HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Engineering Physics and Mathematics

Miia Hermelin

CRYPTOGRAPHIC PROPERTIES OF THE BLUETOOTH COMBINATION GENERATOR

Master's Thesis

Supervisor: Professor Olavi Nevanlinna
Instructor: Research Fellow, Docent Kaisa Nyberg

Helsinki, 29th February 2000

Tekijä:	Miia Hermelin
Työn nimi:	Bluetooth yhdistäjägeneraattorin kryptografisia ominaisuuksia
English title:	Cryptographic properties of the Bluetooth combination generator
Päivämäärä:	29. helmikuuta 2000 **Sivumäärä:** 55
Osasto:	Teknillisen fysiikan ja matematiikan osasto
Professuuri:	Mat-1 Matematiikka
Työn valvoja:	Professori Olavi Nevanlinna
Työn ohjaaja:	Research Fellow, dosentti Kaisa Nyberg

Bluetooth on ad-hoc-verkoissa sovellettava tekninen spesifikaatio, josta vastaa Bluetoothin Special Interest Group (SIG). Työssä on keskitytty Bluetoothia varten kehittyyn yhdistäjägeneraattori-tyyppisen vuosalaimeen E_0 ja erityisesti sen kryptografisiin ominaisuuksiin. Työssä tutkitaan erityisesti korrelaatio-ominaisuuksia, mutta myös muita turvallisuuteen vaikuttavia tekijöitä tarkastellaan.

Yhdistäjägeneraattori koostuu lineaarisista takaisinkytkentäsiirtorekistereistä (LFSR), muistista, sekä lineaarisista ja epälineaarisista yhdistelyfunktioista. LFSRien yhteispituus on Bluetoothissa 128 bittiä ja muistissa on neljä bittiä. LFSRien ominaisuuksia tarkastellaan äärellisten kuntien teorian avulla. Tässä yhteydessä lukijalta oletetaan perustiedot algebrasta ja lukuteoriasta. Lisäksi tarkastellaan, millaiset LFSRt ovat turvallisuuden kannalta optimaalisia.

Matemaattisen tarkastelun jälkeen otetaan huomioon LFSRien tietotekniset ominaisuudet. Näiden todetaan olevan sellaisenaan riittämättömät turvalliseen tiedonsiirtoon. Näin ollen päädytään yhdistämään rekisterejä yhdistäjägeneraattoriksi. Näiden tutkimiseen käytetään korrelaatioanalyysia.

Järjestelmä on sitä turvallisempi mitä vähemmän sen tulosteella on korrelaatiota sen sisäänsyötettyjen bittien minkään yksittäisen lineaarisen yhdistelmän kanssa. Lineaarisuuden mittaamiseksi on työssä todistettu joukko korrelaatiolauseita, joita sitten käytetään yhdistäjägeneraattorien tutkimiseen. Bluetoothin lisäksi tarkastellaan yhdistäjägeneraattoria, Meierin ja Staffelbachin analyysia seuraten.

Työssä esitellään yhdentyyppinen hajoita ja hallitse -hyökkäys. Sitä sovelletaan sekä yhden muistibitin esimerkkiin, että Bluetoothiin. Lisäksi todetaan, että hyökkäys Bluetoothia vastaan on teoreettinen, koska Bluetoothilla on tarkoitus tulostaa yhdellä alustuksella alle 2746 bittiä, kun hyökkäys vaatisi $\mathcal{O}(2^{64})$ bittiä. Työssä todistetaan myös, että tekemällä pieni muutos Bluetoothin yhdistäjäfunktioon sen korrelaatio-ominaisuudet paranevat. Tämän seurauksena esitetään parannusehdotus, joka ei vaikuta järjestelmän suorituskykyyn tai hintaan.

Avainsanat: Bluetooth, yhdistäjägeneraattorit, ad-hoc verkot, lineaariset takaisinkytkentäsiirtorekisterit, korrelaatio, lineaarisuus

Ei lainata ennen: **Työn sijaintipaikka:**

Author: Miia Hermelin

Title of thesis: Cryptographic properties of the Bluetooth combination generator

Finnish title: Bluetooth yhdistäjägeneraattorin kryptografisia ominaisuuksia

Date: 29th February 2000 **Pages:** 55

Department: Department of Engineering Physics and Mathematics

Chair: Mat-1 Mathematics

Supervisor: Professor Olavi Nevanlinna

Instructor: Research fellow, docent Kaisa Nyberg

Bluetooth is a technical specification designed for ad-hoc nets. It is developed and maintained by the Bluetooth Special Interest Group. This thesis concentrates on the combination generator defined in the algorithm E_0 of Bluetooth and on its cryptographical properties. The thesis is mainly concentrated on the correlation properties of Bluetooth, but other security issues are also considered.

A combination generator consists of linear feedback shift registers (LFSR), a memory and linear and nonlinear combiner functions. In Bluetooth, there are four bits of memory and four LFSRs of total length of 128 bits. Finite fields are used in studying the properties of LFSRs. Therefore the reader is assumed to have the basic knowledge about algebra and number theory. Based on earlier studies, it is also investigated what kind of LFSRs are optimal for provision of cryptographic security.

In addition to the mathematical point of view, the computer scientific properties of the LFSRs are studied. It is concluded that they do not provide adequate security for data transfer. This can be corrected by joining several LFSRs as a combination generator.

A combination generator is the more secure the less its output has correlation to a single linear function of its inputs. To measure the linearity of an output function, a few correlation theorems are proved in the thesis. They are later used when studying the combination generator. In addition to Bluetooth, that has four bits of memory, a one-bit memory system is also studied.

An Ultimate Divide and Conquer attack is presented in the thesis. It is applied to both the case of one bit as well as the Bluetooth. It is noted, that the attack against the Bluetooth is theoretical, since the Bluetooth is intended to output blocks of at most 2746 bits while the attack needs approximately $\mathcal{O}(2^{64})$ bits. It is also proved that making a small modification in the Bluetooth combination function would improve its correlation properties to some extent. As a consequence, a modification that does not affect the system's performance or cost is presented.

Keywords: Bluetooth, combination generators, ad-hoc nets, linear feedback shift registers, correlation, linearity

Not borrowable till: **Library code:**

Preface

The writing and studying the subject of this thesis has been very educational as well as interesting. Therefore I would like to thank Olavi Nevanlinna for encouragement and advises concerning this work. Further, I wish to express my gratefulness for Kaisa Nyberg for guidance and Valtteri Niemi for his advises. I would also like to thank Asko Vilavaara for the opportunity to work here at Nokia Research Center and Shreekanth Lakshmeshwar for computing the needed Walsh-transforms.

I am also grateful for the help and advises given me by the people of the Department of Mathematics, especially Ville Havu and Kenrick Bingham. Finally, I thank my parents for their support and buns and Kimmo for being. Faith manages.

Helsinki, 29th February 2000

Miia Hermelin

Contents

Chapter 1

Introduction

More and more information is transferred every day between different equipment. Depending on the amount and quality of the information, as well as the machines used, different technologies for exchange of information have been adopted. An ad hoc network is a dynamic multi-hop wireless network that is established by a group of mobile hosts on a shared wireless channel by virtue of their proximity to each other. Since wireless transmissions are locally broadcast in the region of the transmitting host, hosts that are in close proximity can hear each other and are said to be neighbors. The transitive closure of the neighborhood of all the hosts in the set of mobile hosts under consideration forms an ad hoc network. All hosts in the ad-hoc net need some common way to transmit data securely. One proposal is a chip that can be installed in the mobile hosts for example PCs or mobile phones.

The Bluetooth specification [SIG99] introduces a fast, short ranged, and low cost technology that defines such a chip, its bandwidth, the conditions it should be able to work in etc. With Bluetooth, no cables are needed. In the future, it might replace the need of multiple cables with a single radio link. However, Bluetooth's range of operation is less than ten meters so it can not be used in long range data transfer.

Bluetooth is the code name for the technology specification, shown in [SIG99]. The specification is public in the sense that all the parts that have adopted it, can have access to it. The Bluetooth Special Interest Group (SIG), consisting of Ericsson, IBM Corporation, Intel Corporation, Nokia, Toshiba Corporation, Com Corporation, Lucent Technologies, Microsoft Corporation and Motorola Inc., is responsible for developing and marketing the Bluetooth.

To secure the exchange of information, Bluetooth has been equipped with a stream cipher, which is a nonlinear combination generator defined as the algorithm E_0. It produces the key sequence, which is used to encipher the messages to be sent. This thesis concentrates on analyzing the cryptographical, i.e., security related properties of this algorithm. Specifically, the correlation properties of the Bluetooth are investigated.

There are many other properties that should be considered. For example, the key management is mostly a wide open problem. It is not known, yet, how the original secret – a random key – should be established between two parties. In this thesis it is assumed that both the sender and receiver has the needed information.

Linear Feedback Shift Registers (LFSR) are used to produce pseudo-random sequences. They are however not adequately secure, since they are linear. There are several ways to reduce the linearity. Rueppel has proposed one solution in [Rue86]: The linear system of several LFSRs could be equipped with a memory of one or more bits and some nonlinear combination function. This kind of system is called a combination generator with memory.

Combination generators are analyzed by their statistical properties. That means that their linearity is "measured" by studying their correlation properties. As a consequence, some correlation theorems presented in the paper of K. Nyberg [Nyb99] are introduced, and then applied in the analysis of Bluetooth.

Meier and Staffelbach have proved in [MS92] that a combination generator with one bit of memory is easy to break. This paper has been analyzed in this thesis. Some of the assumptions and results have been modified.

The Bluetooth stream cipher E_0 has a four-bit memory and can be seen as a more complicated case of the combination generator with one memory bit. Bluetooth's cryptographical properties are shown to be quite adequate for its intended usage, however, a slight theoretical improvement is shown in the end of the thesis.

The presentation in this thesis is self-contained. Information about the Bluetooth specification that is needed to follow the analysis in this thesis is presented. The reader should be familiar with the basics about finite fields, number theory and algebra.

1.1 Thesis outline

The thesis consists of four main chapters, excluding the introduction and conclusions.

In Chapter 2, the basics about stream ciphers, pseudo-random generators and LFSRs are stated. It also introduces a few postulates that state, which properties a stream cipher system should fulfill. It is proved that certain LFSRs are optimal in this respect, and the conditions to be met to fulfill the requirements are stated. In the end, it is noticed that an LFSR does not fulfill all the postulates and hence is not adequately secure on its own.

Chapter 3 introduces mathematical tools including the Walsh-transform and correlations. A few correlation theorems are stated and proved. The linearity of a cryptosystem is also defined.

Chapter 4 shows, how a summation combiner is formed from several LFSRs. It presents a simple example with a one-bit memory and analyzes the example shown in [MS92]. It is noted that this system resists the basic Divide and Conquer attack, but fails under it if correlation relations are extended to cover more time instances. In [Nyb99] a more complex attack, named Ultimate Divide and Conquer has been presented. It is stated in a theorem that gives a lower bound for the number of output bits needed to make the attack. Ultimate Divide and Conquer is also applied to the one-bit case.

In Chapter 5 Bluetooth's combination generator algorithm E_0 is introduced and analyzed. The correlation properties of the E_0's combination function are determined and used in estimating the complexity of the Ultimate Divide and Conquer attack to Bluetooth. The correlation properties are also analyzed in more detail in proving a theorem that states how the system could be improved.

Chapter 2

Linear Feedback Shift Registers

A cryptosystem is used to encrypt *plaintext* into *ciphertext*. The system involves a secret key, on which most of the security of the system is based. The formal definition of a cryptosystem is presented next.

Definition 2.1. A cryptosystem is a five-tuple $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, where the following conditions are satisfied:

1. $\mathcal{P}$ is a finite set of possible plaintexts
2. $\mathcal{C}$ is a finite set of possible ciphertexts
3. The keyspace $\mathcal{K}$ is a finite set of possible keys
4. For each key $K \in \mathcal{K}$, there is an encryption rule $e_K \in \mathcal{E}$ and a corresponding decryption rule $d_K \in \mathcal{D}$. Each $e_K : \mathcal{P} \to \mathcal{C}$ and $d_k : \mathcal{C} \to \mathcal{P}$ are functions such that $d_K(e_K(a)) = a$ for every plaintext $a \in \mathcal{P}$.

Computers use binary digits in their calculations, so it is customary to have $\mathcal{P} = \mathcal{C} = \mathbb{Z}_2^m$, where m is the length of the plaintext to be enciphered. This is the case in this thesis. The key is also binary valued.

Sometimes the security of a cryptosystem is based on the secrecy of the encryption algorithm. This method is called "security by obscurity", and it is not recommended. Nowadays most cryptosystems - including the Bluetooth specification – obey the Kerkhoff's principle. It says that the algorithm is public and the security depends only on the secrecy of the key.

A *stream cipher* is a binary encryption system. It employs a function, which generates a key bit sequence $K = (k_0, k_1, \ldots, k_{n-1})$, $k_i \in GF(2)$, where n is the length of the message. It then adds modulo two the plaintext bits $\mathbf{a} = (a_0, a_1, \ldots, a_{n-1})$, $a_i \in GF(2)$, with the corresponding key bits to produce the ciphertext $\mathbf{b} = (b_0, b_1, \ldots, b_{n-1})$, $b_i \in GF(2)$:

$$a_i + k_i \equiv b_i \pmod 2,\ i = 0, \ldots, n-1.$$

Clearly, deciphering is done by adding the encrypted message and the secret key modulo two. This implies that both the sender and the user must have access to the secret key.

A perfectly secure cryptosystem called the *one-time-pad* is a stream cipher system with the key generation being perfectly random. This means usually measuring some random phenomena in nature, like the movements of particles.

The security of one-time-pad lies completely on K. If the attacker does not know K, she can not have any information on $\mathbf{a}$ even if she knows $\mathbf{b}$. Moreover, knowing part of K does reveal only the corresponding part of $\mathbf{a}$, as K is random. This seems to be a perfect way to encipher data, but this is true only in theory. In practice this is not very useful, at least in usual applications. This is because of long messages, and, thus, because of long keys, which makes the key generation both time consuming and difficult. In addition, the length of the key puts some limitation on the use of the system, as the key management (transmitting and storing the key) becomes difficult.

Because of these practical limitations of one-time-pad it can not be used as such. The key generation should be modified to be easier, but it should still be somewhat random to maintain security.

The trick is to lengthen the original secret key with a *pseudo-random generator* (PRG). The key K of finite length n is used to initialize a PRG which outputs a bitstring $\{x_i\}_{i\geq 0}$, of length greater than n.

It should be stressed that this is not as secure as one-time-pad, since the new key is not perfectly random. Also, if the system needs a perfectly random key sequence, then PRG must not be used to replace it. The PRG is the better, the more the PRG's output resembles the output of a one-time-pad, i.e., the harder it is to distinguish the two from each other. Different kind of statistical tests have been used to make the distinction. In this thesis these tests are based on the correlation properties of the system.

The next section defines PRGs more precisely and states some statistical properties the PRG should fulfill.

2.1 Pseudo-random generators

As mentioned above, pseudo-random generators, PRGs, are used to produce a keystream for a stream cipher cryptosystem.

The pseudo-random generator inputs some *initial state vector* $\{x_0, \dots, x_{n-1}\}$, which is derived from the fixed-length key with perhaps some additional non-secret information such as the sender's identifier. The output of the generator is a bitstream $\{x_i\}_{i\geq 0}$, which is used to encrypt the message $\mathbf{a} = (a_0, a_1, \dots)$

as

$$a_i + x_i \equiv b_i \pmod 2,\ 0 \leq i \leq \mathrm{len}(\mathbf{a}) - 1,$$

where $\{b_i\}_{i\geq 0}$ is the cipher text.

A common type of PRG, a feedback shift register, is depicted in Figure 2.1.

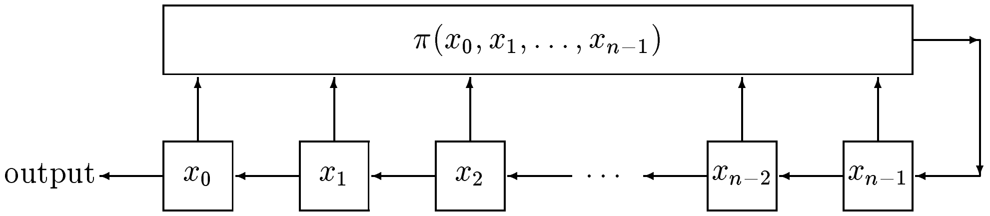

Figure 2.1: A feedback shift register.

We define the *period* of a binary sequence $\{x_i\}_{i\geq 0}$ to be the smallest integer p satisfying

$$x_{i+p} = x_i,$$

for all $i \geq 0$. A *run* of length k starts at moment t, if

$$x_{t-1} \neq x_t = x_{t+1} = \cdots = x_{t+k-1} \neq x_{t+k}.$$

A *block* and a *gap* of length k is a run of k consecutive 1's and 0's, respectively.

The *autocorrelation* AC(k) of a periodic sequence $\{x_i\}_{i\geq 0}$ with period p is

$$\mathrm{AC}(k) = \frac{A - D}{p},$$

where

$$A = |\{0 \leq i < p | x_i = x_{i+k}\}|$$

and

$$D = |\{0 \leq i < p | x_i \neq x_{i+k}\}|.$$

This means that A is the number of agreements between $\{x_i\}$ and $\{x_{i+k}\}$ over a full period of time. Respectively, D gives the number of disagreements.

If k is a multiple of period p, then the autocorrelation, whose value is then 1, is said to be *in-phase*. If this is not the case, then it is called *out-of-phase*.

There are different ways to define the quality of a PRG. The *Golomb's Randomness Postulates* state a few facts a PRG should fulfill [vT88].

G-1 The number of zeros and the number of ones are as equal as possible per period, i.e., $p/2$ if p is even, and $(p \pm 1)/2$ if p is odd.

G-2 Half of the runs in a cycle have length 1, one quarter have length 2, ..., 2^{-k} runs have length k. Half of the runs of a certain length are gaps and the other half are blocks.

G-3 The out-of-phase autocorrelation $\mathrm{AC}(k)$ has the same value for all k.

The first rule G-1 says that zeros and ones occur with roughly the same probability. The rule G-2 states that after 01 the symbol 0 has almost the same probability as the symbol 1 etc. The last rule G-3 says that counting the number of agreements between a sequence and a shifted version of that sequence does not give any information about the period of the sequence unless one shifts over a multiple of the period.

A brief list of *Cryptographic properties* for the PRGs can be given as follows:

C-1 The period p of $\{x_i\}$ has to be taken very large.

C-2 The sequence $\{x_i\}$ should be easy to generate.

C-3 The system should resist the *known-plaintext* attack.

"Very large" in the rule C-1 depends mainly on the computing power that could be assumed to be in use. Roughly speaking, there is no use in solving some cryptosystem, if the price of the solving outweighs the value of the messages. So "very large" could be different for different senders and receivers and also for different cryptosystems. In general, rule number one is fulfilled, if the result is exponential with respect to the length of the PRG's initial state.

The second rule C-2 states the properties of the implementation either in hardware or software.

The rule C-3 demands that knowledge of a part of the plaintext with corresponding ciphertext should not enable a cryptanalyst to generate any more terms of the sequence $\{x_i\}$.

All of these rules are practical in nature and more dependent on specific user requirements than Golomb's postulates. For example, a cryptosystem could produce a very long period and it could resist the known-plaintext attack. However, if the sequence is hard to generate, i.e., it costs too much or is too time consuming, then the PRG is not practical. It should be noted that in one-time-pad the random key does resist the known-plaintext attack, but as said before, the key generation and distribution is usually too difficult.

The next section will give some algebraic background for the analysis of linear feedback shift registers.

2.2 About finite fields

Most of these calculations are done in $GF(2)$, where modulo two arithmetic is used. However, for clarity, and because many of these results can be generalized to any field [LN94], we use the notation "+" and "$\sum$" even in the case of $GF(2)$. The distinction between sums in $\mathbb{Z}$ and $GF(2) = \mathbb{Z}_2$ shall be done later in Chapter 3.

Definition 2.2. A subring $(\pi(t)) = \{\pi(t)\tau(t) | \tau \in \mathbb{Z}_2[t]\}$ of a polynomial ring $\mathbb{Z}_2[t]$ is called an ideal generated by $\pi(t)$.

We denote by $\deg(\pi(t))$ the degree of the polynomial $\pi(t)$. If π is irreducible and $\deg(\pi) = n$, then one can define Galois field $GF(2^n)\mathbb{Z}_2[t]/(\pi(t))$.

Theorem 2.3. *Let $\pi \in \mathbb{Z}_2[t]$ be a polynomial with $\deg(\pi) = n \geq 1$ and $\pi(0) \neq 0$. Then there exists a positive integer $m \leq 2^n - 1$ such that $\pi(t)$ divides $t^m - 1$.*

Proof. There are $2^n - 1$ non-zero residue classes in the residue class ring $\mathbb{Z}_2[t]/(\pi(t))$. On the other hand, there are 2^n non-zero elements in

$$\mathcal{D} = \{t^j + (\pi(t)) \,|\, j = 0, 1, \ldots, 2^n - 1\}.$$

This means that there must be two equivalent elements in $\mathcal{D}$, i.e., there are integers s and r, such that $0 \leq s < r \leq 2^n - 1$ and $t^s \equiv t^r \pmod{\pi(t)}$. Since $\pi(0) \neq 0$, then t and $\pi(t)$ are relatively prime, so there exists t^{-1} modulo $\pi(t)$. We get $t^{r-s} \equiv 1 \pmod{\pi(t)}$, or $\pi(t)|(t^{r-s} - 1)$. □

Since Theorem 2.3 applies, the following definition is sensible:

Definition 2.4. The order of polynomial $\pi(t) \in \mathbb{Z}_2[t]$ is the smallest positive integer m, such that $\pi(t)|(t^m - 1)$.

The concept of order for an element of a finite field is needed in the following.

Definition 2.5. The order of $\alpha \in GF(2^n)$ is the smallest positive $d \in \mathbb{N}$, with $\alpha^d = 1$.

Assume, we have an irreducible polynomial $\pi(t)$, whose order is m and $\deg(\pi(t)) = n$. We can then define a Galois field as above with $\pi(t)$. But then for $\alpha = \alpha(t) = t \in GF(2^n)$ it applies that $\pi(\alpha) = 0$ and so $\alpha^m = 1$. Let the order of α be d. Since d is the smallest positive integer with $\alpha^d = 1$ it must be that $d \leq m$. On the other hand, since $\pi(t)$ is irreducible and $\pi(\alpha) = 0$, it follows that $\pi(t)$ is the minimal polynomial of α, [vT88], i.e., it divides all other polynomials which have α as a root, especially, $\pi(t)$ divides $t^d - 1$. Hence, $m \leq d$ and we conclude that $m = d$. By Euler's theorem $\alpha^{2^n - 1} = 1$ [LN94], so $m = d$ divides $2^n - 1$. We have proved the following lemma.

Lemma 2.6. *The order of an irreducible polynomial* $\pi(t)$ *divides* $2^{\deg(\pi(t))} - 1$.

Definition 2.7. If the order of an irreducible polynomial $\pi(t) \in \mathbb{Z}_2[x]$ is $2^n - 1$, where $n = \deg(\pi(t))$, then the polynomial is called primitive.

Primitivity will be of great importance in the next section that handles linear feedback shift registers.

2.3 LFSRs mathematically

A *linear feedback shift register* (LFSR) is a special case of a pseudo-random sequence generator. For implementation purposes the field $GF(2)$ is usually chosen, but theoretically any field would apply.

A general LFSR of length n is shown in Figure 2.2.

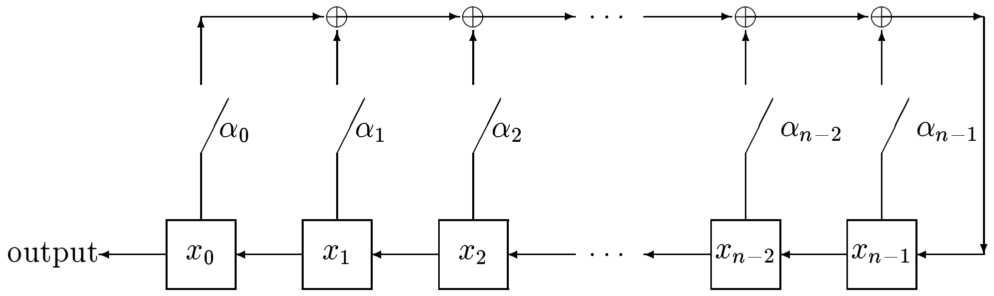

Figure 2.2: A linear shift feedback register, LFSR.

The coefficients $\alpha_i \in F(= GF(2)), i = 0, ..., n-1$ are called the *feedback coefficients*. Having $\alpha_i = 1$ means that the corresponding gate in an LFSR is closed, otherwise it is open. Mathematically this can be stated as

$$x_{k+n} = \sum_{i=0}^{n-1} \alpha_i x_{k+i},\ k \geq 0. \tag{2.1}$$

or

$$\sum_{i=0}^{n} \alpha_i x_{k+i} = 0,\ k \geq 0, \tag{2.2}$$

where $\alpha_n = 1$. It is also assumed that $\alpha_0 = 1$. Were it not, the LFSR would produce a sequence that is just a delayed version of a sequence generated by an other LFSR with $\alpha_0 = 1$.

We can write equation (2.1) as a matrix equation

$$\mathbf{x}^{k+1} = \begin{pmatrix} x_{k+1} \\ x_{k+2} \\ \vdots \\ x_{k+n} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ \alpha_0 & \alpha_1 & \alpha_2 & \dots & \alpha_{n-1} \end{pmatrix} \begin{pmatrix} x_k \\ x_{k+1} \\ \vdots \\ x_{k+n-1} \end{pmatrix},$$

where the matrix

$$T = \begin{pmatrix} \mathbf{0} & I_{n-1} \\ \alpha_0 & \alpha_1 \dots \alpha_{n-1} \end{pmatrix}$$

is the state transition matrix. Here I_{n-1} denotes the $(n-1) \times (n-1)$ identity matrix and $\mathbf{0}$ is the $1 \times (n-1)$ zero-vector. Matrix T is invertible if and only if

$$\det T = \alpha_0 = 1.$$

Since we assumed this (by the notion above) we have the next theorem.

Theorem 2.8. *The sequence produced by an LFSR is uniquely determined by its initial state and its feedback coefficients. Moreover, there is $n \in \mathbb{N}$ such that $T^n = I$, and all sequences produced by the LFSR are periodic.*

The feedback coefficients α_i are often given in a form of a *characteristic polynomial*

$$\pi(t) = \sum_{i=0}^{n} \alpha_i t^i. \tag{2.3}$$

The output sequence, on the other hand, is associated with a *generating polynomial* or a *power series*:

$$S(t) = \sum_{i=0}^{\infty} x_i t^i \tag{2.4}$$

An important cryptographic requirement is that an LFSR should produce sequences with as long periods as possible. There are 2^n different states in an LFSR of length n. Clearly the all-zero state just keeps repeating itself. That is why the period is at most $m = 2^n - 1$. Next we shall see that it is possible to reach this period. It will be proved that primitive polynomials, and only them, generate sequences of period $2^n - 1$. We first give the following definition.

Definition 2.9. An output sequence of an LFSR of length n is called PN (pseudo-noise) sequence if its period is $2^n - 1$.

From Definition 2.9 of PN-sequences it can be seen that if an LFSR of length n does not run cyclically through all $2^n - 1$ non-zero states, it does not generate a PN-sequence. This can be stated as follows:

Theorem 2.10. *If an LFSR of length n generates a PN sequence, then it runs through $2^n - 1$ states. Hence all non-zero sequences of this LFSR are shifted versions of each other.*

Definition 2.11. Let $\pi(t)$ be a characteristic polynomial defined as in (2.3). Then

$$\Omega(\pi) = \left\{ \{x_i\} | \{x_i\} \text{ satisfies equation (2.1)} \right\}.$$

This means that $\Omega(\pi)$ is the set of the output sequences generated by $\pi(t)$. The group $\Omega(\pi)$ is also a vector space, as the following theorem states.

Theorem 2.12. *The generated set $\Omega(\pi)$ of an LFSR of length n with characteristic polynomial $\pi(t)$ is a binary vector space with dimension* $\dim(\Omega(\pi)) = n$.

Proof. The laws of a vector space are clearly satisfied since equation (2.1) is linear, but let us prove one of the rules as an example.

Take $\{x_i\} \in \Omega(\pi)$ and $\{y_i\} \in \Omega(\pi)$. For both of them equation (2.1) applies. So, we have that

$$x_{k+n} + y_{k+n} = \sum_{i=0}^{n-1} \alpha_i (x_{k+i} + y_{k+i}),\ k \geq 0,$$

from where

$$x_{k+n} + y_{k+n} \in \Omega(\pi).$$

Next we show that $\dim(\Omega(\pi)) = n$.

Since every sequence $\{x_i\} \in \Omega(\pi)$ is uniquely determined by the initial state $\{x_0, x_1, \ldots, x_{n-1}\}$, and these initial states span a vector space of dimension n, then $\dim(\Omega(\pi)) \leq n$.

On the other hand, the n different sequences corresponding to initial states

$$\underbrace{00\cdots0}_{i}1\underbrace{0\cdots00}_{n-i-1},\ i = 0, 1, \ldots, n-1$$

are linearly independent. Hence $\dim(\Omega(\pi)) \geq n$. □

Theorem 2.13. *Let $\{x_i\} \in \Omega(\pi)$, where π is as in equation* (2.3). *Let $S(t)$ be the generating function of $\{x_i\}$ given by equation* (2.4). *Then*

$$S(t) = \frac{\tau(t)}{\pi^*(t)}, \tag{2.5}$$

where

$$\tau(t) = \sum_{j=0}^{n-1} \left(\sum_{i=0}^{j} \alpha_{n-i} x_{n-i} \right) t^j$$

and

$$\pi^*(t) = t^n \pi(1/t) = \sum_{i=0}^{n} \alpha_{n-i} t^i$$

is the reciprocal of $\pi(t)$.

Proof.

$$S(t)\pi^*(t) = \left(\sum_{k=0}^{\infty} x_k t^k\right)\left(\sum_{l=0}^{n} \alpha_{n-l} t^l\right)$$

by the definitions of S and the reciprocal of π. We proceed by calculating the product

$$S(t)\pi^*(t) = \sum_{j=0}^{\infty}\left(\sum_{l=0}^{\min\{j,n\}} \alpha_{n-l} x_{j-l}\right) t^j.$$

This is then divided into two sums. The first sum contains the terms with $j < n$ and the second sum the terms with $j \geq n$.

$$S(t)\pi^*(t) = \sum_{j=0}^{n-1}\left(\sum_{l=0}^{j} \alpha_{n-l} x_{j-l}\right) t^j + \sum_{j\geq n}\left(\sum_{l=0}^{n} \alpha_{n-l} x_{j-l}\right) t^j.$$

Here the first sum is $\tau(t)$. Putting $n-l = i$ we get by (2.2) that all coefficients of the second sum

$$\sum_{j\geq n}\left(\sum_{l=0}^{n} \alpha_{n-l} x_{j-l}\right) t^j = \sum_{j\geq n}\left(\sum_{i=0}^{n} \alpha_i x_{(j-n)+i}\right) t^j$$

are zero by setting $k = j - n$ in equation 2.2.

So we have proved

$$S(t)\pi^*(t) = \tau(t).$$

□

Now we show that the representation given by Theorem 2.13 is unique given $\pi(t)$. Instead of writing $\{x_i\} \in \Omega(\pi)$ the notation $S(t) \in \Omega(\pi)$ is also used.

Theorem 2.14.

$$\Omega(\pi(t)) = \left\{ \frac{\tau(t)}{\pi^*(t)} \middle| \deg(\tau(t)) < n \right\}.$$

Proof. By Theorem 2.13 every $S(t) \in \Omega(\pi)$ can be written as

$$S(t) = \frac{\tau(t)}{\pi^*(t)},$$

where $\deg(\tau(t)) < n$. Hence,

$$\Omega(\pi) \subset \left\{ \frac{\tau(t)}{\pi^*(t)} \middle| \deg(\pi(t)) < n \right\}.$$

By Theorem 2.12, the set $\Omega(\pi)$ has 2^n elements. The number of polynomials of degree $n-1$ is at most 2^n so the number of elements of the set on the right hand side is at most 2^n. Hence the sets must be equal, and both have 2^n elements. □

Lemma 2.15. *Let $\{x_i\}$ be a periodic sequence with period p. Then p divides all other integers m with the property: $x_{k+m} = x_k$, for all $k \in N$.*

Proof. Assume the opposite. Then $m = pq + r$, where $0 < r < p$. On the other hand by the assumption, for all $k \in N : x_k = x_{k+m} = x_{k+pq+r} = x_{(k+r)+pq} = x_{k+r}$ and we have a contradiction, since p is the least integer with this property. □

Lemma 2.16. *Let $S(t)$ be the generating polynomial of the periodic sequence $\{x_i\}$ with period p. Then $S(t) \in \Omega(t^m+1)$ if and only if p divides m. Especially, $S(t) \in \Omega(t^p + 1)$.*

Proof. From the previous Lemma 2.15 we know that p divides m if and only if $\{x_i\}$ starts repeating itself after m bits, which happens if and only if $\{x_i\}$ can be generated using an LFSR with characteristic polynomial $t^m + 1$. But this means that $S(t) \in \Omega(t^m + 1)$. □

Lemma 2.17. *Let an irreducible polynomial $\pi(t)$ have order m. Let $\{x_i\} \in \Omega(\pi(t))$. Then the period of $\{x_i\}$ is m.*

Proof. Let $\{x_i\}$ have period p. It is first proved that p divides m. By Definition 2.4 there exists a polynomial ρ such that

$$t^m + 1 = \pi(t)\rho(t), \text{ with } \deg(\rho) = m - n. \tag{2.6}$$

By taking reciprocals of both sides of (2.6) we get

$$t^m + 1 = \pi^*(t)\rho^*(t). \tag{2.7}$$

By Theorem 2.13, there exists a polynomial $\tau(t)$ with $\deg \tau < n$ such that

$$S(t) = \frac{\tau(t)}{\pi^*(t)}.$$

Inserting (2.7) to equation above we get

$$S(t) = \frac{\tau(t)\rho^*(t)}{1 + t^m}.$$

Now $\deg(\rho) \geq \deg(\rho^*)$, so $\deg(\tau\rho^*) = \deg(\tau)+\deg(\rho^*) < m$. Then by Theorem 2.14, $S(t) \in \Omega(t^m + 1)$, and by Lemma 2.16, p divides m.

Next it will be proved that actually $m \leq p$, so $p = m$. By Lemma 2.16 and Theorem 2.14 we can write

$$S(t) = \frac{u(t)}{1 + t^p}, \tag{2.8}$$

where $\deg(u) < p$. Combine equation (2.8) and Theorem 2.14 to get

$$(1 + t^p)\tau(t) = u(t)\pi^*(t),$$

where $\deg(\tau) < m$. Take reciprocals of both sides of the previous equation, and get

$$(1 + t^p)\tau^*(t) = u^*(t)\pi(t).$$

By the assumption, $\pi(t)$ is irreducible, so it must divide either $1 + t^p$ or $\tau^*(t)$. As $\deg(\tau) = \deg(\tau^*) < n$, it follows that $\pi(t)$ must divide $1+t^p$. This is means that $m \leq p$, so $p = m$. ☐

Lemma 2.18. *Let $\pi(t)$ be a polynomial with $\deg(\pi) = n$. Let $\{x_i\} \in \Omega(\pi)$ be a PN-sequence. Then $\pi(t)$ is irreducible.*

Proof. The polynomial $\pi(t)$ can be written as $\pi(t) = \pi_1(t)\pi_2(t)$, where $\pi_1(t)$ is irreducible and $\deg(\pi_1(t)) = n_1 \leq n$. It is now proved that $n = n_1$, whence $\pi_2(t) = \text{constant} = 1$, so that $\pi(t) = \pi_1(t)$.

As $\pi_1(t)$ is irreducible, then its order m_1 divides $2^{n_1} - 1$ by Lemma 2.6. By Theorem 2.14 $\frac{1}{\pi_1^*(t)} \in \Omega(\pi_1)$ and so the period of $\frac{1}{\pi_1^*(t)}$ divides m_1 by the result of Lemma 2.16. This means that $2^{n_1} - 1$ is divided by the period of $\frac{1}{\pi_1^*(t)}$.

Taking reciprocals over $\pi(t)$, and dividing with $\pi^*(t)\pi_1(t)$ one gets by Theorem 2.14

$$\frac{1}{\pi_1^*(t)} = \frac{\pi_2^*(t)}{\pi^*(t)} \in \Omega(\pi). \tag{2.9}$$

From Theorem 2.10, and equation (2.9) it follows that $\frac{1}{\pi_1^*(t)}$ is a shift of $\{x_i\}$, and its order is $2^n - 1$. But as the period of $\frac{1}{\pi_1^*(t)}$ also divides $2^{n_1} - 1$, it must be that $2^n - 1 = 2^{n_1} - 1$, or $n = n_1$, which again means that $\pi(t) = \pi_1(t)$. ☐

Theorem 2.19. *An output sequence of an LFSR with characteristic polynomial π is a PN sequence, i.e., a sequence of maximal length $2^n - 1$, if and only if π is primitive.*

Proof. First assume that π is primitive. Then its period is $2^n - 1$ and it is irreducible. By Lemma 2.17, $\{x_i\}$ has also period $2^n - 1$ and so the LFSR generates a PN-sequence.

Then assume that the LFSR generates a PN-sequence. Then by Lemma 2.18 π is irreducible. Now Lemma 2.17 can be applied here, to get that the period of π is equal to the period of $\{x_i\}$ which as a PN-sequence has period $2^n - 1$, so π is primitive. ☐

2.3.1 Analyzing LFSRs

In this section we analyze a general LFSR with the aid of the Golomb's postulates and the Cryptographic axioms given in Section 2.1. The initial state is usually formed with the aid of a random key. Since the all-zero input-vector will produce nothing but 0's as output, it should be verified that the initial state differs from 0. There are some other limitations that depend on the generating polynomial.

Let us now analyze an LFSR that produces a PN-sequence with the aid of Golomb's random postulates.

G-1 The first rule is fulfilled, as by Theorem 2.10 each non-zero state occurs exactly once per period. The leftmost bit of each state will be the next output bit. So the number of ones per period is 2^{n-1} and the number of zeros per period is $2^{n-1} - 1$, as the all-zero state is removed. The result stays the same if the output bit is the second one in the LFSR and so forth. So the number of ones and the number of zeros are as equal per period as possible.

G-2 In an LFSR whose length is n there are 2^{n-k-2} states, whose leftmost $k+2$ bits are $011\ldots110$ respectively $100\ldots001$. So gaps and blocks of the length $k \leq n-2$ occur exactly 2^{n-k-2} times per period. When $k = n-1$ the state $011\ldots1$ occurs exactly once. Its successor is $11\ldots1$ whose successor in turn is $11\ldots10$. So there is no block of length $n-1$ and one block of length n. Similarly the block $100\ldots00$ is followed by $00\ldots01$. Hence, there is one gap of length $n-1$ and no gap of length n.

G-3 With $\{x_i\} \in \Omega(\pi)$ also $\{x_{i+k}\} \in \Omega(\pi)$. The linearity of $\Omega(\pi)$ implies that also $\{x_i + x_{i+k}\} \in \Omega(\pi)$. The number of agreements per period between $\{x_i\}$ and $\{x_{i+k}\}$ equals the number of zeros in one period of$\{x_i + x_{i+k}\}$, which is $2^{n-1} - 1$ by Theorem 2.10 and rule G-1. Similarly the number of disagreements is 2^{n-1}. So the out-of-phase autocorrelation is

$$\mathrm{AC}(k) = -\frac{1}{2^n - 1},$$

which is independent of k.

So LFSRs with primitive generating polynomials satisfy the Golomb's randomness postulates. Let us see next the cryptographic properties of an LFSR.

C-1 A LFSR of length n produces a sequence of the length $2^n - 1$, so in this sense the strength of LFSR is exponential. Rule C-1 is satisfied.

C-2 LFSR is easy to implement both in software and in hardware.

C-3 Equation (2.1) can be written in a matrix form

$$\begin{pmatrix} x_k & x_{k+1} & \cdots & x_{k+n-1} \\ x_{k+1} & x_{k+2} & \cdots & x_{k+n} \\ \vdots & \vdots & \ddots & \cdots \\ x_{k+n-1} & x_{k+n} & \cdots & x_{k+2n-2} \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{n-1} \end{pmatrix} = \begin{pmatrix} x_{k+n} \\ x_{k+n+1} \\ \vdots \\ x_{k+2n-1} \end{pmatrix}.$$

It follows that the knowledge of only $2n$ consecutive bits of the output sequence enables the cryptanalyst to determine the feedback coefficients α_i, $i = 0, \ldots, n-1$ uniquely, and hence obtain the knowledge of the whole $\{x_i\}$ sequence. The complexity of solving the matrix equation is at most of the $\mathcal{O}(\mathrm{n}^3)$, i.e., breaking the systems needs only polynomial time. So rule C-3 is not fulfilled.

On can deduce that although LFSRs are easy to implement, and they do have certain properties of pseudo-random sequences, they are not a good way to encrypt data as such. Their problem is due to linearity, as seen in C-3. They should be made nonlinear somehow. Several such methods are discussed in [Rue92].

In this thesis we study the combiner generators, which are constructed by combining the outputs of at least two LFSRs using a nonlinear Boolean function. However, such a construction is typically vulnerable to various forms of correlation attacks, also discussed in [Rue92]. In [Rue86] introduced a method for strengthening combiner generators against correlation attacks. He proposed to equip the combiner function with memory. This paradigm will be discussed in Chapter 4 after we have introduced the necessary mathematical tools in Chapter 3.

Chapter 3

Nonlinearity of Boolean functions

This chapter introduces the main mathematical tools and methods that are needed in the rest of the thesis, Walsh-transform and correlation. We also give a brief overview of correlation immunity properties of Boolean functions.

We start with a few basic definitions. A *Boolean function* is a binary-valued function of some finite number, say n, binary variables. We write $f : GF(2^n) \to GF(2)$, assuming that the field $GF(2^n)$ is equipped with a linear space structure and a basis. In this manner, each element $x \in GF(2^n)$ is set into one-to-one correspondence with a sequence $(x^1, x^2, \ldots, x^n)$ of n binary variables, $x^i \in GF(2)$ for $i = 1, 2, \ldots, n$.

Let us denote by $|\mathcal{A}|$ the number of the elements in set $\mathcal{A}$. A Boolean function is called *balanced*, if it takes the value one as many times as the value zero, or

$$|\{x \in GF(2^n) | f(x) = 1\}| = |\{x \in GF(2^n) | f(x) = 0\}|.$$

To make the distinction between integer-summation and summation modulo two we use from now on the notation "$\oplus$" and "$\bigoplus$" instead of "$+$" and "$\sum$", when referring to sum modulo two:

$$\bigoplus_{i=1}^{n} x^i = x^1 \oplus x^2 \oplus \cdots \oplus x^n,\ x^i \in GF(2).$$

The inner product in $GF(2)$, denoted by "$\cdot$", between two vectors $w, x \in GF(2^n)$ is defined as

$$w \cdot x = \bigoplus_{i=1}^{n} w^i x^i = w^1 x^1 \oplus w^2 x^2 \oplus \cdots \oplus w^n x^n.$$

The linear function $L_w(x)$ is then

$$\begin{aligned} &L_w : GF(2^n) \to GF(2) \\ &L_w(x) = w \cdot x = w^1 x^1 \oplus \cdots \oplus w^n x^n,\ w, x \in GF(2^n). \end{aligned}$$

The *Algebraic Normal Form* (ANF) of a Boolean function is its expression as a constant plus a sum of distinct products of its variables in $GF(2)$, i.e.

$$\begin{aligned} f(x^1, x^2, \ldots, x^n) = a_0 &\oplus a_1 x^1 \oplus \cdots \oplus a_n x^n \\ &\oplus a_{12} x^1 x^2 \oplus a_{13} x^1 x^3 \oplus \cdots \oplus a_{n-1,n} x^{n-1} x^n \\ &\oplus \cdots \oplus a_{12\ldots n} x^1 x^2 \ldots x^n. \end{aligned}$$

The *nonlinear order* of a Boolean function is the degree of its ANF polynomial.

3.1 Walsh-transform

The Walsh-transform is a type of discrete Fourier-transform of a Boolean function. It is integer-valued.

Definition 3.1. Let $f : GF(2^n) \to GF(2)$ be a Boolean function. Then the Walsh-transform $\mathcal{W}_f$ of f is defined as

$$\begin{aligned} &\mathcal{W}_f : GF(2^n) \to \mathbb{Z} \\ &\mathcal{W}_f(w) = \sum_{x \in GF(2^n)} (-1)^{f(x) \oplus w \cdot x} = \sum_{x \in GF(2^n)} (-1)^{f(x) \oplus L_w(x)}, \end{aligned} \tag{3.1}$$

where $w \in GF(2^n)$. The sum over $x \in GF(2^n)$ is an integer sum.

The next lemma gives a useful way to simplify certain equations of Walsh-transforms. The proof of the lemma is obvious.

Lemma 3.2. *Let $x, y \in GF(2^n)$. Then*

$$\sum_{w \in GF(2^n)} (-1)^{w \cdot x \oplus w \cdot y} = \sum_{w \in GF(2^n)} (-1)^{w \cdot (x \oplus y)} = \begin{cases} 2^n, & \text{when } x = y \\ 0, & \text{when } x \neq y. \end{cases} \tag{3.2}$$

Next we state Parseval's theorem. It is of fundamental importance when analyzing the cryptographical properties of some enciphering systems.

Theorem 3.3 (Parseval's theorem for Walsh-transforms).
Let $f : GF(2^n) \to GF(2)$ and let $\mathcal{W}_f$ be the Walsh-transform of f. Then

$$\sum_{w \in GF(2^n)} \mathcal{W}_f(w)^2 = 2^{2n}. \tag{3.3}$$

The sum above is the usual integer sum.

Proof. By Definition 3.1 of the Walsh-transform

$$\sum_{w \in GF(2^n)} \mathcal{W}_f(w)^2 = \sum_{w \in GF(2^n)} \left(\sum_{x \in GF(2^n)} (-1)^{f(x) \oplus w \cdot x} \right) \left(\sum_{y \in GF(2^n)} (-1)^{f(y) \oplus w \cdot y} \right).$$

The order of summation is changed. One can always apply the usual rules for these modulo two exponents and get

$$\sum_{w \in GF(2^n)} \mathcal{W}_f(w)^2 = \sum_{x,y} (-1)^{f(x) \oplus f(y)} \sum_{w} (-1)^{(x \oplus y) \cdot w}.$$

For the inner sum over w one can use equation (3.2) and get

$$\sum_{w \in GF(2^n)} \mathcal{W}_f(w)^2 = 2^n \sum_{x} (-1)^{f(x) \oplus f(x)} = 2^n 2^n = 2^{2n}.$$

The first equality is obtained by (3.2) and the second one by putting $y = x$ in the outer sum. □

3.2 Correlations

The concept of correlation will be used later in evaluating nonlinearity properties of Boolean functions. Correlation between Boolean functions has also an interpretation as the correlation between random variables [MS92].

Let us first define the correlation between Boolean functions.

Definition 3.4. The correlation between two Boolean functions $f : GF(2^n) \to GF(2)$ and $g : GF(2^n) \to GF(2)$ is

$$c(f, g) = 2^{-n}(|\{x | f(x) = g(x)\}| - |\{x | f(x) \neq g(x)\}|).$$

The definition is the same as in [MS92] and is also called *normalized correlation*. If needed, the notation $c_x(f(x), u \cdot x)$ shall be used to emphasize the variable with respect to which the correlation is taken.

The terms of the sum in the Definition 3.1 of Walsh-transformation are 1 for those x for which $f(x) = L_w(x)$. On the other hand, when $f(x) \neq L_w(x)$, then the value is -1. With the aid of this notion, one can then write equation (3.1) as

$$\mathcal{W}_f(w) = |\{x \in GF(2^n) | f(x) = L_w(x)\}| - |\{x \in GF(2^n) | f(x) \neq L_w(x)\}|.$$

Hence, we have the connection between the Walsh-transform and correlations. This is stated in the next theorem.

Theorem 3.5. *Let $f : GF(2^n) \to GF(2)$ and let L_w be a linear function $L_w(x) = w \cdot x$. Then, for the correlation $c(f, L_w)$ and for the Walsh-transform of f at point w, it applies that*

$$\mathcal{W}_f(w) = 2^n c(f, L_w). \tag{3.4}$$

When the Boolean function depends only on a small number of variables, then it is possible to calculate the correlations by hand. At some point, however, more computing power is needed. There are some practical algorithms to compute Walsh-transforms, c.f. [Mas93] for example, with complexity $\mathcal{O}(n2^n)$.

In Section 3.1 Parseval's equation was proved. Taking equation (3.4) into consideration, one gets

$$\sum_w c(f, L_w)^2 = \sum_w \left(2^{-n}\mathcal{W}_f(w)\right)^2 = 2^{-2n}2^{2n} = 1.$$

So we have another version of Parseval's theorem:

Theorem 3.6 (Parseval's theorem for correlations). *Let f be a Boolean function $f : GF(2^n) \to GF(2)$. Then for the sums of the squares of the correlations between f and linear combinations of its input variables the following equality holds:*

$$\sum_w c(f, L_w)^2 = 1. \tag{3.5}$$

Especially, for every Boolean function f, there is always some linear function L, such that the correlation between these functions satisfies

$$|c(f, L)| \geq 2^{-\frac{n}{2}},$$

and particularly, it is non-zero. This has some important cryptanalytical consequences.

Next some important correlation theorems shall be presented and proved.

Theorem 3.7. *Given functions $f : GF(2^n) \times GF(2^k) \to GF(2)$ and $g : GF(2^m) \to GF(2^k)$ we set*

$$h(x, y) = f(x, g(y)),\ x \in GF(2^n),\ y \in GF(2^m).$$

Then, for all $u \in GF(2^n)$, $v \in GF(2^m)$,

$$\begin{aligned} &c_{x,y}(h(x, y), u \cdot x \oplus v \cdot y) \\ &= \sum_{w \in GF(2^k)} c_{x,z}(f(x, z), u \cdot x \oplus w \cdot z)c_y(w \cdot g(y), v \cdot y). \end{aligned} \tag{3.6}$$

Proof. We start from the right hand side of (3.6). We first express the correlations with the aid of Walsh-transform as in equation (3.4):

$$\begin{aligned} &\sum_{w \in GF(2^k)} c_{x,z}(f(x, z), u \cdot x \oplus w \cdot z)c_y(w \cdot g(y), v \cdot y) \\ &= \sum_{w \in GF(2^k)} \left(2^{-n-k} \sum_{(x,z)} (-1)^{f(x,z)\oplus u\cdot x \oplus w \cdot z}\right) \left(2^{-m} \sum_y (-1)^{w\cdot g(y) \oplus v \cdot y}\right). \end{aligned}$$

Next we change the order of summation in such a way that everything not depending on w will be in the outermost summation.

$$\begin{aligned}&\sum_{w\in GF(2^k)} c_{x,z}(f(x,z), u\cdot x\oplus w\cdot z)c_y(w\cdot g(y), v\cdot y)\\&= 2^{-n-m-k}\sum_{(x,y,z)}(-1)^{f(x,z)\oplus u\cdot x\oplus v\cdot y}\sum_{w}(-1)^{w\cdot z\oplus w\cdot g(y)}.\end{aligned}$$

For the inner summation we use Lemma 3.2, by which only the terms with $z = g(y)$ are non-zero:

$$\begin{aligned}&\sum_{w\in GF(2^k)} c_{x,z}(f(x,z), u\cdot x\oplus w\cdot z)c_y(w\cdot g(y), v\cdot y)\\&= 2^{-n-m}\sum_{(x,y)}(-1)^{f(x,g(y))\oplus u\cdot x\oplus v\cdot y}\\&= 2^{-n-m}\sum_{(x,y)}(-1)^{h(x,y)\oplus u\cdot x\oplus v\cdot y}\\&= c_{x,y}(h(x,y), u\cdot x\oplus v\cdot y).\end{aligned}$$

This applies for all $u \in GF(2^n)$ and $v \in GF(2^m)$. □

As a consequence of the previous theorem we get the following corollary. It is also a generalized version of the Lemma 1 in [MS92].

Corollary 3.8. *Let f, g and h be as in Theorem 3.7 with $k = 1$. Then the correlation $c(h, L_{(u,v)})$ of a Boolean function h and a linear function $L_{(u,v)}$ can be stated as*

$$c_{x,y}(h(x,y), u\cdot x\oplus v\cdot y) = c_{x,z}(f(x,z), u\cdot x\oplus z)c_y(g(y), v\cdot y). \tag{3.7}$$

The next theorem deals with the situation, where two Boolean functions have common input variables.

Theorem 3.9. *Let $f : GF(2^n)\times GF(2^k) \to GF(2)$ and $g : GF(2^m)\times GF(2^k) \to GF(2)$ be Boolean functions. Then, for all $u, v \in GF(2^n)$,*

$$\begin{aligned}&c_{x,y,\xi}(f(x,y)\oplus g(\xi,y), u\cdot x\oplus v\cdot\xi)\\=&\sum_{w\in GF(2^k)} c_{x,y}(f(x,y), u\cdot x\oplus w\cdot y)c_{\xi,y}(g(\xi,y), v\cdot\xi\oplus w\cdot y).\end{aligned}$$

Proof. We proceed as in the proof of Theorem 3.7. First we write the right hand side in the terms of Walsh-transform as follows:

$$\sum_{w\in GF(2^k)}\left(2^{-n-k}\sum_{(x,y)}(-1)^{f(x,y)\oplus u\cdot x\oplus w\cdot y}\right)\left(2^{-m-k}\sum_{(\xi,y')}(-1)^{f(\xi,y')\oplus w\cdot y'\oplus v\cdot z}\right).$$

Next the order of summation is changed, and then Lemma 3.2 is applied:

$$\begin{aligned}
&2^{-n-m-2k} \sum_{(x,y,y',z)} (-1)^{f(x,y)\oplus g(\xi,y')\oplus u\cdot x\oplus v\cdot\xi} \sum_{w} (-1)^{w\cdot y\oplus w\cdot y'} \\
&= 2^{-n-m-k} \sum_{(x,y,z)} (-1)^{f(x,y)\oplus g(\xi,y)\oplus u\cdot x\oplus v\cdot\xi} \\
&= c_{x,y,\xi}(f(x,y)\oplus g(\xi,y), u\cdot x\oplus v\cdot\xi).
\end{aligned}$$

□

In the case, where the two functions f and g, and the two linear combinations u and w are the same, we have the following corollary.

Corollary 3.10. *Let $f : GF(2^n)\times GF(2^k)\rightarrow GF(2)$ be a Boolean function. Then, for all $u\in GF(2^n)$,*

$$\begin{aligned}
&c_{x,y,\xi}(f(x,y)\oplus f(\xi,y), u\cdot(x\oplus\xi)) \\
&= \sum_{w\in GF(2^k)} c_{x,y}(f(x,y), u\cdot x\oplus w\cdot y)^2. \qquad (3.8)
\end{aligned}$$

3.3 Correlation immunity

In cryptanalysis the concept of linearity is often found to be useful. Linear systems are easy to break as shown in Section 2.3.1. Let us now briefly describe the Divide and Conquer attack, which can be used to attack systems with nonlinear combination function.

Assume that the system consist of n LFSRs, whose output is denoted by the vector $x = (x_1, \ldots x_n)$, and a combination function $f : GF(2^n)\rightarrow GF(2)$. Assume further that there is non-zero correlation between a subset of the LFSRs, say the m first, and the output function:

$$c(f, x^1\oplus x^2\oplus\cdots\oplus x^m) = \rho\neq 0.$$

Then the outputs $x^{m+1},\ldots,x^n$ of the $n-m$ LFSRs can be ignored, and the analyst can concentrate on the m first LFSRs thus reducing the amount of trials when searching for the initial states. The number of output bits needed for the attack is then proportional to $1/\rho^2$. [Gol96]

Let us next consider $Z = f(V)$ and $Y = L_w(V)$, where $V = (V^1,\ldots,V^n)\in GF(2^n)$ and $Y, Z\in GF(2)$ are binary random variables. Then it can be seen that Z and Y are statistically independent if and only if the correlation $c(f, L_w)$ vanishes, or, what is equivalent, $\mathcal{W}_f(w) = 0$.

Definition 3.11. A Boolean function $f : GF(2^n)\rightarrow GF(2)$ is mth order correlation-immune if $Z = f(V)$ is statistically independent of any subset $V^{i_1}, V^{i_2},\ldots,V^{i_m}$ of m input variables.

The *Hamming weight* $W_H(w)$ of a binary vector $w \in GF(2^n)$ is the number of 1's in w.

It has been shown in [XM88] that mth order correlation-immunity is equivalent to the condition that $Z = f(V)$ is statistically independent of the linear combination $w \cdot V$, for every $w \in GF(2^n)$ such that $1 \leq W_H(w) \leq m$. So, we have the next theorem.

Theorem 3.12 (Xiao and Massey). *A Boolean function f is mth order correlation-immune if and only if*

$$\mathcal{W}_f(w) = 0, \textit{ for all } w \textit{ such that } 1 \leq W_h(w) \leq m.$$

Maximum order correlation-immunity occurs, when f is $(n-1)$th order correlation-immune. This means that $f : GF(2^n) \rightarrow GF(2)$ is an XOR (modulo constant) of its inputs. The knowledge of less than n input variables gives no information on the output value of f. This makes f immune to Divide and Conquer attack.

Parseval's theorem 3.6 stated that the output function f has always non-zero correlation to some combination(s) of its input values. If it happened that the only non-zero correlation was between f and the XOR of all of its input values, then f would of course be XOR. This means that combination generators with a simple combination function is either linear or can always be attacked by the Divide and Conquer attack.

But using XOR as combination function is not a good choice, since then the system can always be reduced to a single LFSR whose generating polynomial is the product of the original polynomials [vT88]. If the generating polynomials have common divisors, the generating polynomial of the resulting system can be further reduced to the least common multiple of the original polynomial.

To make the system resist the attack, Rueppel came up with the idea to equip the system with memory [Rue86]. These combination generators with memory are studied in the next chapter.

Chapter 4

Combination generators with memory

In Chapter 2 the LFSRs were introduced. At the end of Section 2.3.1 it was noted that LFSR is not cryptographically secure, since it is linear. Hence methods for introducing nonlinearity are needed.

Several ways of making an LFSR system nonlinear are proposed for example in [Rue92]. One can clock certain parts of the generator irregularly in order to achieve nonlinear effects. Other kind of nonlinear generators are for example a threshold generator and a knapsack generator. In [Rue86] the use of memory was proposed to achieve nonlinearity. This method is also the main object of study in this thesis.

4.1 Defining combination generators

The combination function of the system will be a combination of XOR and some nonlinear function. This way the system is as correlation immune as possible, and at the same time it resists the attacks on a purely linear systems.

In this work we present detailed analysis of two specific combination generators with memory. The first one is the summation generator [Rue86] and the second one is a more complex variant of the basic idea of the summation generator. A general form of this type of generators is depicted in Figure 4.1. The system consists of n LFSRs, a memory sized $(r + 1) \times m$, and three functions T, f_0 and f_1, of which the first two are linear and f_1 is nonlinear.

The memory – or the *carry* – consists of $r + 1$ blocks, with m bits each. The memory is implemented as a shift register. At each time step t, all blocks are shifted one block to the right, the rightmost block c_{t-r} is discarded, and the leftmost block is updated by the output c_{t+1} of function $T \circ f_1$.

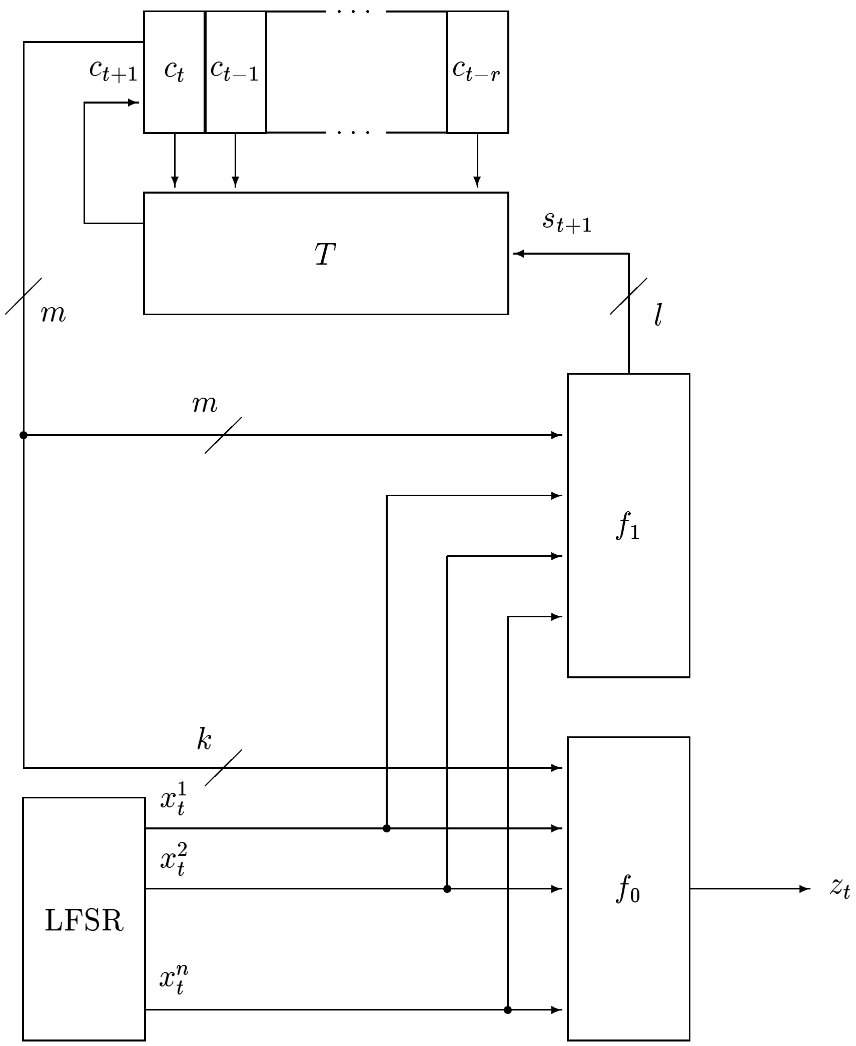

Figure 4.1: A combination generator with memory.

The function f_0 produces an output sequence $z_1, z_2, \ldots$, where $z_t \in GF(2)$. It takes x_t as an input vector, as well as $k \leq m$ bits of the current contents of the memory, computes the modulo two sum of these input bits and outputs z_t.

Nonlinearity is created to the system by function $f_1 : GF(2^{n+m}) \to GF(2^l)$. It takes x_t as input, as well as the latest memory update vector c_t. The output is an l-bit vector s_{t+1}.

The function T is a linear part of the memory update function. It takes all the $m \times (r+1)$ memory bits and s_{t+1} as inputs, mixes them, and produces an m-bit vector c_{t+1} to be put in the memory.

The whole encryption procedure then consists of the following steps:

1. Put t=1
2. Initialize the LFSRs with the secret key and perhaps some other information such as addresses, clocking bits etc.
3. Initialize the memory, depending on the choice of f_1 it could be defined to be zero or some arbitrary number or the value the memory has already.
4. for t=1 to N:
   (a) Output x_t
   (b) Calculate $z_t = f_0(x_t, c_t)$
   (c) Calculate the encrypted message bit $b_t = z_t \oplus a_t$, were a_t is the corresponding plaintext bit.
   (d) Calculate $s_{t+1} = f_1(x_t, c_t)$
   (e) Calculate $c_{t+1} = T(s_{t+1}, c_t)$
   (f) Put $c_{t-r+i} = c_{t-r+i+1}$, $i = 0, 1, \ldots, r$
5. Stop, if the message has been encrypted, otherwise goto Step 2.

The constant N is the maximum number of plaintext bits that can be encrypted before the generator must be re-initialized. The limit N is set, because strings of output bits of sufficient length would enable various kinds of attacks based on statistical analysis. One such attack is described later in this thesis.

The decryption is done similarly to encryption. The receiver has access to the same secret key and other necessary initialization data. The only difference is in the Step 4 (c), where the calculation is $a_t = z_t \oplus b_t$, with b_t being the received encrypted bit.

The cryptanalytical attacks against a combiner generator aim at recovering an initial state of the LFSRs. If there were a correlation between the output of f_0 and a linear combination of x_t's components, such that part of x_t could be dropped out, then the Divide and Conquer attack could be performed. Being defined as XOR function f_0 is maximum order correlation-immune. This means that such correlations cannot be found. However, by including more time steps in the analysis something similar could be done, given sufficiently many output bits. Section 4.3 presents such an attack.

In the next section we shall consider the summation generator, which is a combination generator with one bit of memory. We shall follow the analysis presented by Meier and Staffelbach [MS92]. We shall see then that combiners with one-bit memory and two LFSRs are easy to break.

4.2 A combination generator with one-bit memory

Let us have n LFSRs denoted LFSR_i, $i = 1, 2, \ldots, n$, which produce the output $x_t = (x_t^1, \ldots, x_t^n)$ at time t. Let the length of $LFSR_i$ be L_i. We assume the generating polynomials to be primitive, whence their periods are $p_i = 2^{L_i} - 1$, and also that the initial states are non-zero.

The generator has one bit of memory, denoted by c_t. The output of the system is the sequence $z_t \in GF(2)$, $t = 0, 1, 2, \ldots$ The system is defined by the recursive equations

$$z_t = \bigoplus_{i=1}^{n} x_t^i \oplus c_{t-1}$$
$$c_t = f(x_t, c_{t-1}),$$

where f is a nonlinear function. Here the calculations shall start on time $t = 0$, so we have to initialize the value c_{-1} to some arbitrary value, say $c_{-1} = 0$. The reader is also encouraged to consult Section 3.2 for more background.

The example is shown in Figure 4.2.

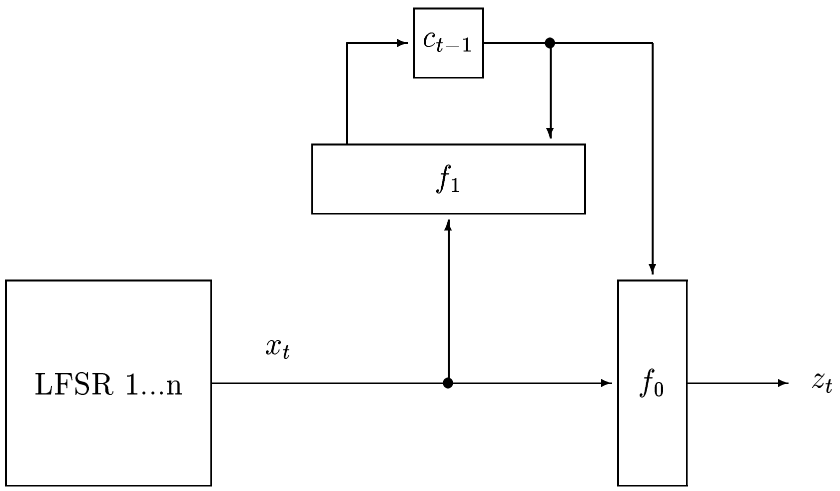

Figure 4.2: A combination generator with one-bit memory.

It is assumed that the sequences $\{x_t^i\}$, $i = 1, 2, \ldots, n$, are *independent and uniformly distributed* sequences of binary random variables. This is not exactly the case, since when $t = 2^{L_i} - 1$, $i = 1, 2, \ldots, n$, the sequence produced by LFSR_i will start repeating itself. But since the period is exponential, we can make these local assumptions. Then we have that the sequence $\{z_t\}$ is statistically independent of each x_t^i.

We shall also assume that the function f is balanced, which means that

$$|\{(x_t, c_t) : f(x_t, c_t) = 0\}| = |\{(x_t, c_t) : f(x_t, c_t) = 1\}|.$$

Further we assume that the carry bits c_t are balanced and statistically independent of each other. Since c_t depends on all vectors x_k, $k = -1, 0, \ldots, t-1$, z_t becomes statistically dependent on certain combinations of x_k, for $k \leq t$. This is stated more accurately in Theorem 4.1. It is a generalization of Theorem 1 in [MS92]. Before proving it, we define some notation.

Let us denote by X_i, $i \leq t$, the vector $X_i = (x_t, x_{t-1}, \ldots, x_{t-i})$. Define further $u_t = (u_t^1, \ldots, u_t^n)$ and $U_t = (u_t, \ldots, u_{t-i})$. The memory consist of one bit, $c_t \in GF(2)$. Assume that the initial value of the memory bit c_{-1} is known.

In Theorem 4.1 it will be proved that z_t is correlated (with a non-zero correlation coefficient) to some combinations of X_i, $1 \leq i \leq t$ of the form

$$U_i \cdot X_i = \bigoplus_{k=t-i}^{t} u_k \cdot x_k,\ u_k, x_k \in GF(2^n), \tag{4.1}$$

where $u_t = (1, \ldots, 1)$ and $u_{t-i} \neq 0$. There are $N = N(i) = (2^n - 1)^i$ such combinations.

Let us fix i and denote by ρ_h, $h = 1, 2, \ldots, N$, the correlation coefficient for the correlation between the combination $U_i \cdot X_i$ and z_t. In the case of Parseval's theorem 3.6, the sum of the squares of the correlation coefficients equals 1. The next theorem will give a similar result for correlation coefficients ρ_h. It follows, in particular, that $\rho_h \neq 0$ at least for one $h = 1, 2, \ldots, N$, i.e., z_t is correlated to some combinations of X_i.

Theorem 4.1. *Assume the definitions above for $x_t, z_t, c_t, X_i, u_t, U_i$ and ρ_h. Then*

$$\sum_{h=1}^{N} \rho_h^2 = 1 - (1-d)^i \neq 0, \tag{4.2}$$

where

$$\begin{aligned} d &= \sum_{u_{t-1} \in GF(2^n)} c(c_{t-1}, u_{t-1} \cdot x_{t-1})^2 \\ &= \sum_{u_{t-k} \in GF(2^n)} c(c_{t-k}, u_{t-k} \cdot x_{t-k})^2, k = 1, ..., i, \end{aligned}$$

whence z_t has non-zero correlation to some $U_i \cdot X_i$, unless $d = 0$.

If $d \neq 0$, the obvious implication of this theorem is that the sum of the squares of the correlations goes to 1, which is analogue to Parseval's theorem.

If $d = 0$, then all the correlations $c(c_{t-k}, u_{t-k} \cdot x_{t-k})$ vanish. This means that the memory does not depend on the LFSRs output bits and vice versa. Hence, the memory has no affect on the output keysequence and the system can be regarded to be memoryless.

Proof. By the definition of ρ_h we have

$$\sum_{h=1}^{N} \rho_h^2 = \sum_{k=1}^{i} \sum_{U_k} c(z_t, U_k \cdot X_k)^2. \tag{4.3}$$

Let

$$e = \sum_{u_{t-k}} c(c_{t-k}, u_{t-k} \cdot x_{t-k} \oplus c_{t-k-1})^2.$$

We note that the value of e does not depend on k, and $e + d = 1$ by Parseval's theorem.

Next it will be shown that

$$\sum_{U_k} c(z_t, u_k \cdot x_k \oplus c_{t-k-1})^2 = e^k. \tag{4.4}$$

The proof is by induction. First, assume $k = 1$. Then

$$\begin{aligned}
&\sum_{U_1} c(z_t, U_1 \cdot X_1 \oplus c_{t-2})^2 \\
&= \sum_{u_{t-1}} c(z_t, u_t \cdot x_t \oplus c_{t-1})^2 c(c_{t-1}, u_{t-1} \cdot x_{t-1} \oplus c_{t-2})^2 \\
&= e,
\end{aligned}$$

where the first equality is obtained by (3.7) and the second follows from the fact that

$$c(z_t, u_t \cdot x_t \oplus c_{t-1})^2 = c(z_t, \bigoplus_{i=1}^{n} x_t^i \oplus c_{t-1})^2 = 1.$$

Hence, (4.4) holds for $k = 1$. Assume next that (4.4) applies for k and prove that it also applies for $k + 1$. Here we use (3.7) again, now for $c = c_{t-k-1}$:

$$\begin{aligned}
&\sum_{U_{k+1}} c(z_t, U_{k+1} \cdot X_{k+1} \oplus c_{t-k-2})^2 \\
&= \sum_{U_k} c(z_t, U_k \cdot X_k \oplus c_{t-k-1})^2 \sum_{u_{t-k-1}} c(c_{t-k-1}, u_{t-k-1} \cdot x_{t-k-1} \oplus c_{t-k-2})^2.
\end{aligned}$$

Here the first of the terms of the product is e^k by induction hypothesis and the second one equals e, because e was independent of k. Hence (4.4) applies for $k + 1$, and (4.4) is true.

Having obtained (4.4) , the calculation of the sum $\sum_{U_k} c(z_t, U_k \cdot X_k)^2$ gives

$$\begin{aligned}
&\sum_{U_k} c(z_t, U_k \cdot X_k)^2 \\
&= \sum_{U_{k-1}} \sum_{u_{t-k}} c(z_t, U_{k-1} \cdot X_{k-1} \oplus c_{t-k})^2 c(c_{t-k}, u_{t-k} \cdot x_{t-k})^2 \qquad (4.5) \\
&= e^{k-1} d,
\end{aligned}$$

where the first equality follows from (3.7) by setting $c = c_{t-k}$.

Now (4.5) can be inserted into (4.3) and by writing $e = d - 1$, one gets

$$\sum_{h=1}^{N} \rho_h^2 = \sum_{k=1}^{i} d(1-d)^k = 1 - (1-d)^i,$$

and the claim is proved. □

In the next section we discuss an example of a combiner with memory and with two LFSRs. The example presentation follows the paper [MS92], but is self-contained.

4.2.1 The summation generator

The summation generator computes the integer sum of the output bits of two LFSRs, outputs the least significant bit, and puts the more significant bit, the carry, into the memory. Hence the summation generator is a special case of the general type of generators depicted in Figure 4.1. The recursive equations are now as follows:

$$\begin{aligned} z_t &= x_t^1 \oplus x_t^2 \oplus c_{t-1} \\ c_t &= f(x_t, c_{t-1}) = x_t^1 x_t^2 \oplus x_t^1 c_{t-1} \oplus x_t^2 c_{t-1}. \end{aligned}$$

We can now apply Theorem 4.1, so we must first calculate the value of

$$d = \sum_{u_{t-1} \in GF(2^2)} c(c_{t-1}, u_{t-1} \cdot x_{t-1})^2.$$

Let us first calculate the correlation $c(c_{t-1}, x_{t-1}^1) = c(c_{t-1}, x_{t-1}^2)$:

$$\begin{aligned} &c(c_{t-1}, x_{t-1}^1) \\ &= \frac{|\{(x_{t-1}, c_{t-2}) | c_{t-1}(x_{t-1}, c_{t-2}) = x_{t-1}^1\}| - |\{(x_{t-1}, c_{t-2}) | c_{t-1}(x_{t-1}, c_{t-2}) \neq x_{t-1}^1\}|}{2^3} \\ &= \frac{6-2}{8} = 0,5. \end{aligned}$$

Similarly, the correlation $c(c_{t-1}, x_{t-1}^1 \oplus x_{t-1}^2) = 0$. Hence, $d = 0,5^2 + 0,5^2 = 0,5 \neq 0$. By Theorem 4.1 we have

$$\sum_{h=1}^{N} \rho_h^2 = 1 - \frac{1}{2^i},$$

where ρ_h, $h = 1, 2, \ldots, N$ are the correlation coefficients and N is the number of combinations $U_i \cdot X_i$, $i = 0, 1, \ldots, t$. An example of a correlation between z_t and $U_i \cdot X_i$ is

$$\rho = c(z_t, x_t^1 \oplus x_t^2 \oplus x_{t-1}^1),$$

which can be calculated by Corollary 3.8 to be

$$\rho = c(z_t, x_t^1 \oplus x_t^2 \oplus c_{t-1})c(c_{t-1}, x_{t-1}^1).$$

Since $c(z_t, x_t^1 \oplus x_t^2 \oplus c_{t-1}) = 1$ and $c(c_{t-1}, x_{t-1}^1) = 0,5$, we have $\rho = 0,5$.

Next we shall show how the summation generator is broken according to [MS92].

Attack on one-bit memory system

The change of the carry bit can be seen as a Markov-process. The probability for the carry to take the value 0 at time t is denoted with $q_t(0)$, and the probability for value 1 is written as $q_t(1)$, respectively. The analysis proceeds by writing the Markov-process in matrix form and then assuming that there is a sequence of zero (or one) output digits. After this, a linear equation for solving the initial bits is formed. This equation gives the correct solution with some probability. Solving several such equation gives the right answer (with a probability going to one). This procedure proves to be faster than the so called algorithm A presented in [Rue92].

Denote $I_t = x_t^1 + x_t^2$, where "+" stands for an integer sum. The probabilities for different values of I_t are $P(I_t = 0) = 0,25, P(I_t = 1) = 0,5$ and $P(I_t = 2) = 0,25$. Now, the carry bit c_t can be written as a function of I_t and c_{t-1}. These values are shown in Table 4.1. The boxed figures are those for which z_t has the value zero.

$c_{t-1} \backslash I_t$	0	1	2
0	$\boxed{0}$	0	$\boxed{1}$
1	0	$\boxed{1}$	1

Table 4.1: The values of c_t as functions of I_t and c_{t-1}.

The Markov-process can be written as $q_t = Aq_{t-1}$, where A is the transition matrix

$$A = \begin{pmatrix} \frac{3}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{3}{4} \end{pmatrix},$$

and $q_t = (q_t(0), q_t(1))^T$. Let the initial distribution of the system be $q_0 = (q_0(0), q_0(1))^T$. Then the sth state is $q_s = A^s q_0$. Using diagonalisation, we write A as $A = S^{-1}DS$, where

$$D = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{2} \end{pmatrix} \quad \text{and} \quad S = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}.$$

Now $A^s = S^{-1}D^sS$ and

$$D^s = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{2^s} \end{pmatrix},$$

so that

$$\begin{aligned} q_s(0) &= \frac{1}{2} + \frac{1}{2^{s+1}}(q_0(0) - q_0(1)) \\ q_s(1) &= \frac{1}{2} - \frac{1}{2^{s+1}}(q_0(0) - q_0(1)), \end{aligned}$$

since $q_0(0)+q_0(1) = 1$. Hence, $\lim_{s\to\infty} q_s = (\frac{1}{2}, \frac{1}{2})$, and the probability distribution tends to be uniform. This proves that, independently of how the initial values were chosen, the system terminates in a balanced state.

To break the system, we try another approach and assume that we have a known output $z_t = 0$. Then only the boxed values in Table 4.1 are to be considered. We see that then the probability for c_t to have value 0 when $c_{t-1} = 0$ is 0,5, whereas c_t can only have value 1 if $c_{t-1} = 1$. The new Markov-process is then

$$q_t = \begin{pmatrix} \frac{1}{2} & 0 \\ \frac{1}{2} & 1 \end{pmatrix} q_{t-1}.$$

Assume next that $z_{t+1} = z_{t+2} = \cdots = z_{t+s} = 0$. Then $q_{t+s} = A^s q_t$, where A^s can again be calculated by diagonalisation to be

$$A^s = \begin{pmatrix} \frac{1}{2^s} & 0 \\ 1 - \frac{1}{2^s} & 1 \end{pmatrix}.$$

Then

$$q_{t+s}(0) = \frac{1}{2^s} q_0(0) \leq \frac{1}{2^s}$$

and

$$q_{t+s}(1) = 1 - q_{s+t}(0) \geq 1 - \frac{1}{2^s}.$$

Take $1 \leq r \leq s$. Then $P(c_{t+s} = c_{t+s-1} = \cdots = c_{t+r+1} = 1 | c_{t+r} = 1) = 1$. We also have that the probability for $c_{t+r} = 1$ is at least $1 - 2^{-r}$. Hence,

$$P(c_{t+s} = \cdots = c_{t+r} = 1) \geq 1 - 2^{-r}. \tag{4.6}$$

In the same way, if there are s consecutive output bits $z_{t+1} = \ldots z_{t+s} = 1$, then

$$P(c_{t+s} = \cdots = c_{t+r} = 0) \geq 1 - 2^{-r} \tag{4.7}$$

for $1 \leq r \leq s$. Based on these calculations, we can state the following theorem for calculating the initial bits of the two LFSRs.

Theorem 4.2. *Assume that the output of a two LFSR combination generator satisfies*

$$z_{t+1} = z_{t+2} = \cdots = z_{t+s} = b,\ z_{t+s+1} = 1 \oplus b,\ b \in \{0, 1\}.$$

Then for every r, $1 \leq r \leq s$ the $s - r + 2$ equations

$$\begin{aligned}
z_{t+r+1} &= x^1_{t+r+1} \oplus x^2_{t+r+1} = 1, \\
z_{t+r+2} &= x^1_{t+r+2} \oplus x^2_{t+r+2} = 1, \\
&\vdots \\
z_{t+s+1} &= x^1_{t+s+1} \oplus x^2_{t+s+1} = 0, \\
z_{t+r+2} &= x^1_{t+r+2} \oplus x^2_{t+r+2} \oplus x^1_{t+s+1},
\end{aligned}$$

are simultaneously satisfied with probability at least $1 - 2^{-r}$.

Proof. The start of the proof has already been done in the derivation of equations (4.6) and (4.7). We shall only prove the case $b = 0$, since the other case is similar.

Since $z_{t+s} = 0$ and $z_{t+s+1} = 1$, we have that $P(c_{t+s} = 1) \geq 1 - 2^{-r}$ by equation (4.6). With this probability

$$1 = z_{t+s+1} \equiv I_{t+s+1} + 1 \pmod 2,$$

whence $I_{t+s+1} \equiv 0 \pmod 2$. If $I_{t+s+1} = 0$, then $c_{t+s+1} = 0 = x^1_{t+s+1} = x^2_{t+s+1}$. If $I_{t+s+1} = 2$, then $c_{t+s+1} = 1 = x^1_{t+s+1} = x^2_{t+s+1}$. Then, with probability $1 - 2^{-r}$

$$z_{t+s+2} = x^1_{t+s+2} \oplus x^2_{t+s+2} \oplus x^1_{t+s+1}.$$

□

Note that the equations are fulfilled simultaneously with this probability.

The attack can be exploited when a run of s consecutive output bits, either ones or zeros, has been observed. Then, after a desired reliability $\rho = 1/2^r$ has been chosen, one can have a set of $s - r + 2$ linear equations that are satisfied with probability greater than or equal to $1 - 2^{-r}$.

Let L be the length of the known output sequence and k be the number of initial bits, i.e., the sum of the lengths of the LFSRs. We look for runs of at least s ones or zeros. Denote by d the number of equations in a block that is, $d = s - r + 2$. Assuming there are n such blocks we then have dn equations. It must be that $nd > k$ for the initial state to be solvable. Assuming this, we can write for some $\alpha > 1 : nd = \alpha k$.

There is of course the chance that part of these equations are wrong, i.e., they give an incorrect answer. To solve all k initial bits correctly, we must have $m = \left\lceil \frac{k}{d} \right\rceil \approx \frac{n}{\alpha}$ correct equation blocks. To find this, we use the following algorithm:

1. Choose m of the n available blocks and solve the system for the k unknowns.

2. Test all the possible solutions of the Step 1. If a solution is correct, stop, otherwise go to Step 1.

To assess the usefulness of this algorithm, we calculate its complexity. Denote by M the number of the trials needed to solve the problem. Now M is the reciprocal of the probability q that, by sampling without replacement, m randomly chosen blocks are correct. The probability for a block being incorrect is $\rho \leq 2^{-r}$, so the number of incorrect blocks is approximately ρn. We assume for simplicity that ρn is an integer. Then

$$q = \left(1 - \frac{\rho n}{n}\right)\left(1 - \frac{\rho n}{n-1}\right) \cdots \left(1 - \frac{\rho n}{n-(m-1)}\right).$$

We can then estimate q with

$$q > \left(1 - \frac{\rho n}{n-m}\right)^m \approx \left(1 - \frac{\alpha 2^{-r}}{\alpha - 1}\right)^m.$$

In the paper of Meier and Staffelbach [MS92] a realistic example with two LFSRs, each of length about 200, was studied. It was assumed that the attacker has $N = 50000$ bits of the keys stream sequence.

By the properties of LFSRs, see Section 2.3.1, we know that on average there are $n \approx \frac{N}{2^s}$ runs of length at least s. Choosing $s = 7$ and $r = 4$, we then have $n \approx 390$, the block length $d = 5$ and the probability for a block being incorrect is quite small, $\rho = 2^{-4} = \frac{1}{16}$. Then $m = k/d = 80$ blocks is needed to solve the problem. From above we see that $q > 0,001427$ and $M < 699$. Hence, less than 700 trials are sufficient. This is quite a small value in a typical situation. Hence we can regard the system as broken.

It is also stated in [MS92] that this attack is not plausible if there are more than two LFSRs in use. Hence it is concluded that one should use several LFSRs instead of two.

4.3 Ultimate Divide and Conquer

In this section it is shown that Divide and Conquer attack becomes possible if the length of the given keystream is longer that the period of the shortest LFSR used in the key stream generation. Assume that there is a relation with a non-zero correlation ρ between a linear combination of the shift register output bits

$$U_d \cdot X_d = \bigoplus_{i=0}^{d} u_{t-i} \cdot x_{t-i},$$

(see page 53 of notations) and a combination of the key stream bits

$$w_0 z_t \oplus w_1 z_{t-1} \oplus \ldots \oplus w_d z_{t-d},$$

over a number of $d+1$ time steps. Choose k out of n LFSRs, $1 \leq k \leq n$, and let p be the joint period of $\mathrm{LFSR}_1, \ldots \mathrm{LFSR}_k$. Then it follows by Corollary 3.10 that we have a correlation relation between a linear combination of the keystream bits

$$w_0(z_t \oplus z_{t+p}) \oplus w_1(z_{t-1} \oplus z_{t+p-1}) \oplus \ldots \oplus w_d(z_{t-d} \oplus z_{t+p-d})$$

and a linear combination of the LFSR output bits

$$\begin{gathered} u_t^{k+1}(x_t^{k+1} \oplus x_{t+p}^{k+1}) \oplus \ldots \oplus u_t^n(x_t^n \oplus x_{t+p}^n) \oplus \ldots \\ u_{t-d}^{k+2}(x_{t-d}^{k+2} \oplus x_{t+p-d}^{k+2}) \oplus \ldots \oplus u_{t-d}^n(x_{t-d}^n \oplus x_{t+p-d}^n), \end{gathered}$$

where the output bits from the k first shift register cancel, since they are equal. In order to form these linear combinations at least $p+d$ output bits are required.

By Corollary 3.10, the strength of the correlation over the period p is at least ρ^2. At this point, we can apply the ordinary Divide and Conquer principle. Hence about $(\frac{1}{\rho^2})^2$ bits more are needed. This Ultimate Divide and Conquer attack is stated in the following theorem:

Theorem 4.3 (Ultimate Divide and Conquer). *Assume that in a combination generator we have the following correlation*

$$\begin{aligned} &c(w_0 z_t \oplus w_1 z_{t-1} \oplus \ldots \oplus w_d z_{t-d}, \\ &(u_0^1 x_t^1 \oplus \ldots \oplus u_0^n x_t^n) \oplus \ldots \oplus (u_d^1 x_{t-d}^1 \oplus \ldots \oplus u_d^n x_{t-d}^n)) = \rho \neq 0. \end{aligned}$$

Let the lengths of the registers be $L_1, \ldots, L_n$ and the periods $p_1, \ldots, p_n$. Then given a keystream of length $p_1 p_2 \cdots p_k + \frac{1}{\rho^4} + d$ one can do exhaustive search over the $L_{k+1} + \ldots + L_n$ bits which form the initial contents of $n-k$ registers.

If the registers have primitive feedback polynomials, then $p_i = 2^{L_i} - 1$, as proved in Section 2.3. In most applications n is even and the lengths L_i are about the same. The amount of output bits is determined by the total period of the cancelled registers, namely

$$p = p_1 \ldots p_k = \mathcal{O}(2^{L_1 + \cdots + L_k}),$$

Then the complexity to determine the complete initial state of length $L_{k+1} + \cdots + L_n$ is $\mathcal{O}(2^{L_{k+1} + \cdots + L_n})$. Specifically choosing $k = n/2$ and by generating key stream of length $\mathcal{O}(2^{L/2})$ one can successfully carry out exhaustive search over $L/2$ bits of the initial state, which is of complexity $\mathcal{O}(2^{L/2})$.

Ultimate Divide and Conquer (UDC) attack applies to all types of combination generators, if some correlation between input and output bits has been identified. For example, in Section 4.2.1 we saw that for the summation generator the following correlation holds

$$c(z_t, x_t^1 \oplus x_t^2 \oplus x_{t-1}^1) = 0,5. \tag{4.8}$$

This can be used directly to launch the UDC attack provided that sufficiently many that is, the entire period of one LFSR is available.

Similarly, by Theorem 4.1, correlations of the form (4.8) always exist in a combiner generator with one bit of memory.

4.4 Some criteria for the design of LFSR-combiners

Meier and Staffelbach have studied in [MS90] the effect of the number of LFSRs on the security properties of a combination generator. They note that for an even number of LFSRs the least significant carry bit is balanced, whereas for an odd number it becomes biased. Moreover, the bias goes to zero as the number of LFSRs grows. To make the system secure, one should then have an even number - or very many - LFSRs, since the security of the system depends essentially on the randomness of the carry. The amount of the LFSRs is then limited by the cost and other properties of the system.

We have seen that correlation relations can always be identified between the input and output bits of the combiner functions. We have also seen how such correlations can be exploited in cryptanalysis. In [Gol96] it is claimed that as the size of the memory grows, then grows the security, assuming that the function has been chosen properly. However, the increase in the size of the memory makes the system more expensive.

Finally, J. Massey has proposed in his lecture notes [Mas93] the following criteria for a good nonlinear combiner of LFSRs:

- The LFSRs should be maximal-length LFSR's (or produce PN-sequences) to get good short-term statistics.
- The lengths $L_1, L_2, \ldots, L_n$ of the LFSRs should be relatively prime to get large linear complexity.
- Choose the nonlinear order of the nonlinear combiner function to obtain a good compromise between correlation immunity and linear complexity.
- Choose the nonlinear combiner function to have many terms of each degree up to its nonlinear order being sure to obtain the desired order of correlation immunity.

Linear complexity of a periodic sequence means the length of the shortest LFSR with which the sequence can be produced. The second criterion gives a rule to be followed in the case of a general nonlinear combiner. However, if the combiner is linear, then it is known that to achieve the maximum linear complexity it is sufficient to demand the generating polynomials to be relatively prime, see [vT88] or [LN94].

Chapter 5

Bluetooth

Bluetooth is the code name for a technology specification for a small size, low-cost, short range radio links between mobile PCs, mobile phones and other portable devices. It will enable users to connect a wide range of computing and telecommunications devices, without the need to buy, carry, or connect cables. It enables opportunities for rapid ad hoc connections. Because Bluetooth can be used for a variety of purposes, it will also potentially replace multiple cable connections via a single radio link. Bluetooth's range of operation is approximately 10 meters.

Bluetooth is equipped with point-to-point security mechanisms. Stream cipher encryption is used to protect confidentiality of communication between two Bluetooth components. The stream cipher algorithm, called E_0, is a combination generator with memory. It can be considered as a further development of the summation generator which was analyzed in the preceding chapter. To achieve the speed and adequately small size, the system uses stream ciphering instead of the widely used block ciphers. The key generation is done on E_0, and the rest of the thesis will concentrate on analyzing it.

5.1 Basic definitions

The algorithm E_0 is a special case of a combination generator depicted in Figure 4.1 on page 25 and we will call it the Bluetooth combiner in the sequel. It consists of four LFSRs, the total length of which is 128 bits, and a nonlinear combiner function with memory.

The parameters are as follows: $r = m = l = 2$ and $k = 1$. There are $n = 4$ LFSRs. At some point in the manufacturing process an initial random key is stored into the Bluetooth chip. The actual key management is an open question, but it is beyond the scope of this thesis. We just assume that the needed information is available on the chip. The LFSRs, along with the memory bits are initialized with the key, an address, a random number and clocking bits.

The clocking bits are used to ensure that the system can not be run many times with the same initialization to reveal the bits of the key. After initialization, the system is used to produce the key bit sequence $\{z_t\}_{t\geq 0}$, of 200 bits, and after that it is initialized again, without changing the memory bits. A more detailed description can be found in the specification [SIG99].

The function f_0 is the usual XOR and it outputs the key stream

$$z_t = f_0(x_t, c_t^0) = x_t^1 \oplus x_t^2 \oplus x_t^3 \oplus x_t^4 \oplus c_t^0.$$

The nonlinearity comes from the function f_1, which produces the two-bit sequence s_t :

$$s_{t+1} = (s_{t+1}^1, s_{t+1}^0) = f_1(x_t, c_t) = \left\lfloor \frac{x_t^1 + x_t^2 + x_t^3 + x_t^4 + c_t^1 + c_t^2}{2} \right\rfloor \in \{0,\ 1,\ 2,\ 3\}$$

Here "+" is the ordinary integer-sum. The function is nonlinear, since integer addition is nonlinear in $GF(2)$. The memory update function is a composition of f_1 and T, which is a linear function of the form

$$c_{t+1} = (c_{t+1}^1, c_{t+1}^0) = T(s_{t+1}, c_t, c_{t-1}) = T_0(s_{t+1}) \oplus T_1(c_t) \oplus T_2(c_{t-1})$$

Here T_0, T_1 and T_2 are linear functions. They shall be analyzed later. The function f_1 could be substituted to T, but it is more convenient to handle them separately.

The nonlinearity is the crucial part of the system's security. However, the choice of T does affect the security as well, as we shall later see in Sections 5.3 and 5.4.

5.1.1 The LFSRs of the Bluetooth

Bluetooth has four LFSRs. It was noted in Section 4.4 that this is quite an optimal solution, being an even number, since Bluetooth was meant to be as simple as possible. The LFSRs' feedback polynomials are all primitive and their Hamming weight $(|\{k = 0, 1, \ldots, 2^n - 1 | f(k) = 1\}|)$ is five, which is a compromise between the limitations of hardware implementation and security [SIG99]. The polynomials and some of their properties are presented in Table 5.1 on the following page.

Since the polynomials are primitive, the period of a LFSR of length n is $2^n - 1$, as was proved in Theorem 2.19. The smallest period of all the LFSRs in Bluetooth is then $2^{25} - 1 \approx 3 * 10^7$. This is still quite a long period. The period of the four LFSRs is the product of these four periods, which is $(2^{25} - 1)(2^{31} - 1)(2^{33} - 1)(2^{39} - 1) \approx 10^{38}$. Bluetooth's E_0 algorithm in its intended use generates only strictly limited segments of keystream. After each 2745 bits the generator is reinitialized with new input values, so the entire period is never generated in practise.

The output sequence is taken from the second stage of LFSRs one and three, and from the seventh stage of LFSRs two and four. This does not affect the analysis of the system.

LFSR i	feedback π_i	period length
LFSR 1	$t^{25} \oplus t^{20} \oplus t^{12} \oplus t^8 \oplus 1$	$2^{25} - 1$
LFSR 2	$t^{31} \oplus t^{24} \oplus t^{16} \oplus t^{12} \oplus 1$	$2^{31} - 1$
LFSR 3	$t^{33} \oplus t^{28} \oplus t^{24} \oplus t^4 \oplus 1$	$2^{33} - 1$
LFSR 4	$t^{39} \oplus t^{36} \oplus t^{28} \oplus t^4 \oplus 1$	$2^{39} - 1$

Table 5.1: The feedback polynomials of Bluetooth

5.1.2 The memory in Bluetooth

Bluetooth has two blocks of memory, each consisting of two bits, so that the total number of carry bits is four. This means that two moments, namely t and $t-1$, are kept in the memory.

In Chapter 4 we saw that one bit of memory is not enough at least if only two LFSRs are involved. More should be used to increase the security margin. As noted in [Gol96], only significantly larger memory can offer notable improvement in security. Since the cost of the system is also of importance, the memory must be quite small. The designer's choice of four carry bits can be considered appropriate.

5.1.3 The T-function

The linear T-function used in Bluetooth mixes the carry-bits. It consists of three mappings, T_0, T_1 and T_2 and they operate as follows:

$$\begin{aligned} &c_{t+1} \\ &= T(s_{t+1}, c_t, c_{t-1}) \\ &= T_0(s_{t+1}) \oplus T_1(c_t) \oplus T_2(c_{t-1}). \end{aligned}$$

In matrix form $T_0 = T_1 = I$, where I is the 2×2 identity matrix and

$$T_2 = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}.$$

The functions T_0, T_1 and T_2 can then be written as

$$\begin{aligned} T_0(c^1, c^0) &= T_1(c^1, c^0) = (c^1, c^0) \\ T_2(c^1, c^0) &= (c^1 \oplus c^0, c^1). \end{aligned}$$

This means that the components of $c_t = (c_t^1, c_t^0)$ are

$$c_t^1 = s_t^1 \oplus c_{t-1}^1 \oplus c_{t-2}^0, \tag{5.1}$$
$$c_t^0 = s_t^0 \oplus c_{t-1}^0 \oplus c_{t-2}^1 \oplus c_{t-2}^0. \tag{5.2}$$

With different choices of T the cryptographic properties of the system become different. This shall be analyzed in Section 5.4.

5.2 Analyzing Bluetooth

The Bluetooth combiner is a special case of the combination generator studied in Chapter 2. Since XOR-function is used as the combination function the basic Divide and Conquer attack is not possible. If the memory consists of one bit, by Theorem 4.1 we know that some correlation relations must exist between the inputs and the output of the combination function. Golić has studied the existence of such correlation relations and concludes that they exist also for the case of more than one memory bits [Gol96].

The purpose of this section is to find such correlation relations for the Bluetooth combiner. We will also investigate the role of the mapping T, and show that a small modification in T will slightly improve the correlation properties of the Bluetooth combiner.

5.2.1 Correlations

When determining correlations over the Bluetooth combiner it is practical to start with the nonlinear function f_1. The correlations are calculated between the output of f_1 (i.e., the two-bit vector s_{t+1}) and f_1's inputs that are x_t and c_t.

The Walsh-transforms for the functions s_{t+1}^0, s_{t+1}^1 and $s_{t+1}^0 \oplus s_{t+1}^1$ have been calculated using a computer and the algorithm given in [Mas93] on page 4.63. It was assumed that on time t, the components of the memory are statistically independent of each other. After this the correlations were easy to gain using equation (3.4). In a general form the correlation is written as

$$c(w \cdot s_{t+1}, u \cdot x_t \oplus v \cdot c_t),$$

where $w \in GF(2^2)$ and $w \neq (0,0), u \in GF(2^4)$ and $v = (v^1, v^0) \in GF(2^2)$. With different u and v we get all different linear combinations of $x_t^1, x_t^2, x_t^3, x_t^4, c_t^1$ and c_t^0. In Table 5.2 on the next page the resulting correlations are presented.

We note that since the system is symmetric with respect to each x_t^j, only the Hamming weight of u is of importance. On the other hand for s_{t+1}^0, it should be noted that, if $v^1 = 0$, then the whole correlation is always zero. This has

v^1	v^0	weight of u	s^1_{t+1}	s^0_{t+1}	$s^1_{t+1} \oplus s^0_{t+1}$
0	0	0	0	0	$-\frac{5}{8}$
		1	$\frac{1}{4}$	0	0
		2	0	0	$\frac{1}{8}$
		3	0	0	0
		4	0	0	$-\frac{1}{8}$
0	1	0	$\frac{1}{4}$	0	0
		1	0	0	$\frac{1}{8}$
		2	0	0	0
		3	0	0	$-\frac{1}{8}$
		4	$-\frac{1}{4}$	0	0
1	0	0	$\frac{5}{8}$	$-\frac{1}{4}$	0
		1	0	0	$\frac{1}{4}$
		2	$-\frac{1}{8}$	$\frac{1}{4}$	0
		3	0	0	0
		4	$\frac{1}{8}$	$-\frac{1}{4}$	0
1	1	0	0	0	$\frac{1}{4}$
		1	$-\frac{1}{8}$	$\frac{1}{4}$	0
		2	0	0	0
		3	$\frac{1}{8}$	$-\frac{1}{4}$	0
		4	0	0	$-\frac{1}{4}$

Table 5.2: The correlations for s^0_{t+1}, s^1_{t+1} and $s^0_{t+1} \oplus s^1_{t+1}$.

some security related consequences, which shall be analyzed more carefully in Section 5.4.

Next we shall present the correlation relation we found within the Bluetooth combiner.

We start by rearranging equation (5.2), and add c^0_{t-3} to the both sides of it:

$$c^0_t \oplus c^0_{t-1} \oplus c^0_{t-3} = s^0_t \oplus c^1_{t-2} \oplus c^0_{t-2} \oplus c^0_{t-3}. \tag{5.3}$$

Next we use Theorem 3.7 to get

$$\begin{aligned} & c(c^0_t \oplus c^0_{t-1} \oplus c^0_{t-3}, 0) \\ &= c(s^0_t, c^1_{t-2} \oplus c^0_{t-2} \oplus c^0_{t-3}) \\ &= \sum_{w \in GF(2^2)} c(s^0_t, w \cdot c_{t-1}) c(w \cdot c_{t-1}, c^0_{t-2} \oplus c^1_{t-2} \oplus c^0_{t-3}), \end{aligned}$$

with $u = 0$, $v = (0, 0, 1, 1, 1, 0)$ and $y = (s^0_{t-1}, s^1_{t-1}, c^0_{t-2}, c^1_{t-2}, c^0_{t-3}, c^1_{t-3})$. Now from Table 5.2 we know that the terms of the sum are zero for $w = (0, 1)$, $w = (1, 1)$ and $w = (0, 0)$. Only the term with $w = (1, 0)$ remains. So, the correlation equation is simplified to

$$\begin{aligned}
&c(c_t^0 \oplus c_{t-1}^0 \oplus c_{t-3}^0, 0) \\
&= c(s_t^0, c_{t-1}^1) c(c_{t-1}^1, c_{t-2}^0 \oplus c_{t-2}^1 \oplus c_{t-3}^0) \\
&= c(s_t^0, c_{t-1}^1) c(s_{t-1}^1, c_{t-2}^0).
\end{aligned}$$

Here the last equation is obtained by moving back in time for one step in equation (5.1), so that

$$c_{t-1}^1 = c_{t-1}^1 \oplus s_{t-1}^1 \oplus c_{t-3}^0.$$

Using the values of Table 5.2, we finally get

$$c(c_t^0 \oplus c_{t-1}^0 \oplus c_{t-3}^0, 0) = -\frac{1}{4} \cdot \frac{1}{4} = -\frac{1}{16}. \tag{5.4}$$

Now, (5.4) can be used in the sum

$$z_t \oplus z_{t-1} \oplus z_{t-3} = \bigoplus_1^4 x_t^i \oplus \bigoplus_1^4 x_{t-1}^i \oplus \bigoplus_1^4 x_{t-3}^i \oplus c_t^0 \oplus c_{t-1}^0 \oplus c_{t-3}^0,$$

such that a non-zero correlation between certain output-bits z_t and LFSR bits x_t is achieved. We conclude by equation (5.4) that

$$c(z_t \oplus z_{t-1} \oplus z_{t-3}, \bigoplus_1^4 x_t^i \oplus \bigoplus_1^4 x_{t-1}^i \oplus \bigoplus_1^4 x_{t-3}^i) = -\frac{1}{16}. \tag{5.5}$$

This correlation was found by an exhaustive search. The Bluetooth is still such a small system that this kind of "trial and error"-search is possible. In bigger systems, however, a more advanced techniques for finding these correlation equations should be used. One is presented in [Gol96].

The correlation relation given by (5.4) shall be exploited further in Section 5.3 to apply the UDC-attack on Bluetooth.

5.3 Ultimate Divide and Conquer on Bluetooth

In this section we extend the correlation (5.4) over the period of the shortest LFSR in Bluetooth. One lower bound to the resulting correlation is given by Corollary 3.10, and it is equal to $\rho^2 = (-\frac{1}{16})^2 = \frac{1}{256}$. However, working a little harder we can somewhat increase this lower bound. Applying first Corollary 3.8 we get

$$\begin{aligned}
\rho &= c(c_t^0 + c_{t-1}^0 + c_{t-3}^0 + c_{t+p}^0 + c_{t+p-1}^0 + c_{t+p-3}^0, 0) \\
&= c(s_t^0 + s_{t+p}^0, c_{t-2}^0 + c_{t-2}^1 + c_{t-3}^0 + c_{t+p-2}^0 + c_{t+p-2}^1 + c_{t+p-3}^0) \\
&= \sum_{w,w' \in GF(2^2)} c(s_t^0 + s_{t+p}^0, w \cdot c_{t-1} + w' \cdot c_{t+p-1}) \\
&\quad \cdot c(w \cdot c_{t-1} + w' \cdot c_{t+p-1}, c_{t-2}^0 + c_{t-2}^1 + c_{t-3}^0 + c_{t+p-2}^0 + c_{t+p-2}^1 + c_{t+p-3}^0).
\end{aligned}$$

Now we apply Theorem 3.9 to the first correlation in the product and have

$$\begin{aligned} & c(s_t^0 + s_{t+p}^0, w \cdot c_{t-1} + w' \cdot c_{t+p-1}) \\ & = \sum_{u \in GF(2^k)} c(s_t^0, w \cdot c_{t-1} + u \cdot x) c(s_t^0, w' \cdot c_{t-1} + u \cdot x). \end{aligned} \tag{5.6}$$

Here k is one, two, or three, depending on whether p is the least common period of one, two, or three LFSRs, respectively.

Let us first consider the case of one LFSR. We see from Table 5.2 that only cases $w = w' = (1,1), u = 1$ and $w = w' = (1,0), u = 0$ are non-zero. The correlation then becomes

$$\begin{aligned} \rho &= c(s_t^0, c_{t-1}^1)^2 c(c_{t-1}^1 \oplus c_{t-1+p}^1, c_{t-2}^0 \oplus c_{t-2}^1 \oplus c_{t-3}^0 \oplus c_{t-2+p}^0 \oplus c_{t-2+p}^1 \oplus c_{t-3+p}^0) \\ &+ c(s_t^0, c_{t-1}^1 \oplus c_{t-1}^0 \oplus x_t^1)^2 \\ & c(c_{t-1}^1 \oplus c_{t-1}^0 \oplus c_{t-1+p}^1 \oplus c_{t-1+p}^0, c_{t-2}^0 \oplus c_{t-2}^1 \oplus c_{t-3}^0 \oplus c_{t-2+p}^0 \oplus c_{t-2+p}^1 \oplus c_{t-3+p}^0) \\ &= \sigma_1 + \sigma_2. \end{aligned}$$

We calculate first σ_1. With the aid of Table 5.2 and equations (5.1) and (5.2) σ_1 transforms to

$$\sigma_1 = \frac{1}{16} c(s_{t-1}^1 \oplus s_{t-1+p}^1, c_{t-2}^0 \oplus c_{t-2+p}^0),$$

which again is by Corollary 3.10

$$\sigma_1 = \frac{1}{16} \sum_{w \in GF(2)} c(s_{t-1}^1, c_{t-2}^0 \oplus w x_{t-2}^1)^2 = \frac{1}{256}.$$

Similar calculation are done to σ_2.

$$\begin{aligned} \sigma_2 = \frac{1}{16} & c(s_{t-1}^1 \oplus s_{t-1}^0 \oplus s_{t-1+p}^1 \oplus s_{t-1+p}^0, c_{t-2}^1 \oplus c_{t-2}^0 \oplus c_{t-2+p}^1 \oplus c_{t-2+p}^0) \\ & c(s_{t-2}^1 \oplus s_{t-2}^0 \oplus s_{t-2+p}^1 \oplus s_{t-2+p}^0, c_{t-3}^1 \oplus c_{t-3}^0 \oplus c_{t-3+p}^1 \oplus c_{t-3+p}^0). \end{aligned}$$

We use Theorem 3.7 and Corollary 3.10, so that

$$\begin{aligned} \sigma_2 &= \frac{1}{16} c(s_{t-1}^1 \oplus s_{t-1}^0, c_{t-2}^1 \oplus c_{t-2}^0 \oplus 0 x_{t-2}^1)^2 \\ & \quad c(s_{t-2}^1 \oplus s_{t-2}^0, c_{t-3}^1 \oplus c_{t-3}^0 \oplus 0 x_{t-3}^1)^2 \\ &= \frac{1}{16} \cdot \frac{1}{16} \cdot \frac{1}{16} = \frac{1}{4096}. \end{aligned}$$

Note that the correlations are positive, since we can use Corollary 3.10 instead of just Theorem 3.9 to calculate the correlation. Finally the original correlation is

$$\rho = \sigma_1 + \sigma_2 = \frac{1}{256} + \frac{1}{4096} = \frac{17}{4096} \approx 2^{-8}.$$

We have chosen the one coordinate of x to be x_t^1, as it corresponds to the shortest LFSR in Bluetooth. By Theorem 4.3, the attacker now needs an output sequence of length $p + \rho^{-4} + 3 \approx 2^{25} + 2^{32} \approx 2^{32}$ to do an exhaustive search over the $31 + 33 + 39 = 103$ bits of the remaining three registers.

Next we shall consider the case $k = 2$, where p is the least common period of two LFSRs. From Table 5.2 on page 41 we see that the correlations in equation (5.6) are nonzero if and only if $u = (0,0)$ and $w = w' = (1,0)$, or $u = (1,1)$ and $w = w' = (1,0)$, or $u = (0,1)$ and $w = w' = (1,1)$, or finally, $u = (1,0)$ and $w = w' = (1,1)$.

The value $w = w' = (1,1)$ leads to a longer correlation relation extending over at least two rounds, and hence the resulting amount of correlations is expected to be smaller, however, non-negative, by Corollary 3.10. Therefore, we discard the corresponding terms, and get a lower bound to the correlation from the remaining terms with $w = w' = (1,0)$ as follows

$$
\begin{aligned}
&c(c_t^0 + c_{t-1}^0 + c_{t-3}^0 + c_{t+p}^0 + c_{t+p-1}^0 + c_{t+p-3}^0, 0)\\
&\geq \left(c(s_t^0, c_{t-1}^1)^2 + c(s_t^0, c_{t-1} + x_t^1 + x_t^2)^2\right)\\
&\quad \cdot c(c_{t-1}^1 + c_{t+p-1}^1, c_{t-2}^0 + c_{t-2}^1 + c_{t-3}^0 + c_{t+p-2}^0 + c_{t+p-2}^1 + c_{t+p-3}^0)\\
&= \left(c(s_t^0, c_{t-1}^1)^2 + c(s_t^0, c_{t-1} + x_t^1 + x_t^2)^2\right) \cdot \sum_{u \in V_2} c(s_{t-1}^1, c_{t-2}^0 + u \cdot x)^2\\
&= ((-1/4)^2 + (1/4)^2)(1/4)^2 = 2^{-7},
\end{aligned}
$$

using the correlation values given in Table 5.2. The UDC-attack can be applied. Approximately $2^{28} + 2^{25+31} \approx 2^{56}$ output bits are needed. We can see that the length of the period is the dominating factor in this requirement, while in the case of one LFSR the main requirement was due to the weak correlation.

It should be stressed, however that the presented Ultimate Divide and Conquer attack is of theoretical nature, and practical only if the analyzer is given access to a key stream extending over periods of partial input. Bluetooth's E_0 algorithm in its intended use generates only strictly limited segments of keystream. After each 2745 bits the generator is reinitialized.

5.4 Different choices for mixing the carry bits

When searching for the optimal correlation relation in Section 5.2.1, we made the observation that given the large number of zeros in the correlation table, Table 5.2, the mapping T plays an important role when the correlation relations from subsequent time steps are joined together to a finite chain. In some cases we saw that an approximation may even lead to a continuous chain going back all the way to time step $t = 0$. Motivated by such phenomena we investigate now, what the correlation properties of the Bluetooth would be if T is replaced by some other invertible linear transformation.

The study in this section follows the one presented in [HN00]. Let us first define in general case the functions T_0, T_1 and T_2 to be

$$T_0 = \begin{pmatrix} t_0^1 & t_0^2 \\ t_0^3 & t_0^4 \end{pmatrix}$$

$$T_1 = \begin{pmatrix} t_1^1 & t_1^2 \\ t_1^3 & t_1^4 \end{pmatrix}$$

and

$$T_2 = \begin{pmatrix} t_2^1 & t_2^2 \\ t_2^3 & t_2^4 . \end{pmatrix}$$

Then

$$c_t = T_0 s_t \oplus T_1 c_{t-1} \oplus T_2 c_{t-2}. \tag{5.7}$$

We can write $T_1 = A \oplus B$. Here the analyst - the one who tries to break the system - can choose A and B in which way ever is convenient as long as their sum is T_1.

Equation (5.7) can be written as

$$c_t = T_0 s_t \oplus A c_{t-1} \oplus B c_{t-1} \oplus T_2 c_{t-2}. \tag{5.8}$$

Inserting equation (5.7) with t replaced by $t-1$ to Ac_{t-1} in equation (5.8) yields

$$c_t = T_0 s_t \oplus B c_{t-1} \oplus A T_0 s_{t-1} \oplus A T_1 c_{t-2} \oplus A T_2 c_{t-3} \oplus T_2 c_{t-2}. \tag{5.9}$$

Let D be a matrix

$$D = \begin{pmatrix} d_1 & d_2 \\ d_3 & d_4 \end{pmatrix}.$$

Now, A may be chosen. As the analyst wishes to minimize the amount of approximations done, she wants c_t not to depend on c_{t-3}^1. She chooses $AT_2 = D$, with $d_3 = 0$. Such a choice is always possible if T_2 is invertible. Let us assume this.

Inserting $B = T_1 \oplus A$ into equation (5.9), as well as $A = DT_2^{-1}$, we have

$$\begin{aligned} c_t = {} & T_0 s_t \oplus (T_1 \oplus D T_2^{-1}) c_{t-1} \oplus (D T_2^{-1} T_0) s_{t-1} \\ & \oplus (D T_2^{-1} T_1 \oplus T_2) c_{t-2} \oplus D c_{t-3}. \end{aligned} \tag{5.10}$$

Now, the approximations can be written in matrix form as

$$s_t = E_t c_{t-1},$$

where

$$E_t = \begin{pmatrix} e_t^1 & e_t^2 \\ e_t^3 & e_t^4 \end{pmatrix}.$$

Let us study the correlations of the form

$$c(s_t^0, (e_t^3, e_t^4) \cdot c_{t-1} \oplus u \cdot x_{t-1}),$$

where $u \in GF(2^4)$. From Table 5.2 it can be seen that the correlation is zero if $e_t^3 = 0$ and as this is not an applicable choice, we shall assume $e_t^3 = 1$.

The choice of u does not affect but the signs of the non-zero values of the correlations. This is why $u \cdot x_t$ shall be left out from the calculations, and the main interest lies on the combinations of s_{t+1}^i's and c_t^j's.

We approximate twice by inserting $s_t = E_t c_{t-1}$ into equation (5.10), which yields

$$\begin{aligned} c_t = &(T_0 E_t \oplus T_1 \oplus D T_2^{-1}) c_{t-1} \\ &\oplus (D T_2^{-1} T_0 E_{t-1} \oplus D T_2^{-1} T_1 \oplus T_2) c_{t-2} \oplus D c_{t-3}. \end{aligned} \tag{5.11}$$

We are now ready to state, when two approximations are enough. It is done in the next theorem.

Theorem 5.1. *Let $T_0 = T_1 = I$ and T_2 an invertible 2×2 matrix of the Bluetooth combination generator. If $t_2^3 = 1$ then at most two approximations are needed to establish a non-zero correlation between some combination of input bits and the corresponding output bits.*

Proof. We are only interested in the c_t^0 component of c_t, so we insert $T_0 = T_1 = I$ and $t_2^3 = 1$ to equation (5.11) and have the following equation for c_t^0 :

$$\begin{aligned} c_t^0 = &(1 \oplus d_4, e_t^4 \oplus 1 \oplus d_4 t_2^1) \cdot c_{t-1} \\ &\oplus (d_4 e_{t-1}^1 \oplus d_4 t_2^1 \oplus d_4 \oplus 1, d_4 e_{t-1}^2 \oplus d_4 t_2^1 e_{t-1}^4 \oplus d_4 t_2^1 \oplus t_2^4) \cdot c_{t-2} \\ &\oplus d_4 c_{t-3}^0. \end{aligned}$$

The need of only two iteration approximations means that neither c_{t-1}^1 nor c_{t-2}^1 can be present in the previous equation. This is equivalent to having both of the following equations to be satisfied:

$$\begin{aligned} 1 \oplus d_4 &= 0 \\ d_4 e_{t-1}^1 \oplus d_4 t_2^1 \oplus d_4 \oplus 1 &= 0. \end{aligned}$$

If $d_4 = 0$, the c_t^0-component of equation (5.11) would simplify to equation (5.2), which would mean that the approximation done earlier would cancel out. As this is not what the analyst wants, and as she can choose D in any way convenient, we have $d_4 = 1$. Now for the second equation, we have

$$e_{t-1}^1 \oplus t_2^1 = 0.$$

From Table 5.2 we see that e_{t-1}^1 can be chosen freely, as the correlation $c(s_{t-1}^1, e_{t-1}^1 c_{t-2}^1 \oplus e_{t-1}^2 c_{t-2}^0 \oplus u \cdot x_{t-2}) \neq 0$, for some $u \in GF(2^4)$. So, the second equation can also be satisfied, and only two approximations are needed. ☐

Example 5.2. The correlation matrices for the original T_2 of Bluetooth are

$$E_t = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \text{and} \quad E_{t-1} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}.$$

Even larger correlations can be achieved, if we choose

$$T_2 = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix},$$

since it gives the possibility to use the largest correlation coefficient, $\frac{5}{8}$. ■

Next we investigate whether it would be possible to decrease the correlations by choosing T_2 in a different way.

Theorem 5.3. *Let T_0, T_1 and T_2 be as in Theorem 5.1. Then, if $t_2^3 = 0$, at least three approximation rounds are needed to establish a correlation between some linear combinations of input bits and the corresponding output bits.*

Proof. As $t_2^3 = 0$, and T_2 is invertible, then actually $t_2^1 = t_2^4 = 1$. From equation (5.11) we see that if $d_3 = 1$, then c_{t-3}^1 is within the equation of c_t^0 in which case at least three rounds are needed. This means that two rounds are sufficient only if $d_3 = 0$. But if this is the case, then the coefficient of c_{t-1}^1 is $1 \oplus d_3 = 1$, which means that the two approximations are not sufficient. So, two approximation rounds are never enough. □

Example 5.4. The choice $T_2 = I$ would mean that it takes at least three approximations to get non-zero correlation between the combinations of the output bits z_t and the input bits x_t. Next we shall show that three iterations are actually enough. We will assume that $W_H(u) = 0$, so that no x_t is involved.

Take first equation (5.9) with $T_0 = T_1 = T_2 = I$. Then

$$c_t = s_t \oplus As_{t-1} \oplus (A \oplus I)c_{t-2} \oplus Bc_{t-1} \oplus Ac_{t-3}.$$

In the case of two approximations, $E_t c_{t-1}$ and $E_{t-1} c_{t-2}$ would have been inserted in the place of s_t and s_{t-1}, respectively. This can not be done now, since by the Theorem 5.3, we would end up with c_{t-1}^1 or c_{t-3}^1 in the equation of c_t^0. To be able to drop these c_j^1, $j = t, t-1, t-2, \ldots$ variables from the equation of c_t^0 we use the same trick as earlier, when we wrote $A \oplus B$ instead of T_1. We put $A \oplus I = M \oplus N$, where M and N are arbitrary 2×2 matrices the attacker can choose. Inserting $M \oplus N$ in the place of $A \oplus I$ into the equation above yields

$$c_t = s_t \oplus As_{t-1} \oplus Mc_{t-2} \oplus Nc_{t-2} \oplus Bc_{t-1} \oplus Ac_{t-3},$$

and then, as before, insert equation (5.7) with $t-1$ in the place of t to the equation above to get

$$c_t = s_t \oplus As_{t-1} \oplus Ms_{t-2} \oplus Bc_{t-1} \oplus Nc_{t-2} \oplus (M \oplus A)c_{t-3} \oplus Mc_{t-4}.$$

Now we can approximate three times by inserting $E_j c_{j-1}$, $j = t, t-1, t-2$ in the place of s_j :

$$c_t = (E_t \oplus B)c_{t-1} \oplus (AE_{t-1} \oplus N)c_{t-2} \oplus (ME_{t-2} \oplus M \oplus A)c_{t-3} \oplus Mc_{t-4}.$$

Putting $B = I \oplus A$ and $N = A \oplus I \oplus M$, it yields

$$\begin{aligned} c_t &= (E_t \oplus I \oplus A)c_{t-1} \oplus (AE_{t-1} \oplus A \oplus I \oplus M)c_{t-2} \\ &\quad \oplus (ME_{t-2} \oplus M \oplus A)c_{t-3} \oplus Mc_{t-4}. \end{aligned}$$

Let us denote the entries of A and M as follows:

$$A = \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix} \quad \text{and} \quad M = \begin{pmatrix} m_1 & m_2 \\ m_3 & m_4 \end{pmatrix}.$$

The coefficients of the c_j^1-variables in the expression of the c_t^0-component of the equation above are then

$$\begin{aligned} 1 \oplus a_3 &\text{ for } c_{t-1}^1 \\ e_{t-1}^1 \oplus a_4 \oplus a_3 &\text{ for } c_{t-2}^1 \\ m_3 e_{t-2}^1 \oplus m_4 \oplus m_3 \oplus a_4 &\text{ for } c_{t-3}^1 \\ &\text{and} \\ m_3 &\text{ for } c_{t-4}^1. \end{aligned}$$

These should all be zero for there to be exactly three approximations. We then have $m_3 = 0$, $a_3 = m_4 = 1$, and $e_{t-1} \oplus a_4 = 1$. These can be fulfilled, since M and A were both arbitrary. We now write the equation for c_t^0 :

$$\begin{aligned} c_t^0 &= s_t^0 \oplus c_{t-1}^1 \oplus (a_4 \oplus 1)c_{t-1}^0 \\ &\oplus s_{t-1}^1 \oplus a_4 s_{t-1}^0 \oplus c_{t-2}^1 \oplus a_4 c_{t-2}^0 \\ &\oplus s_{t-2}^0 \oplus c_{t-3}^1 \oplus (1 \oplus a_4)c_{t-3}^0 \oplus c_{t-4}^0. \end{aligned} \tag{5.12}$$

This equation can be used as an aid when calculating the correlation between c_t^0 and c_{t-4}^0 using Theorem 3.7. The summation in the theorem was done over a vector w, which in this case takes values in $GF(2^2)$. In this application, the vector w depends on a_4, such that in the first approximation it is $(1, a_4 \oplus 1)$ and so on. Assuming that the Hamming weight of u is always zero, so that no x_t is involved, we have by Theorem 3.7 and with the aid of Table 5.2 on page 41

$$\begin{aligned} c(c_t^0, c_{t-4}^0) &= c(s_t^0, c_{t-1}^0 \oplus c_{t-2}^0 \oplus c_{t-4}^0) \\ &= \sum_w c(s_t^0, w \cdot c_{t-1})c(w \cdot c_{t-1}, c_{t-1}^0 \oplus c_{t-2}^0 \oplus c_{t-4}^0) \\ &= c(s_t^0, c_{t-1}^1)c(c_{t-1}^1, c_{t-1}^0 \oplus c_{t-2}^0 \oplus c_{t-4}^0). \end{aligned}$$

In the first equality we merely inserted $T_0 = T_1 = T_2 = I$ into equation (5.7), in the second the Theorem 3.7 was used and in the third we noted that the only non-zero correlation $c(s_t^0, w \cdot c_{t-1})$ occurs for $w = (1,0)$. So actually $a_4 = 1$. We continue in the same manner as in the earlier equation:

$$
\begin{aligned}
&c(c_t^0, c_{t-4}^0)\\
&= -\frac{1}{4} c(s_{t-1}^1 \oplus s_{t-1}^0, c_{t-2}^1 \oplus c_{t-3}^1 \oplus c_{t-3}^0 \oplus c_{t-4}^0)\\
&= -\frac{1}{4} \sum_w c(s_{t-1}^1 \oplus s_{t-1}^0, w \cdot c_{t-2}) c(w \cdot c_{t-2}), c_{t-2}^1 \oplus c_{t-3}^1 \oplus c_{t-3}^0 \oplus c_{t-4}^0)\\
&= -\frac{1}{4} \left(c(s_{t-1}^1 \oplus s_{t-1}^0, 0) c(c_{t-2}^1 \oplus c_{t-3}^1 \oplus c_{t-3}^0 \oplus c_{t-4}^0, 0) \right)\\
&+ c(s_{t-1}^1 \oplus s_{t-1}^0, c_{t-2}^1 \oplus c_{t-2}^0) c(c_{t-2}^0, c_{t-3}^1 \oplus c_{t-3}^0 \oplus c_{t-4}^0).
\end{aligned} \tag{5.13}
$$

By equation (5.12) we see that from these two sums inside the brackets only the last one is non-zero, since with $a_4 = 1$ only the case with $w = (1,1)$ can occur. In an other way we see this by noting that

$$
\begin{aligned}
&c(c_{t-2}^1 \oplus c_{t-3}^1 \oplus c_{t-3}^0 \oplus c_{t-4}^0, 0)\\
&= c(s_{t-2}^1, c_{t-3}^0 \oplus c_{t-4}^1 \oplus c_{t-4}^0)\\
&= \sum_w c(s_{t-2}^1, w \cdot c_{t-3}) c(w \cdot c_{t-3}, c_{t-3}^0 \oplus c_{t-4}^1 \oplus c_{t-4}^0)\\
&= c(s_{t-2}^1, c_{t-3}^0) c(c_{t-4}^1 \oplus c_{t-4}^0, 0) + c(s_{t-2}^1, c_{t-3}^1) c(c_{t-3}^1, c_{t-3}^0 \oplus c_{t-4}^1 \oplus c_{t-4}^0).
\end{aligned}
$$

In the last equality the first term in the sum is zero, since the memory bits on the same moment are statistically independent. We continue with the second part of the sum:

$$
\begin{aligned}
&c(c_{t-3}^0 \oplus c_{t-3}^1, c_{t-4}^1 \oplus c_{t-4}^0)\\
&= c(s_{t-3}^1 \oplus s_{t-3}^0, c_{t-5}^1 \oplus c_{t-5}^0)\\
&= \sum_w c(s_{t-3}^1 \oplus s_{t-3}^0, w \cdot c_{t-4}) c(w \cdot c_{t-4}, c_{t-5}^1 \oplus c_{t-5}^0)\\
&= -\frac{5}{8} (c(s_{t-3}^1 \oplus s_{t-3}^0, 0) c(c_{t-5}^1 \oplus c_{t-5}^0, 0)\\
&+ c(s_{t-3}^1 \oplus s_{t-3}^0, c_{t-4}^1 \oplus c_{t-4}^0) c(c_{t-4}^1 \oplus c_{t-4}^0, c_{t-5}^1 \oplus c_{t-5}^0))\\
&= -\frac{5}{8} \cdot \frac{1}{4} c(c_{t-4}^1 \oplus c_{t-4}^0, c_{t-5}^1 \oplus c_{t-5}^0)).
\end{aligned}
$$

These calculations are easy to generalize to any moment $t-k$, $k = 0, 1, 2, \ldots,$ so actually we have

$$
\begin{aligned}
&c(c_{t-3}^0 \oplus c_{t-3}^1, c_{t-4}^1 \oplus c_{t-4}^0)\\
&= \left(-\frac{5}{32} \right)^k c(c_{t-3-k}^1 \oplus c_{t-3-k}^0, c_{t-4-k}^1 \oplus c_{t-4-k}^0).
\end{aligned}
$$

Hence, infinitely many approximations should be done, and so the correlation can be regarded as zero:

$$
c(c_{t-3}^0 \oplus c_{t-3}^1, c_{t-4}^1 \oplus c_{t-4}^0) = \lim_{k \to \infty} \left(-\frac{5}{32} \right)^k = 0.
$$

Let us now return to equation (5.13), where we get

$$\begin{aligned}
&c(c_t^0, c_{t-4}^0) \\
&= -\frac{1}{4} \cdot \frac{1}{4} c(c_{t-2}^0, c_{t-3}^1 \oplus c_{t-3}^0 \oplus c_{t-4}^0)) \\
&= -\frac{1}{16} c(s_{t-2}^0, c_{t-3}^1) = -\frac{1}{64},
\end{aligned}$$

which is notably smaller than $\frac{1}{16}$ in equation (5.4). The correlation can be calculated for other weights of u, too. We see, however that as a_4 can have only one value, we always get one product, as above, so that the product of the correlations is always smaller that $\frac{1}{16}$. ■

Combining Theorems 5.1 and 5.3 gives a necessary and sufficient condition under which three approximations are needed. We conclude that making a minor change in the Bluetooth combiner would significantly improve its correlation properties.

5.5 Conclusions about Bluetooth

We shall end this Chapter with some basic conclusions about Bluetooth. Since there were opposite issues, like security and cost that had to be taken into account when the specification was written the analysis is not quite easy.

The obvious security criteria were however fulfilled. There were four LFSRs in use instead of two which would have been a very weak choice. The registers' generating polynomials were primitive, which is the minimum criterion for them to be acceptable. The polynomials are somewhat sparse, but as noted in the specification [SIG99], this is a compromise between higher cost and security.

The size of the memory could not be any smaller. The choice four for the memory's size is quite good. Two carry bits from two consequent time steps that is, four bits in total, is kept in the memory.

The function f_1 is not balanced so we have correlations such as

$$c(s_t^0, u \cdot x_{t-1} \oplus c_{t-1}^1)$$

that help the correlation attack on the system. However, we can see in Table 5.2 that f_1 has correlation to several linear combinations of its inputs and the correlations are approximately of the same magnitude.

It was concluded that the function T could have been chosen better. The choice $T_2 = I$ does not affect the implementation or performance of the system but it gives the system better correlation properties than the original choice for T_2.

Chapter 6

Conclusions

The main goal of this thesis was to study and analyze the algorithm E_0 defined in the Bluetooth specification. The study concentrated on the internal properties of this stream cipher algorithm, while key management, initialization, and other practical security issues were left outside the scope of this work.

Our starting point was to realize the fact that the Bluetooth algorithm has its origins in a previously existing and well-studied principle of summation combiner, or more generally, combination generator with memory. Reviewing existing literature was essential for understanding the design principles of the Bluetooth algorithm. We further investigated the actual design choices of Bluetooth, and observed that they are not quite optimal since better and stronger alternatives would have been available.

The study started with the definition of pseudo-random generators and continued further to finite fields and linear feedback shift registers, which form the driving component of E_0. As all the generation polynomials are primitive, each LFSR produces a PN-sequence, i.e., the longest possible sequence with respect to the length of the LFSR.

The main analytical tool was correlations. The concept of correlation was defined for Boolean functions. A few new correlation theorems were proved. They can be applied to any combiner generators or similar situations were fixed functions are iterated [Nyb99].

The original summation combiner presented by Rueppel [Rue86] and studied by Meier and Staffelbach [MS92] is cryptographically weak. We present the analysis of [MS92]. It was concluded that there should be at least four LFSRs and preferably, more memory bits for the combination generator to be cryptographically secure. Bluetooth has four LFSRs and four bits of memory.

We computed the correlations over the nonlinear part of the combination function. After that we determined the shortest (and strongest) correlation between the input bits and the output bits of the combination function. This correlation involves bits from four subsequent time steps. A more thorough analysis enables us to prove a necessary and sufficient condition under which at least five time steps are involved. This shows, in particular that the nonlinear and linear parts of the combiner function must be handled together in order to determine the correlation behavior of the entire system.

The Ultimate Divide and Conquer attack was introduced. The attack is based on the periodicity of the LFSRs. In a system with four LFSRs such as Bluetooth, after the period of one, two or three of the LFSRs we can leave, respectively, one, two or three of the variables out. As a result we have a non-zero correlation between some linear combinations of the key stream bits and some linear combinations the output bits of three, two or one LFSRs, respectively.

It was noted that when the registers have somewhat equal lengths, like in Bluetooth, and their generating polynomials are primitive, then having half of the initial L bits solved means that $\mathcal{O}(2^{L/2})$ output bits must be studied. In Bluetooth this is $\mathcal{O}(2^{64})$. But in Bluetooth the number of bits produced in one time is bounded by 2745. This means that the attacker does not have access to a sufficient amount of keystream if the Bluetooth component is used according to the specification.

Based on this study, we conclude that the algorithm E_0 must not be applied to some other system without modification. If the new system produces longer key sequences, the Ultimate Divide and Conquer attack would soon turn into a practical attack. This is of course true with most of the encryption algorithms, they might provide sufficient security in one application, but be weak in an other.

Some topics remain unexplored and can be suggested for further research. It would be useful to develop an algorithm for searching strong correlations over the combiner with memory. Then one might be able to tackle the problem of designing combiners with optimal correlation properties. In the case of Bluetooth, this would make it possible to study all the parts of the T-function, i.e., T_0, T_1 and T_2.

More theoretical work could be done to investigate generalizations of Theorem 4.1 to cases with more than one bit memory. In particular, the optimal memory structure could be investigated. It is an open question, whether a memory of n bits should be arranged to contain one bit for each n consequent time steps, or should all n bits be kept for one time step only. There are only heuristic argument to support the choice made in Bluetooth to keep $\sqrt{n}$ bits in memory for $\sqrt{n}$ consequent time steps.

Also many other correlation based attacks presented for example in [Rue92] could be compared with the UDC in different summation combiners. Also UDC could be generalized for any combination generators, with general pseudorandom generators and nonlinear combiners.

Appendix A

Table of notations and abbreviations

Notation	Meaning	See page
PRG	pseudo-random generator	5
$\{x_i\}_{i\geq 0}$	sequence $x_0, x_1, x_2, \ldots$	5
$\mathrm{AC}(k)$	autocorrelation of a periodic sequence $\{x_i\}$	6
$\mathbb{Z}_2$	group of integers modulo 2, the same as GF(2)	8
$\deg(\pi(t))$	the degree of polynomial $\pi(t)$	8
LFSR	linear feedback shiftregister	9
α_i	the ith feedback coeffiecient of an LFSR	9
$\pi(t)$	characteristic polynomial of an LFSR	10
$S(t)$	generating polynomial of an LFSR	10
PN	a sequence of maximal length	10
$\Omega(\pi)$	set of output sequences generated by $\pi(t)$	10
$\dim(V)$	the dimension of a vector space V	11
$\tau^*(t)$	the reciprocal of polynomial $\tau(t)$	11
$\lvert\mathcal{A}\rvert$	number of elements in $\mathcal{A}$	17
$a \cdot b$	innerproduct between vectors a and b	17
L_w	a linear Boolean function	17
$GF(2^n)$	the Galois Field of 2^n-elements	17
XOR	modulo 2 summation	17
$\sum_{i=b}^{n} x_i$	the integer sum of x_i's	17
$\bigoplus_{i=b}^{n} x_i$	modulo 2 sum of the variables x_i	17
$\mathcal{W}_f$	Walsh-transform of function f	18
$c_x(f(x), w \cdot x)$	correlation between $f(x)$ and $w \cdot x$ with respect to x	19
x_t	output-vector of an LFSR-system	27
X_i	$X_i = (x_t, \ldots, x_{t-i})$	28

Bibliography

[Gol96] J. Golić. Correlation properties of a general binary combiner with memory. *Journal of Cryptology*, 9(2):111–126, 1996.

[HN00] M. Hermelin and K. Nyberg. Correlation Properties of the Bluetooth Combiner Generator. In J.S. Song, editor, *The 2nd International Conference on Information Security and Cryptology of ICISC'99*, volume 1787 of *Lecture Notes in Computer Science*. Springer-Verlag, 2000.

[LN94] R. Lidl and H. Niederreiter. *Introduction to finite fields and their applications*. Press Syndicate of the University of Cambridge, second edition, 1994.

[Mas93] J. Massey. *Cryptography: Fundamentals and applications*. Advanced technology seminars, first edition, 1993.

[MS90] W. Meier and O. Staffelbach. Cryptographic Significance of the Carry for Ciphers Based on Integer Addition. In A. J. Menezes and S. A. Vanstone, editors, *Advances in Cryptology—CRYPTO '90*, volume 537 of *Lecture Notes in Computer Science*, pages 601–614. Springer-Verlag, 1990.

[MS92] W. Meier and O. Staffelbach. Correlation Properties of Combiners with Memory in Stream Ciphers. *Journal of Cryptology*, 5(1):67–86, 1992.

[Nyb99] K. Nyberg. Correlation Theorems in Cryptanalysis. Unpublished manuscript, 1999.

[Rue86] R.A. Rueppel. Correlation immunity and the summation generator. In H. Williams, editor, *Advances in Cryptology - CRYPTO '85*, volume 218 of *Lecture Notes in Computer Science*, pages 260–272. Springer-Verlag, 1986.

[Rue92] R.A. Rueppel. Stream Ciphers. In G.J. Simmons, editor, *Contemporary Cryptology, The Science of Information Integrity*. IEEE Press, 1992.

[SIG99] Bluetooth™ SIG. *The Bluetooth specification version 1.0*, July 1999.

[vT88] Henk van Tilborg. *An Introduction to Cryptology*. Kluwer Academic Publishers, first edition, 1988.

[XM88] G.Z. Xiao and J.L. Massey. A spectral characterization of correlation immune functions. *IEEE Transactions on Information Theory*, 34(3): 569–571, May 1988.