varying transformations.

Block Cipher Design Criteria?

- A block cipher is considered to be suitable for implementation in practical applications if its security and implementation properties are easy to understand and verify
- In this context, the security properties are aimed at providing suitable assurance that no attack better than exhaustive search exists for this cipher
- Moreover the gist of implementation properties is to provide assurance that it can be easily and efficiently implemented in certain implementation environments

Oppel-1

- Oppel-1 is a non-Fiestel cipher and is based on iterated substitution-permutation networks
- The claimed security for this block cipher is 2^{128} primitive operations of its encryption algorithm

Oppel-1: Design Features

Oppel-1 Subkey Generation

- The Oppel-1 subkey generation algorithm uses a combination of linear and nonlinear operations to produce subkeys
- These subkeys are used in various rounds of encryption and decryption algorithms

Oppel-1 Subkey Generation

Algorithm 1. Computation of Oppel-1 subkeys

```
1:  input    :=    InK, $r$
```

Oppel-1 Subkey Generation

Algorithm 1. Computation of Oppel-1 subkeys

```
1:  input   :=  InK, $r$
2:  output  :=  KEY$_i$,   $\forall i \in \{0, r-1\}$
```

Oppel-1 Subkey Generation

Algorithm 1. Computation of Oppel-1 subkeys

```
1:  input       :=  InK, $r$
2:  output      :=  KEY$_i$,    $\forall i \in \{0, r-1\}$
3:  initialise  :=  StK, $s_0, s_1, \ldots, s_7$; $S_0, S_1, \ldots, S_7$
```

Oppel-1 Subkey Generation

Algorithm 1. Computation of Oppel-1 subkeys

```
1:  input      :=  InK, $r$
2:  output     :=  $\mathrm{KEY}_i$,   $\forall i \in \{0, r-1\}$
3:  initialise :=  StK, $s_0, s_1, \ldots, s_7$; $S_0, S_1, \ldots, S_7$
4:  $K$ := InK $\oplus$ StK
5:  for $i := 0$ to $r-1$
6:          $x := K \lll 6$
7:          $k_0 := x \oplus s_{(i\%8)}$
8:          $k_0 := b_0||b_1||\cdots||b_{15}$
              where each $b_i$,  $i \in \{0, 15\}$represents a byte
9:          for $j := 0$ to 15
10:             $B_j := S_{(j\%8)}(b_j)$
11:         end for
12:         $\mathrm{KEY}_i := B_0||B_1||\cdots||B_{15}$
13:         $K$ := $\mathrm{KEY}_i$
14: end for
15: return $\mathrm{KEY}_i$,   $\forall i \in \{0, r-1\}$
```

Oppel-1 Subkey Generation

- Algorithm 1 also uses some pre-stored data entities
- These entities are denoted by $\mathrm{StK}, s_0, s_1, \ldots, s_7$ and $S_0, S_1, \ldots, S_7$
- StK is a 128 bit long bitstring

1. The parameter ‘StK’ can be generated using any pseudorandom number generation technique that satisfies following criteria:-
    1. Hamming distance between any two contiguous nonzero bits of StK should not exceed by 7
    2. The overall Hamming weight of StK must lie between 50 and 74
2. The variables $s_0, s_1, \ldots, s_7$ are used to denote eight bitstrings, each of length 128 bits
3. Sample bitstrings used in our experiments are given in Table 1
4. These bitstrings are based on prime numbers, denoted by q, such that 2 is a primitive element in the ring of units $\mathbb{Z}/(q)^*$

Oppel-1 Subkey Generation

Table: Bitstrings used in Oppel-1 subkey generation mechanism

Identifier	**Sample bitstrings each of length** 128 **bits**
s_0	B9C6 A9EA B7E2 5FD6 9E86 369A 1856 EC4A
s_1	AE98 5DFF 2661 9FC5 8623 DC8A AF46 D590
s_2	CB5E 129F AD4F 7E66 780C AA2E C8C9 CEDB
s_3	2102 F996 BAF0 8F39 EFB5 5A6E 3900 02C6
s_4	3DD4 254E EBDC FE2B FF0C D7F7 F7BE 2DC2
s_5	6BFA BF9F FFFE EFFD FECD 9DEF ED3D EDFA
s_6	BDFF 77FF FCFF BDF7 EFDF DFAF FF53 D9FF
s_7	FDFE BBFC FAFF FFDF 47D6 D7FF 7FBF E76E

1. The variables $S_0, S_1, \ldots, S_7$ represent eight S-boxes.
2. Each of these S-boxes can be considered as a 16×16 array.
3. These arrays consist of different permutations of numbers from 0 to 255.
4. These permutations are generated using the stream cipher known as RC4 implemented in a secure manner.
5. The procedure of generation of these S-boxes is summarised as follows:-
    1. Using arbitrary key clock RC4 2^{16} times without taking its output.
    2. Generate eight blocks of keystream, each consisting of 2^{10} contiguous bytes.
    3. Each block is used to generate one S-box by discarding the repetitive bytes and appending the missing bytes of keystream in ascending order.

Algorithm 2. Pseudocode for performing computations in HF

```
1:  input      :=  n, $\text{KEY}_i$, for any $i \in \{0, r-1\}$
2:  output     :=  $D^i$
3:  initialise :=  $n$
4:  $\text{KEY}_i := k_0||k_1||\cdots||k_{n-1}$, where $k_i \in \mathbb{F}_2$, $\forall i \in \{0, r-1\}$
5:  $H := \text{KEY}_i$
6:  while $(n > 7)$
7:          for $j := 0$ to $\frac{n}{2}$
8:              $h_j := h_j \oplus h_{j+\frac{n}{2}}$
9:          end for
10:         $n := \frac{n}{2}$
11: end while
12: $D^i := h_0||h_1||h_2||h_3$
13: return $D^i$
```

Algorithm 3. Computations in RF

```
1:  input      :=  $m_i,\ P_1^i$, where $i \in \{0, r-1\}$
2:  output     :=  $P_3^i$
3:  initialise :=  $S_i,\ \forall i \in \{0, 7\}$, bc1
4:  $P_2^i := P_1^i \lll m_i$
5:  $P_2^i := p_0, p_1, \ldots, p_{127}$
6:  for $j := 0$ to 15
7:          $b_j := \sum_{i=0}^{7} p_{j+bc1+i} \cdot 2^i$
8:          bc1 := bc1 + 1
9:  end for
10: for $i := 0$ to 15
11:         $B_i := S_{(i\%8)}(b_i)$
12: end for
13: $P_3^i := B_0 || B_1 || \cdots || B_{15}$
14: return $P_3^i$
```

The Oppel-1 Encryption

Algorithm 4. Oppel-1 encryption process

```
 1:  input      :=  InP; $r$; KEY$_i$,   $\forall i \in \{0, r-1\}$
 2:  output     :=  The ciphertext denoted by $C$
 3:  initialise :=  StP, $S_0, S_1, \ldots, S_7$
 4:  for $i := 0$ to $r-1$
 5:          $D^i := \mathrm{HF}(\mathrm{KEY}_i)$
 6:          $m_i := \sum_{j=0}^{3} h_j \cdot 2^j$
 7:  end for
 8:  $P_b :=$ InP $\oplus$ StP
 9:  for $i := 0$ to $r-1$
10:          $P_1^i := P_b \oplus \mathrm{KEY}_i$
11:          $P_3^i := \mathrm{RF}(P_1^i, m_i)$
12:          $P_b := P_3^i$
13:  end for
14:  $C := P_b$
15:  return $C$
```

Algorithm 5. Oppel-1 decryption process

```
1:  input      :=  $C$; $r$; $\mathrm{KEY}_i$,   $\forall i \in \{0, r-1\}$
2:  input      :=  $D^i$,   $\forall i \in \{0, r-1\}$
3:  output     :=  $P$
4:  initialise :=  StK; $\mathrm{IvS}_i$,   $\forall i \in \{0, 7\}$
5:  $\mathrm{IP} := C$
6:  for $i := r-1$ down to 0
7:          $D^i := h_0||h_1||h_2||h_3$,
                where $h_j \in \mathbb{F}_2$, $\forall j \in \{0, 3\}$
8:          $m_i := \sum_{j=0}^{3} h_j \cdot 2^j$
9:          $C_1^i := \mathrm{IvS}_{(i\%8)}(\mathrm{IP})$
10:         $C_2^i := C_1^i \ggg m_i$
11:         $C_3^i := C_2^i \oplus \mathrm{KEY}_i$
12:         $\mathrm{IP} := C_3^i$
13:         $i := i - 1$
14: end for
15: $P := \mathrm{StP} \oplus \mathrm{IP}$
16: return $P$
```

Oppel-1: Analyssis

- The subkeys interact with the data blocks in two parts using a linear operator
- These interactions are:-
  - XOR of the round subkeys with the intermediate ciphertext blocks
  - For computing a positive integer $m \in \{0, 15\}$, which is used for giving circular shifts to the data blocks
- Following analysis shows that this does not reveal any information which can help in cryptanalysis of the cipher

- Let b_i^k, where $i \in \{0, 127\}$ and $k \in \{0, 4\}$ denote the bits which are used in the computation of various components
- In this case, the index i denotes the bit number and the index k denotes the iteration number of the algorithm

Table: Dry run of Algorithm 2 for computing the Hash values

First Pass	Second Pass	Third Pass
$b_0^0 \oplus b_{64}^0 = b_0^1$	$b_0^1 \oplus b_{32}^1 = b_0^2$	$b_0^2 \oplus b_{16}^2 = b_0^3$
$b_1^0 \oplus b_{65}^0 = b_1^1$	$b_1^1 \oplus b_{33}^1 = b_1^2$	$b_1^2 \oplus b_{17}^2 = b_1^3$
$\vdots$	$\vdots$	$\vdots$
$b_{62}^0 \oplus b_{126}^0 = b_{62}^1$	$b_{30}^1 \oplus b_{62}^1 = b_{32}^2$	$b_{14}^2 \oplus b_{30}^2 = b_{14}^3$
$b_{63}^0 \oplus b_{127}^0 = b_{63}^1$	$b_{31}^1 \oplus b_{63}^1 = b_{31}^2$	$b_{15}^2 \oplus b_{31}^2 = b_{15}^3$

Table: Dry run of Algorithm 2 for computing the Hash values

Fourth Pass	Fifth Pass
$b_0^3 \oplus b_8^3 = b_0^4$	$b_0^4 \oplus b_4^4 = h_0$
$b_1^3 \oplus b_9^3 = b_1^4$	$b_1^4 \oplus b_5^4 = h_1$
$\vdots$	$b_2^4 \oplus b_6^4 = h_2$
$b_6^3 \oplus b_{14}^3 = b_6^4$	$b_3^4 \oplus b_7^4 = h_3$
$b_7^3 \oplus b_{15}^3 = b_7^4$	

$$h_0 = b_0^4 \oplus b_4^4 \tag{1}$$

$$h_0 = (b_0^3 \oplus b_8^3) \oplus (b_4^3 \oplus b_{12}^3) \tag{2}$$

$$h_0 = (b_0^2 \oplus b_{16}^2) \oplus (b_{24}^2 \oplus b_{32}^2) \oplus (b_4^2 \oplus b_{20}^2) \oplus (b_{12}^2 \oplus b_{28}^2) \tag{3}$$

$$\begin{aligned} h_0 = [\{(b_0^1 \oplus b_{32}^1) \oplus (b_{16}^1 \oplus b_{48}^1)\} \oplus \{(b_{24}^1 \oplus b_{56}^1) \oplus \\ (b_{32}^1 \oplus b_{62}^1)\}] \oplus [\{(b_4^1 \oplus b_{36}^1) \oplus (b_{20}^1 \oplus b_{52}^1)\} \oplus \\ \{(b_{12}^1 \oplus b_{44}^1) \oplus (b_{28}^1 \oplus b_{60}^1)\}] \end{aligned} \tag{4}$$

$$\begin{aligned} h_0 = [\{(b_0^0 \oplus b_{64}^0) \oplus (b_{32}^0 \oplus b_{96}^0)\} \oplus \{(b_{16}^0 \oplus b_{80}^0) \oplus (b_{48}^0 \\ \oplus b_{112}^0)\}] \oplus [\{(b_{24}^0 \oplus b_{88}^0) \oplus (b_{56}^0 \oplus b_{120}^0)\} \oplus \{(b_{32}^0 \oplus b_{96}^0) \\ \oplus (b_{62}^0 \oplus b_{126}^0)\}] \oplus [\{(b_4^0 \oplus b_{68}^0) \oplus (b_{36}^0 \oplus b_{100}^0)\} \\ \oplus \{(b_{20}^0 \oplus b_{88}^0) \oplus (b_{52}^0 \oplus b_{116}^0)\}] \oplus [\{(b_{12}^0 \oplus b_{76}^0) \oplus \\ (b_{44}^0 \oplus b_{108}^0)\} \oplus \{(b_{28}^0 \oplus b_{92}^0) \oplus (b_{60}^0 \oplus b_{124}^0)\}] \end{aligned} \tag{5}$$

- It is pertinent to mention that values of h_1, h_2 & h_3 can also be computed by using Tables—VII & VIII along with techniques used in equations (1), (2), (3), (4) & (5)
- We use these equations to prove following theorems

Theorem

The Oppel-1 hash function is a one-way hash function

Proof.

Consider equation (5)
This equation shows that h_i, for some $i \in \{0, 3\}$ is computed using 32 input bits
$\because$ input bits are part of the keying material, which satisfy randomness properties
$\therefore$ h_i for some $i \in \{0, 3\}$ is a one-way hash function
$\because$ $D^i = h_0 || h_1 || h_2 || h_3$
$\therefore$ Oppel-1 hash function is a one-way hash function □

Definition

A pseudorandom sequence generator is balanced if its output sequence satisfies mono-bit frequency test

Theorem

The Oppel-1 hash function is balanced under the assumption that Oppel-1 keys are generated using a balanced pseudorandom function.

Proof.

Consider equation (5)
This equation shows that value of h_0 depends on 32 bits of keying material.
$\because$ keys are generated using a pseudorandom process.
$\therefore$ each bit has equal probability of being 0 or 1.
$\implies h_0$ is balanced.
$\because h_1, h_2$ & h_3 are computed using the same procedure.
$\therefore h_i, \forall i \in \{0, 3\}$ are balanced.
$\because D^i = h_0 || h_1 || h_2 || h_3$
$\implies D^i$ is also balanced. $\square$

Remark

Theorem 1 & 2 show that Oppel-1 hash function is statistically independent from the corresponding round keys.

Conjecture

The usage of round subkeys in the Oppel-1 hash function does not reveal any information, which can help cryptanalysis of the cipher.

- Oppel-1 contains four trivial keys which might be considered as weak keys
- These trivial keys and their possible impact on the round subkeys are shown in Table IX

Oppel-1 trivial keys and their impact on the subkeys

Table: Oppel-1 trivial keys and their impact on the subkeys

Case #	Case description	Impact on round subkeys
1.	$\mathsf{InK} = \mathsf{StK}$	$K = \underbrace{000\cdots000}_{128}$ $x = \underbrace{000\cdots000}_{128}$ $K_0 = s_0$
2.	$\mathsf{InK} = \overline{\mathsf{StK}}$	$K = \underbrace{111\cdots111}_{128}$ $x = \underbrace{111\cdots111}_{128}$ $K_0 = \overline{s_0}$
3.	$\mathsf{InK} = \underbrace{000\cdots000}_{128}$	$K = \mathsf{StK}$
4.	$\mathsf{InK} = \underbrace{111\cdots111}_{128}$	$K = \overline{\mathsf{StK}}$

Definition

A weak key for a symmetric iterated block cipher is defined as a key that can cause each subkey generated by this key to contain patterns which can assist cryptanalysis of the cipher.

Note:

It is pertinent to mention that weak keys are almost inevitable in most of the symmetric key block ciphers. For example DES contains a number of weak and semi weak keys. However known and small number of cases of weak keys can be easily avoided in implementation of these ciphers.

- The number of rounds r for an iterated block cipher depends on the completion of avalanche effect
- This criterion requires that changing one bit of input key or plaintext block should result in at least 50% random change in the binary representation of corresponding ciphertext blocks
- In order to maintain balance between the processing speed, implementation efficiency and security of the cipher, appropriate number of rounds is to be used in ciphers
- For general applications, the recommended value of r is 16. This number is derived on the basis of results of heuristic experiments which fulfill above criterion for Oppel-1.

- The necessary condition for a cipher to be used for practical applications is that the cipher output should pass a variety of statistical tests
- These tests are used to determine the closeness of a pseudorandom sequence to pure random sequence
- The statistical analysis, based on randomness tests performed on sequences generated by Oppel-1 demonstrates that these sequences do not show any significant bias from true random sequences.

Immunity Against Other Attacks

- Oppel-1 is also tested against various types of attacks, such as, distinguishing attacks, trade-off attacks, linear cryptanalysis, differential cryptanalysis, and algebraic cryptanalysis
- This analysis is briefly discussed in the following sections

Distinguishing Attacks

- Distinguishing attacks are used to distinguish the output of a cipher from a true random sequence
- Successful distinguishing attacks may lead to more severe key recovery attacks
- The analysis performed so far in this direction on sequences generated by Oppel-1 indicates the immunity of Oppel-1 against these attacks
- Therefore, following open problem still prevails and challenges the cryptographic community for solution

Open Problem VIII.1

Open Problem VIII.1

Given a sequence of length n generated as ciphertext of any arbitrary plaintext using Oppel-1, implemented in any mode of operation, and a random sequence of the same length. Can we make a distinction between the two sequences with probability greater than 0.5?

Trade-off Attacks

- Time-Memory-Data trade-off attacks are considered useful for analysing the security of block ciphers
- These attacks are chosen plaintext attacks and are generic in nature
- The main purpose of these attacks is to increase the speed of exhaustive search of the key
- For this purpose, these attacks improve the data, time or memory complexities at the expense of each other

Trade-Off Attacks

- In order to assess the security of Oppel-1 against trade-off attacks, we adopted following approach:
  - Analysis of the cipher to search for flaws in its internal structure that could help in improving either of the data, time or memory complexities
  - Improving the exhaustive search by formulating tables of ciphertext-plaintext pairs along with the corresponding keys used to generate these pairs.

Trade-Off Attacks

- The analysis performed so far in this direction showed that no weakness or technique exist that can be used to increase the exhaustive key search for Oppel-1
- In this context, following open problem still holds

Open Problem VIII.2

Open Problem VIII.2

Develop techniques based on time, memory or data trade-offs to increase the speed of exhaustive key search for cryptanalysis of Oppel-1.

Linear Cryptanalysis

- Linear cryptanalysis is a known plaintext attack
- In this type of attack, the intermediate goal of the cryptanalyst is to find affine approximations of the cipher
- These affine approximations are used to find one or more equations that relate inputs and outputs of ciphers and keying material used in these ciphers
- These types of affine approximations can lead to recovery of the secret key with an effort less than brute force attack, if they hold with probability significantly greater than 0.5

Open Problem VIII.3

- The analysis of Oppel-1 shows that no such affine approximations exist, and consequently following open problem still holds:

Open Problem VIII.3

Find affine approximations for Oppel-1 to cryptanalyse the cipher with attack complexity better than exhaustive search.

Note:

- It is pertinent to mention that the techniques of linear cryptanalysis were developed for DES, and
- up till now no information in the open literature is available for its successful application on any other prominent block cipher

Differential Cryptanalysis?

- Differential cryptanalysis is a technique for analysing the security of block ciphers
- Differential cryptanalysis is a chosen plaintext attack in which a cryptanalyst tries to find differentials

Definition

Consider a block cipher. Let ΔP denote the difference between two chosen plaintext blocks and let ΔC denote the difference between their corresponding ciphertext blocks produced by the block cipher. The combination of differences between input and output blocks, which is denoted by $(\Delta P, \Delta C)$ is defined as a differential.

Differential Cryptanalysis?

The differentials are used to define differential characteristics as follows:

Definition

The propagation of differentials through various stages of a cipher is known as differential characteristic.

Differential Cryptanalysis?

- The success of differential cryptanalysis depends on the existence of high probability differential characteristics
- The larger the probability of differential characteristics, the more vulnerable the cipher is to differential cryptanalysis
- Moreover, the number of plaintext and ciphertext pairs required for the success of differential cryptanalysis is inversely proportional to the probability of differential characteristics

Differential Cryptanalysis–Oppel-1

- In order to observe the impact of differential cryptanalysis on Oppel-1 we proceed as follows:
- Consider two plaintext blocks denoted by P and P'
- Let their corresponding ciphertext blocks be denoted by C and C'
- Then $C = E_K(P)$ and $C' = E_K(P')$, where E_K denotes Oppel-1 encryption function
- Let $\Delta P = P \oplus P'$ and $\Delta C = C \oplus C'$ denote the differences between input blocks and output blocks, respectively
- With this notation we study the propagation of differentials $(\Delta P, \Delta C)$ through various components of the cipher

The analysis performed on Oppel-1, so far could not discover any such differential trails and following open problem is still valid for Oppel-1.

Open Problem VIII.4

Open Problem VIII.4

Prove the existence of differential trails in nonlinear components of Oppel-1. If such differential trails exist then develop techniques to find these differential trails and use them to develop cryptanalytic attacks for cryptanalysing Oppel-1. These attacks should have attack complexity significantly better than brute force attack.

Algebraic Cryptanalysis?

- Traditionally, the concept of algebraic cryptanalysis is attributed to Shannon's work.
- Modern approaches of algebraic cryptanalysis are attributed to the development of efficient algorithms for solving over defined systems of multivariate polynomial equations.
- Main concept of algebraic cryptanalysis is to represent the inputs and outputs of ciphers in the form of algebraic equs.
- A cipher is considered vulnerable to algebraic cryptanalysis if not only such a representation, which connects its inputs and outputs in the form of algebraic equations is possible, but this algebraic system of equations is also solvable with an effort which is less than brute force.

- Theorem—1, i.e. "The Oppel-1 hash function is a one-way hash function" also indicates the immunity of Oppel-1 against algebraic cryptanalysis
- The so far analysis performed on Oppel-1 could not reveal any method for formulating complete systems of algebraic equations
- In this context, following open problem still holds for Oppel-1:

Open Problem VIII.5

Open Problem VIII.5

Formulate a system of mathematical equations representing the input, intermediate processing stages and output of Oppel-1 and find a method to solve such a system of equations, if it exists for cryptanalysing Oppel-1 with attack complexity better than exhaustive search.

Note:

- Although the techniques of algebraic cryptanalysis have been effectively used in combination with probabilistic event based cryptanalysis of stream ciphers.
- Until now, no practical block cipher could have been crypt-analysed by using algebraic cryptanalysis.
- This is due to the fact that modern ciphers are designed in such a way that cipher states involving input, intermediate processing stages, and output cannot be represented with the help of some system of mathematical equations.
- Moreover, if somehow, such a system could be formulated, then it should be unsolvable using prevailing mathematical techniques.

Concluding Remarks

Concluding Remarks

- Oppel-1 operates on blocks of lengths 128 bits using secret input key of the same length
- Claimed security of the cipher is 2^{128} primitive operations of its encryption algorithm
- The results of mathematical analysis of Oppel-1 are also discussed
- Immunity of Oppel-1 to various cryptanalytic attacks is also discussed
- In addition, open problems that briefly describe the design and analysis methodology adopted for Oppel-1 and provide future research directions in the context of Oppel-1 are also provided

Questions