A Thesis Title

Author Name

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of

University College London.

Department of Something

University College London

August 9, 2016

I, Author Name, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

My research is about stuff.

It begins with a study of some stuff, and then some other stuff and things.

There is a 300-word limit on your abstract.

Acknowledgements

Acknowledge all the things!

Contents

List of Figures

List of Tables

Chapter 1

Introductory Material

Some stuff about things.[1] Some more things.

Inline citation: Anne Author. Example Journal Paper Title. *Journal of Classic Examples*, 1(1):e1001745+, January 1970

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Chapter 2

Introduction to Cryptography

Cryptology is the science of hiding and recovering secret information. It mainly divided into two research areas: the area of cryptography and that of cryptanalysis. Traditionally, cryptography is the study and practise of techniques, in order to establish secure communication between two parties in the presence of unauthorized third parties, usually called adversaries or attackers. Cryptography aims to prevent the adversary from learning anything about the original content of the communication, even if he has some type of access to the communication channel.

In general, if two parties would like to share some confidential information, they will share some secret information in advance. The piece of secret information will be used to transfer the original ordinary message (plaintext P) into an unintelligible message (ciphertext C) by the sender S. Additionally, C can be transferred back to P by the receiver R using the same piece of secret information. More formally, this secret information is called key (usually a short string of bits, which is needed to decrypt the ciphertext) while the transformations are called the encryption and decryption algorithms.

Cryptanalysis is a sophisticated analysis and study of the security that a given cryptographic scheme offers. It focuses on the techniques related to recovering either the original content of an encrypted message, without the knowledge of the secret key or some fraction of information from the message. This analysis is performed under different scenarios related to the adversary's resources, type of access and his objectives. In general, The main purpose of cryptanalysis is to find the hidden weaknesses of a cryptosystem, and develop a method of decryption.

In this chapter we will give a briefly introduction about two cryptosystems, symmetric cryptography and asymmetric cryptography. We will practically discuss block ciphers in symmetric cryptography, which is widely used and well implemented in real world.

2.1 Symmetric and Asymmetric Cryptography

Cryptographic algorithms are classified based on how key material is used and managed. Normally they are classified as three groups: There are keyless algorithms which do not use any key and do not need to trust anyone; Another type of algorithms use shared key, which need to trust everyone that has the key; And the third type are private-public key algorithms, which private key is only holding by one person [**?**].

Generally a cryptosystem has sender S and receiver R who want to sent messages over an insecure channel. S and R are assumed to share a small amount of information beforehand, called the key. A cryptosystem is an encryption scheme which aims to protect the communication between S and R over the insecure channel.

A cryptosystem often contains an encryption function ε, which takes a plaintext p and a secret key k which is random bits and outputs a ciphertext $c = \varepsilon_k(p)$, and the decryption function D (inverse of ε), which takes the ciphertext c and the secret key $k^{'}$ as input and recovers the initial plaintext, i.e $D_{k'}(c) = p$. The cryptosystem should be designed in such a way that even when the adversaries can get ciphertext, they can not gain any information regarding the secret key or the plaintext.

In a cryptosystem, if $k = k^{'}$ which means the same secret key is used in both encryption and decryption, then the cryptosystem is called symmetric cryptosystem.

Definition 1 (Symmetric Cryptosystem)**.** *A symmetric encryption scheme is a five-tuple $(P, C, K, \varepsilon, D)$, where P is the finite set of plaintext, C is the finite set of ciphertext and K is the key space such that $\forall k \in K$ there is an efficiently computable* *encryption function $E_k \in \varepsilon$ which respect to random bits k,*

$$E_k : P \rightarrow C$$

and a corresponding efficiently computable decryption function $D_k \in D$,

$$D_k : C \rightarrow P$$

such that $D_k(E_k(p)) = p$ for all plaintext $p \in P$

If the keys used for encryption and decryption are different to each other, but related in a way so that decryption of a given ciphertext c results in plaintext p, then the cryptosystem is called asymmetric cryptosystem.

Definition 2 (Asymmetric Cryptosystem)**.** *An Asymmetric encryption scheme is a five-tuple $(P, C, K, \varepsilon, D)$ where P is the finite set of plaintexts, C is the finite set of ciphertexts and K is the key space, an efficiently computable key generation algorithm keyGen() randomly generate a pair of public key p_k and secret key s_k, such that $\forall p_k \in K$ there is an efficiently computable encryption function*

$$E_k \in \varepsilon$$

with respect to p_k, $E_{p_k} : P \rightarrow C$ and an $s_k \in K$ corresponding to an efficiently computable decryption function $D_{s_k} \in D$,

$$D_{s_k} : C \rightarrow P$$

such that $D_{s_k}(E_{p_k}(p)) = p$ for all plaintexts $p \in P$.

Note that symmetric cryptosystems have two algorithms: encryption and decryption. Asymmetric cryptosystems normally have at least three algorithms: key generation, encryption and decryption. In a symmetric cryptosystem, if the key is compromised, then an adversary can decrypt any message passed from sender to receiver and gains a full control over the system. Asymmetric cryptosystems solve this problem by using different but corresponding keys for encryption and decryption. However, in modern cryptography, a network requires a great number of keys which must be distributed securely.

Researchers start to solve the problem by combining both types of cryptosystem, called *hybird* encryption schema. This cryptosystem combines the convenience of asymmetric with the efficiency of a symmetric cryptosystem [**?**]. In Hybrid encryption, symmetric cryptosystem is used for encryption while the secret key shared using a protocol based on public-key cryptography. There are two main components of hybird encryption schema, Key Encapsulation Mechanism (KEM) and Data Encapsulation Mechanism (DEM). Key feature is that the two parts are independent of one another. The framework was first formalised by Cramer and Shoup in 2003 and we refer reader to [**?**] for more details.

2.2 Block Ciphers

There are two kinds of symmetric cryptosystem, stream ciphers and block ciphers. In stream ciphers a long sequence of bits is generated from a short string of key bits, and is then added bitwise modulo 2 to the plaintext to produce the ciphertext. In block ciphers the plaintext is divided into blocks of a fixed length, which are then encrypted into blocks of ciphertexts using the same key. Block ciphers are deterministic algorithms, means that the same inputs result to the same outputs. Block ciphers are considered as the hardest part of cryptography. Normally block ciphers are designed to be used for 50 years while asymmetric cryptosystems are normally designed for 10 years.

The efficiently computable encryption algorithm $E_K(P)$ and decryption algorithm $D_k(C)$ in a block cipher both uses blocks of n-bits as input and k-bits as a key K. $D_k(C)$ is the inverse of the encryption map $E_K(P)$. More formally we have that

$$C = E_K(P) : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$$

$$D_k(C) = E_K^{-1}(P) : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$$

such that $D(E_k(P)) = P \quad \forall K \in \{0,1\}^k$.

In general, for every key, a block cipher is a permutation of the form $\{0,1\}^n \to \{0,1\}^n$ for n-bit block. So in total there are $(2^n!) \simeq (2^{n-1})^{2^n}$ possible permutations. A block cipher which operates on n-bit blocks and uses k-bit keys is equivalent to give 2^k distinct permutations on n-bits. A good design of a block cipher aims to choose the 2^k permutations uniformly at random [1] from the set of all $(2^n!)$ permutations.

Block ciphers can be divided into three groups: substitution ciphers, transposition ciphers and product ciphers.

2.2.1 Substitution Ciphers

As indicated in the name, in substitution ciphers, every character in plaintext is substituted by some ciphertext character. There are four kinds of substitution ciphers: simple substitution, polyaphabetic substitution, homophonic substitution and polygram substitution [**?**]. Here we only discuss the first two kinds:

Simple Substitution In a simple substitution cipher, each plaint text character is transformed into a ciphertext character via the same encryption function E. More formally, Let $P = p_0,...,p_{n-1}$ be an n-character plaintext, $C = c_0,...c_{n-1}$ be a ciphertext, $\forall i : 0 \le i < n$

$$E : P \to C$$

$$c_i = f(p_i)$$

Around 50 B.C., Julius Caesar wrote to Marcus Cicero, using a cipher which encrypted messages by shifting every letter in the plaintext three positions to the right in the alphabet. This cipher is based *shifted alphabets*. For Caesar cipher, the secret key k is $+3 \mod 26$. In general, the cipher is easily broken by shifting the ciphertexts one position until the plaintext arises.

Polyalphabetic Substitution In a polyalphabetic substitution the characters in plaintext are transformed into ciphertext using a j-character key $K = k_0, k_1, ...k_{j-1}$, which defines j distinct encryption functions $E_{k_0}, E_{k_1}, ..., E_{k_{j-1}}$. More formally,

[1]or it is impossible to see if it was otherwise

$\forall i : 0 \leq i < n$

$$E_{k_l} : P \rightarrow C \qquad \forall l : 0 \leq l < j$$

$$c_i = E_{k_{i \ mod \ j}}(c_i)$$

The Vigenère cipher [?], first published in 1586 which uses polyalphabetic substitution is defined as follows:

$$c_i = E_{k_{i \ mod \ j}}(p_i) = p_i + k_{i \ mod \ j}$$

2.2.2 Transposition Systems

Transposition systems are essentially permutations of the characters in plaintext. Therefore a transposition cipher is defined as follows $\forall i : 0 \leq i < n$

$$\eta : \{0, ..., (n-1)\} \rightarrow \{0, ..., (n-1)\}, \ a \ permutation$$

$$c_i = E(p_i) = p_{\eta(i)}$$

Many transposition ciphers operate by blocks permute characters with a fixed period j. In that case

$$\eta : \{0, ..., (j-1)\} \rightarrow \{0, ..., (j-1)\}, \ a \ permutation$$

$$c_i = E(p_i) = p_{(i \ div \ j) + \eta(i \ mod \ j)}$$

The Vigenère and in general substitution ciphers can be broken when enough ciphertext is available to the cryptanalyst by the index of coincidence, Kasiski's method, etc. [?, ?, ?]. Transposition ciphers can be broken by using the frequency distributions for bigrams, trigrams and N-grams [?, ?, ?]. This knowledge about natural language is also very useful to our later work in password cracking.

2.2.3 Product Ciphers

To produce more stronger ciphers than the ones we have seen so far is to combine substitution and transposition ciphers. These ciphers are called *product ciphers*.

Many product ciphers have been developed, including Rotor machines [**?**]. Most block ciphers which still been using today are product ciphers. An iterated cipher is one of the product ciphers in which the ciphertexts are computed by iteratively applying a round function several times to the plaintext. In each round, a round key is combined with the text input.

Definition 3. *In an r-**round iterated block cipher** the ciphertext is computed by iteratively applying a round function g to the plaintext, such that*

$$C_i = g(C_{i-1}, K_i),\ i = 1, ..., r$$

where C_0 is the plaintext, K_i a round key and C_r is the ciphertext. Decryption is done by reversing the above function, therefore, for a fixed key K_i, g must be invertible when K_i is fixed.

Feistel Ciphers

In general, it is not easy to make an invertible function which makes the encryption and decryption process identical. One method was created by German physicist and cryptographer Horst Feistel, who was the pioneer in this area while working for American IBM. Feistel together with Don Coppersmith introduced the concept of Feistel networks while working in IBMs Lucifer cipher in 1973 [**?**]. Their work gained the respect of the U.S Federal Government who adopted it to Data Encryption standard (DES), which is based on Lucifer project with some changes done by the NSA [**?**].

Definition 4. *(Feistel Network) A Feistel cipher is an iterated cipher which maps a 2t-bit plaintext block (L_0,R_0) where L_0 and R_0 are the left and right t-bit halves respectively, to a 2t-bit block (L_r,R_r) after r-rounds of encryption.*

The result of *i*-rounds encryption $\forall i : 1 \leq i < r-1$ is computed as follows:

$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_{i-1})$$

where K_i is the i-th subkey derived from the secret key K and f the one-round function which takes a subkey and a t-block to as input to map into another t-bit block. This process is iteratively applied for r - 2 rounds. In the last round there is no swap between two halves L_{i-1} and R_{i-1}. This makes the decryption of Feistel network is the same as the encryption process. Only require a reversal of the key schedule. The final output is given by the following:

$$L_r = L_{r-1} \oplus f(R_{r-1}, K_{r-1})$$

$$R_r = R_{r-1}$$

As the encryption and decryption process are identical, the software and hardware implementation of Feistel ciphers is much easier.

Chapter 3

Cryptanalysis of Block Ciphers

The history of cryptanalysis is long and at least as fascinating as the history of cryptography. As an example, in 1917 an article in “Scientific American” claimed the Vigenère cipher is “impossible of translation” [**?**]. Today, most cryptography classes at university use it as an exercise to illustrate that this claim is not true. When discussing the security of a cryptosystem, one needs to define a model works in the real world which we will use the model of Shannon [**?**].

The sender and the receiver share a common key K over a secure channel in advance. The sender encrypts a plaintext P using the secret key K, sends ciphertext C over an insecure channel to the receiver. The receiver then decrypt C to P using K. The attacker has access to the insecure channel and can intercept ciphertext. In this chapter we assume that the sender and receiver use a secret key cipher $E_K(\cdot)$ of n-bits block size and k-bits size of key K. To evaluate the security we assume:

Assumption 1. *All keys are equally likely and a key K is always chosen uniformly at random*

Also we will assume that the attacker knows all details about the cryptographic algorithm used by the sender and receiver, except the secret key. This assumption is known as Kerckhofs's Assumption [**?**].

Assumption 2. *The enemy cryptanalyst knows all details of the enciphering process and deciphering process except for the value of the secret key.*

As an extension of the Assumption 2, we discuss the following two cases.

Assumption 3. *The enemy cryptanalyst knows all the details of the enciphering process and deciphering process except the value of the key and the S-Boxes (Substitution-box) which is a basic component of symmetric key algorithms which performs substitution.*

Under this assumption, the S-Boxes are like te master key or a high level key, which is kept as security. Similar to the rotors using in German Enigma during World War II. The attacker has to first recover the S-boxes through silicon reverse engineering or try for different sets of known S-boxes, then perform normal cryptanalysis as Assumption 2.

Assumption 4. *The enemy cryptanalyst knows all the details of the enciphering process and deciphering process except that the cipher has been tweaked, for example 90% of the cipher is what we know*

attacker has again two steps under this assumption. First recover the cipher through reverse engineering and try different sets of known S-boxes, then perform normal cryptanalysis as Assumption 2.

3.1 Classification of Attacks

Based on these assumptions, we classify the possible attacks an attacker can do [**?**]

1. **Ciphertext only attack**: The attacker processes a set of intercepted ciphertexts
2. **Known plaintext attack**: The attacker obtains $P_1, P_2, ..., P_s$ a set of s plaintexts and the corresponding ciphertexts $C_1, C_2, ..., C_s$
3. **Chosen plaintext attack**: the attacker chooses a priori set of s plaintexts $P_1, P_2, ..., P_s$ and obtains in some way the corresponding ciphertexts $C_1, C_2, ..., C_s$
4. **Adaptively chosen plaintext attack**: The attacker chooses a set of plaintext $P_1, P_2, ..., P_s$ interactively as he obtains the corresponding ciphertext

$C_1, C_2, ..., C_s$. That is the attacker chooses P_1, obtains C_1, **then** chooses P_2 etc.

5. **Chosen ciphertext attacks**: For symmetric ciphers these are similar to those of chosen plaintext attack and adaptively chosen plaintext attack

The chosen text attacks are the most powerful attacks. However, they are also unrealistic in many applications. If there exist redundancy in plaintext space, it will be very hard for an attacker to find an encrypted non-meaningful plaintexts send by the sender, and similarly hard to get ciphertexts decrypted. But if a system is secure against an adaptively chosen text attack then it is also secure against all other attacks.

Modern cryptanalysis on block ciphers has been very focused on finding the secret key K. However, there are other serious attacks for public key cryptography which do not find the secret key. [**?**] classified the types of breaks in his paper as follows:

1. **Total break**: An attacker finds the secret key K

2. **Global deduction**: An attacker finds an algorithm A, functionally equivalent to $E_K(\cdot)(or\ D_K(\cdot))$ without knowing the key K.

3. **Instance (local) deduction**: An attacker finds the plaintext (ciphertext) of an intercepted ciphertext (plaintext), which he did not obtain from the legitimate sender

4. **Information deduction**: An attacker gains some information about the key, plaintexts or ciphertexts, which he did not get directly from sender and which he did not have before the attack.

This classification is hierarchical, i.e., if a total break is possible, then a global deduction is possible etc.

Data Requirement Attacks can also be characterised by resources they require. Those resources include time complexity, memory usage and data requirements.

The first two types of resources are very obvious, however it is worth to point out data requirement is also a key component that makes an attack can work in practices. Full plaintext and ciphertext pairs are not easy to obtain in real world. Some of attacks works only if all possible plaintext and ciphertext pairs for a single key is known, and some of the attacks can work with multiple key scenarios.

3.2 Brute-force Attack

Brute-force attack or exhaustive key search is the most general attack that can be applied to any block cipher. All block ciphers are totally breakable in a ciphertext only attack, just simply by trying all the possible keys one by one and checking whether the computed plaintext is meaningful. This attack requires the computation of about 2^k encryptions. The dimension of the key space k, which is the length of the key, determines the practical feasibility of performing a brute-force attack. For modern block ciphers, brute force is always possible in theory but computationally infeasible in practise.

3.3 Linear Cryptanalysis

Linear cryptanalysis is a known plaintext attack on block ciphers. It was popularised by Matsui in 1993 [**?**]. A preliminary version of the attack on FEAL was described in 1992 [**?**]. Although the published attack on DES requires 2^{43} known plaintexts which is not very practical, it still was a great improvement in experimental cryptanalysis.

Linear cryptanalysis is based on finding affine approximations to the action of a cipher which hold with high probability. The attacker exploits linear approximations of some bits of the plaintext, ciphertext and key. In the attack on the DES (or on DES-like iterated ciphers) the linear approximations are obtained by combining approximations for each round under the assumption of independent round keys.

The attacker hopes to find an expression (equation 3.1), which holds with probability $p_L \neq \frac{1}{2}$ over all keys [**?**], such that $\varepsilon = | p_L - \frac{1}{2} |$, call the bias, is maximal.

$$(P \cdot \alpha) \oplus (C \cdot \beta) = (K \cdot \gamma) \tag{3.1}$$

where $P, C, \alpha, \beta, \gamma$ are m-bit strings and where '$\cdot$' denotes the dot product.

Given an approximation (equation 3.1) a linear attack using N plaintexts and the N corresponding ciphertexts goes as follows[**?**].

1. for all plaintexts, P, and ciphertexts, C, let T be the number of times the left hand side of equation 3.1 is 0.
2. if $T > \frac{N}{2}$ guess that $K \cdot \gamma = 0$, otherwise guess that $K \cdot \gamma = 1$ (majority vote).

By using the above method, the attacker can find one bit of information about the secret key, $K \cdot \gamma$. However, the above linear attack is not very efficient, as it only finds one bit of information about the key. In [**?**], Matsui also showed an extended linear attack which finds more key bits. Instead of approximating the first and last round, the extended linear attack simply repeat the attack for all values of the relevant key bits in those two rounds by using the following approximation equation (equation 3.2).

$$(P \cdot \alpha) \oplus (C \cdot \beta) \oplus (F(P_R, K_1) \cdot \alpha_1) \oplus (F(C_R, K_r) \cdot \alpha_r) = (K \cdot \gamma) \tag{3.2}$$

where P_R, C_R are the right halves of the plaintexts and ciphertexts. K_1 and K_R are the key bits affecting the linear approximation in the first and rth rounds. We refer reader to [**?**] for more details.

Kaliski and Robshaw showed an improved linear attack using multipe linear approximations in [**?**]. In [**?**] Knudsen and Robshaw introduced a linear attack using non-linear approximations in the outer rounds of an block cipher. Both of these are not been able to show an significant improvement compared to Matsui's linear attack. The attacks seem best suited for ciphers with large S-boxes, such as LOKI [**?**] [**?**]. In 2004, new attacks were introduced to attack Feistel schemes in [**?**] which have a small improvements than Matsui's work.

3.4 Differential Cryptanalysis

Differential cryptanalysis is based on tracking changes in the differences between two messages as theypass through the consecutive rounds of encryption. It is one of the oldest classical attacks on modern block ciphers. In cryptographic literature, it was first described and analysed by Biham and Shamir and applied to DES algorithm in the early 1990s [?]. However, Coppersmith, a member of the IBM team which designed DES [?, ?, ?], has reported that this attack was already known to IBM designers around 1974. It was known under the name of T-attack or Tickle attack, and DES had already been designed to resist this type of attack. A detailed discussion of specific original design criteria of DES can be found in [?]. Moreover, it appers that IBM had agreed with the NSA that the design criteria of DES should not be made public, precisely because it would "weaken the competitive advantage the United States enjoyed over other countries in the field of cryptography" cf. [?, ?]

Today, differential cryptanalysis is extremely well known. Numerous authors studied various aspects of differential cryptanalysis extensively in the 1990s. Apart from DES, differential cryptanalysis has also been successfully applied to a wide range of iterated ciphers [?, ?]. Recent research work also showed a great success using differential cryptanalysis to break Russian standard GOST [?, ?] and NSA lightweight cipher SIMON and SPECK [?, ?]

The data requirements for differential cryptanalysis works with multiple key scenario, where linear cryptanalysis requires data on single key. This is the main reason why differential cryptanalysis is considered more stronger than linear cryptanalysis [?].

The main task of Differential cryptanalysis is to study the propagation of input differences from round to round inside the encryption system, and find specific differences which propagate with high probability. Such pairs of input-output can be used to recover some bits of the secret key. In general, differential cryptanalysis exposes the non-uniform distribution of the output differences given one or several input differences.

Definition 5. *a* ***difference*** *between two bit strings, X and $X^{'}$ of equal length as*

$$\triangle X = X \otimes (X^{'})^{-1}$$

where $\otimes$ is the group operation on the group of bit strings used to combine the key with the text input in the round function and where $(X)^{-1}$ is the inverse element of X with respect to $\otimes$

Generally, the selection of the operator $\otimes$ depends on the way the round subkeys are introduced in each round. As in many ciphers use XOR for the key application in round function, the operator in definition 5 usually is exclusive-or ($\oplus$).

The attacker then computes the differences of the corresponding ciphertexts, hoping to detect statistical patterns in their distribution. The resulting pair of differences is called a differential. Their statistical properties depend upon the internal structure of the cipher. The attacker aim to find one particular ciphertext difference which is especially frequent. In this way, the cipher can be distinguished from random.

Truncated Differentials

Truncated Differential Cryptanalysis is a generalization of differential cryptanalysis developed by Lars Knudsen [**?**]. Unlike differential cryptanalysis which studies the propagation of the full difference between two plaintexts, truncated differential cryptanalysis consider differences that are partially determined, as in some ciphers, it is possible and advantageous to predict the values of parts of the differences after each round. This technique have been successfully applied to many block ciphers such as Russian standard GOST [**?**, **?**, **?**]. Truncated differential is defined as follow: [**?**].

Definition 6. *A differential that predicts only parts of an n-bit value is called a truncated differential. More formally, let(a,b) be an i-round differential. If $a^{'}$ is a subsequence of a and $b^{'}$, then ($a^{'}$,$b^{'}$) is called an i-round truncated differential.*

A truncated differential can be considered as a collection of differentials. For example, let $(a^{'},b)$ be the truncated differential of an n-bit block cipher, where $a^{'}$ specifies the least $n^{'} < n$ significant bits of the plaintext difference and b specifies the ciphertext difference of length n. This differential is a collection of all $2^{n-n^{'}}$ differentials (a,b), where a is any value that truncated to the $n^{'}$ least significant bits is $a^{'}$.

3.5 Algebraic Attacks

In general the security of a given block cipher will grow exponentially with the number of rounds and so does the number of required plaintext-ciphertext pairs which are needed in a linear and differential cryptanalytic attack. However, in most cases, only a few plaintext-ciphertext pairs are available for cryptanalysis, linear or differential attacks are not expected to succeed. Thus, a new method needs to be invented which will be able to recover the secret key with only a few information is available.

Claude Shannon, has once suggested that the security of a cipher should be related to the difficulty of solving the underlying system of equations and deriving the key:

> *"if we could show that solving a certain system requires at least as much work as solving a system of simultaneous equations in a large number of unknowns, of a complex type, then we would have a lower bound of sorts for the work characteristic"* [**?**].

This is the core concept behind algebraic attacks.

An algebraic attack is a form of known plaintext attack which consists of the following two steps:

1. **Modelling**: Describe the cipher as a multivariate system of polynomial equations over a small finite field (like GF(2)) or logical constraints in terms of the secret key K, the plaintext P and the ciphertext C.

$$f_1(K,P,C) = 0, \ldots f_r(K,P,C) = 0 \Longleftrightarrow E(K,P) = C$$

2. **Solving**: Solve the underlying multivariate system of equations and obtain the secret key. In order to reduce the complexity of solving the system we substitute to the system all known plaintext/ciphertext pairs. The more the pairs the more the equations we obtain involving the key bits. For many known attack this can make the problem easier to solve.

Generally, each module of a block cipher can be described with a set of algebraic equations. Put these algebraic equation together, we can get a large precise system describing the whole cipher. The main idea of algebraic attack is to establish a series of low complexity algebraic equations of initial plaintext and ciphertext, and then find the secret key by solving the equations. Such equation systems are normally very large systems of quadratic multivariate polynomial equations over GF(2). Each variable of such equation system represents a state-bit of the encryption algorithm. Then the variables which representing state-bits from the initial round are set according to values of the plaintext, and similarly variables which representing state-bits from the last round are set according to values to the ciphertext. Therefore, the security of a block cipher depends on whether there exist an efficient algorithm to solve such large and sparse multivariate polynomial equations. If we can solve the equation system faster than exhaustive key search attack, the cipher will be broken (in academic sense).

In addition, some high profile cryptanalysis problems in public key cryptography can be written in such a form which contains a “block cipher topology” [1]. Such equations have similar structure as a block cipher, where the beginning inputs and final outputs variables are easy to get and the middle part of the equation system are very hard to analyse (see figure 3.1). One example is Semaev's summation polynomial equations for elliptic curve [**?**, **?**] see section **??**.

3.5.1 Algebraic Attacks Solving Stage

Solving a random system of multivariate non-linear boolean equations is an NP-hard problem [**?**]. As many cryptographic primitives can be described by a sparse multivariate non-linear system of equations over $GF(2)$ or any other algebraic systems. Several techniques were develop in order to tackle the problem of solving such equations. A classic approach is to use techniques from algebraic geometry and especially Gröbner bases algorithms to solve the system of equations [**?**]. But most of the times they do not lead to solutions in practise due to the extremely high memory requirements. Then, some heuristic techniques were developed called *lin-*

[1] which is how we / Dr. Courtois classify the equations proposed by Semaev in 2015 [**?**, **?**, **?**]

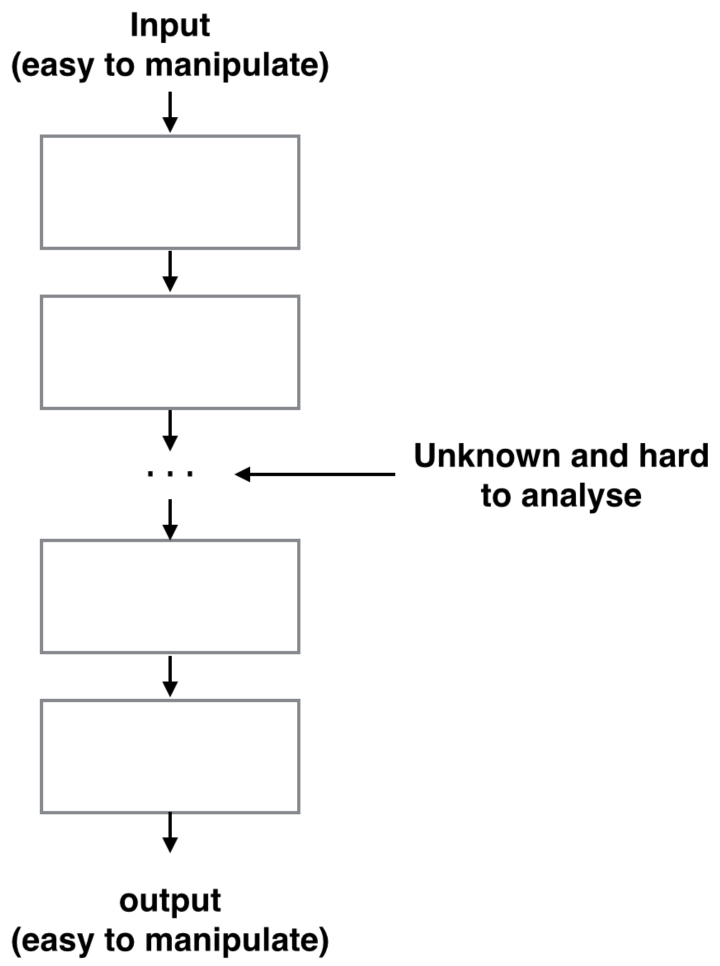

Figure 3.1: Block cipher topology: attackers are easy to control or manipulate the inputs and outputs but very hard to analyse variables in the middle

earisation [**?**], where all the non-linear terms are replaced by an independent linear variable and the resulting linear system can be solved using Gaussian elimination [**?**]. However, it requires that there are enough linearly independent equations and the initial system is highly over-defined and sparse. Then, XL algorithm [**?**, **?**] was developed to make the system over-defined, by addition of new equations to the current system. The proposal of XL made it possible to solve the multi-order nonlinear equations within polynomial time, which accelerated the development of the algebraic attacks. Then researchers focus mainly on the fast solutions to the algebraic equation systems.

In the past few years, researchers try to solve the problem by using tools and software, such as SAT solvers. Nicolas Courtois is the pioneer of bringing SAT solvers into action in the area of symmetric cryptanalysis. His paper [**?**, **?**] described how to covert such equations to a format which can be solved automatically by SAT solvers. The advantage of such technique is that SAT solvers can perform reasonably well and do not require a lot of memory as in case of Gr*ö*bner basis-based techniques [**?**]. The only disadvantage is the unpredictability of its complexity. The first algebraic attack on reduce round block cipher DES was done by Courtois and Bard in [**?**]. Later in 2008, first algebraic attack on full block cipher KeeLoq [**?**]. In the past 10 years, SAT solvers were used for attacks on block ciphers such as GOST[**?**, **?**] and Chinese block cipher SMS4 [**?**], stream ciphers such as Crypto-1, HiTag2 and Bivium [**?**, **?**], and also MiFare Classic smart cards [**?**].

Another method, is to use ElimLin algorithm [**?**]. ElimLin stands for **Elim**inate **Lin**ear and it is a simple algorithm for solving polynomial systems of multivariate equations over small finite fields and was initially proposed as a single tool by Courtois to attack DES and CTC/CTC2 ciphers [**?**]. It is also known as "inter-reduction" step in all major algebra systems.

Its main aim is to reveal some hidden linear equations existing in the ideal generated by the system of polynomials. ElimLin is composed of two sequential stages, as follows:

- **Gaussian Elimination:** Discover all the linear equations in the linear span of initial equations.

- **Substitution:** Variables are iteratively eliminated in the whole system based on linear equations found until there is no linear equation left.

Given an initial multivariate system of equations over S^0 in $\mathbb{F}_2[x_1, x_2, .., x_n]$, then the ElimLin is formally described as below.

This method is iterated until no linear equation is obtained in the linear span of the system. Intuitively, ElimLin seems to work better in cases where there is low non-linearity since this implies the existence of more linear equations. MC is an-

Algorithm 1 ElimLin Algorithm

```
**Input:** $S^0 = \{f_1, f_2, ..., f_m\} \in \mathbb{F}_2[x_1, x_2, .., x_n]$
**Output:** An updated system of equations $S^T$ and a system of linear equations $S_L$
1. Set $S_L \leftarrow \emptyset$ and $S^T \leftarrow S^0$ and $k \leftarrow 1$
2. **Repeat**
For some ordering of equations and monomials perform $Gauss(S^T)$ to eliminate non-linear monomials
Set $S_{L'} \leftarrow$ Linear Equations from $Gauss(S^T)$
Set $S^T \leftarrow Gauss(S^T) \backslash S_{L'}$
Let $l \in S_{L'}$, $l$ non-trivial (if unsolvable then *terminate*)
Let $x_{i_k}$ a monomial in $l$
Substitute $x_{i_k}$ in $S^T$ and $S_{L'}$ using $l$
Insert $l$ in $S_L$
$k \leftarrow k + 1$
```

other notion of non-linearity [**?**, **?**] and possibly such method may work sufficiently well in cryptographic primitives of low MC.

Chapter 4

General Conclusions

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Appendix A

An Appendix About Stuff

(stuff)

Appendix B

Another Appendix About Things

(things)

Appendix C

Colophon

This is a description of the tools you used to make your thesis. It helps people make future documents, reminds you, and looks good.

(example) This document was set in the Times Roman typeface using LaTeX and BibTeX, composed with a text editor.

Bibliography

[1] Anne Author. Example Journal Paper Title. *Journal of Classic Examples*, 1(1):e1001745+, January 1970.