DISCRETE FOURIER TRANSFORM ATTACK

by

Major Rashid Zulfiqar

A thesis submitted to the faculty of Information Security Department Military College of Signals, National University of Sciences and Technology, Rawalpindi in partial fulfillment of the requirements for the degree of MS in Information Security

MAY 2015

SUPERVISOR CERTIFICATE

It is certified that the final copy of thesis has been evaluated by me, found as per specified format and error free.

____

Dr. Mehreen Afzal

ABSTRACT

Discrete Fourier Transform (DFT) attack is a latest cryptanalytic technique to recover the initial state of a keystream generator by reducing the number of unknowns in the system of linear equations to the minimum, by a corresponding increase in complexity of pre-computation and substitution. The complexity increase is a trade off with reduced unknowns and numbers of the required consecutive bits of the keystream sequence equal to the linear span of the sequence. This attack has been categorized in two versions. One version requires the captured keystream sequence equal to linear span of the cipher for recovering the initial state/key. The second version of this attack reduces the requirement of keystream by lowering the linear span by multiplying the sequence by another sequence having the linear span less than the linear span of the original sequence to recover the key. The DFT attack results in an efficient attack than Fast Algebraic Attack (FAA).

This research explores the possibility of the application of the DFT attack on practical symmetric cipher structure. It includes all versions of DFT attack (Ronjom et al. New Attack, Selective and Fast Selective) and has been tried against different structures including block ciphers, stream ciphers with filtering sequence generators and clock controlled ciphers. The attack has been applied on practical stream cipher Welch Gong (WG)-7. WG-7 is a lightweight, hardware oriented stream cipher that uses a word oriented linear feedback shift register (LFSR) and a nonlinear WG transformation that acts on the LFSR output word. The research aims at faster recovery of keystream than predicted complexity of the DFT attack by the designers.

DEDICATION

All praise and thanks to almighty Allah, the most Gracious and the most Compassionate, Master of the Day of Judgment.

I dedicate my work to my mother, my wife and my teachers who have been a constant source of support and direction for me throughout my work.

ACKNOWLEDGEMENT

I would like to thank Allah who has increased my awareness and understanding of the research topic and blessed me to successfully complete my work in time.

I would like to acknowledge my supervisor Dr. Mehreen Afzal. I am extremely thankful and indebted to her for sharing expertise and sincere and valuable guidance and encouragement extended to me. She continuously kept me in the right course by priceless suggestions. Timely completion of my research work would not be possible without her constant support. I take this opportunity to express gratitude to all of the Department faculty members including my guidance committee members Col Dr. Hassan Islam, Lt Col Dr. Babar Aslam and Lecturer Muhammad Waseem Iqbal for their unrelenting support and help in this venture.

I am also thankful to Jing Jing Wang, who helped me in the understanding of her work on DFT attack against Bluetooth Cipher through her emails.

I am thankful to my Head of Department and his team for sharing expertise, and sincere and valuable guidance and encouragement extended to me.

I also thank my family members for the unceasing encouragement, support and attention throughout this venture of MS studies.

TABLE OF CONTENTS

LIST OF FIGURES

Figure Number *Page*

LIST OF TABLES

KEY TO ABBREVIATIONS

RFID	Radio Frequency Identification
DFT	Discrete Fourier Transform
PC	Personal Computer
MAC	Message Authentication Code
IV	Initialization Vector
XOR	Exclusive OR Operation
TC	Toy Cipher
GF	Galois Field
ANF	Algebraic Normal Form
MSB	Most Significant Bit
LSB	Least Significant Bit
M	Plaintext
C	Ciphertext
K	Master Key
K_i	Round Subkey
$\oplus$	Bitwise Exclusive-OR Operation
$<<<$	Left Cyclic Shift Operation
$>>>$	Right Cyclic Shift Operation
$\|\|$	Concatenation
GUI	Graphical User Interface
NLFSR	Non-Linear Feedback Shift Register
LFSR	Linear Feedback Shift Register
CPU	Central Processing Unit

Chapter 1

Introduction

1.1 Overview

Cryptanalysis is the field which has a long history similar to cryptography. It may be regarded as an essential part for the design of cryptographic algorithm. The techniques range from simple frequency analysis to generic methods like correlation, differential, linear, algebraic and many other attacks. The diverse nature of these methods is due to growing complexity and sophistication of ciphers. New cryptanalytic techniques are in a continuous evolving phase to counter this growing complexity and to ensure trust worthiness of these ciphers.

Algebraic and Fast Algebraic attacks have received a great attention in the cryptology community in the last decade. Algebraic cryptanalysis converts the problem of breaking the cipher into the system of equation and their subsequent solution to retrieve initial state. One can be sure of the security of the cipher if one is unable to solve the system of equation in the polynomial time. To counter the threat of Algebraic/Fast algebraic attacks, cryptographers are designing ciphers with better algebraic immunity (AI).

In order to improve upon the existing method of algebraic/fast algebraic attacks, researchers have found the Discrete Fourier Transform (DFT) attack. It is a latest cryptanalytic technique to recover the initial state of a keystream generator by reducing the number of unknowns in the system of linear equations to the minimum, by a corresponding increase in complexity of pre-computation and substitution. The complexity increase is by forming a system of equations over Galois field (GF) with less unknowns and reduction in the number of the required consecutive bits of the keystream sequence equal to the linear span of the sequence. This attack has been categorized in two versions. One version requires the captured keystream sequence equal to linear span of the cipher for recovering the initial state/key. Then the shift equivalent version of the sequence is developed and related with the captured keystream using DFT to recover the shift for finding the key. This version works well for the case where linear complexity is small enough to carry out computations in a polynomial time. The second version of this attack reduces the requirement of keystream by lowering the linear span by multiplying the sequence by another sequence having the linear span less than the linear span of the original sequence to recover the key. The attack version is known as the fast selective DFT attack. The fast selective DFT attack results in an efficient attack than Fast Algebraic Attack (FAA). It can also work where the number of the known consecutive bits of the keystream is too small to apply FAA. Moreover, the fast selective DFT attack will work where the employed Boolean functions have high algebraic immunity.

The DFT attack has been mainly tested against filter generators and LFSR combiner generator ciphers due their structure. Among them, WG [16] and E_0 [17] Cipher has been assessed for DFT attack by Helleseth et al. [2] and Wang et al. [8] respectively. Moreover, WG variants like WG-7[9], WG-8[27] and WG-16[28] have been assessed during their design phase against DFT attack. Beside this, no further attempts have been made to launch this attack for cryptanalysis of practical ciphers. Moreover, limited research is available on application of this attack on block ciphers and none available against clock controlled ciphers.

This research aims at applying the DFT attack on symmetric cipher structure to evaluate the dimensions of the attack including block ciphers, stream ciphers including filtering sequence generators, combination sequence generators and irregular clocked ciphers. All the previous work in literature has been carried out on filter generators and combiner generators with encouraging results. WG and its variants are word oriented stream cipher based on filter generator design. WG-7 has been selected for further investigation by fast selective DFT Attack due to discovery of annihilator against the cipher [10].

WG-7[9] is a lightweight, hardware oriented stream cipher that uses a word oriented LFSR and a nonlinear WG transformation that acts on the LFSR output word. The cipher is designed for RFID tags, mobile and other resource constrained applications. The research aims at faster recovery of keystream than predicted complexity of the DFT attack by the designers [9] and algebraic attack [10]. Trace polynomials which can relate the internal states of the cipher and the output bits as a result of Boolean function will be developed in this work. These polynomials will be analyzed in terms of their characteristics field and using Annihilators to recover the keybits. Annihilators tend to reduce the requirement of keystream bits and lower linear complexity which helps to efficiently solve the system of equations to recover the internal state/key.

The goal of this work is to analyze the application of Discrete Fourier Transform attack against symmetric ciphers with a view to improve the efficiency by reducing complexity in terms of reduced computations/keystream bits.

1.2 Problem Statement

There is a need to test the symmetric ciphers against the DFT attack as it is an emerging threat. Most of the ciphers can be vulnerable to the DFT attack, which include block cipher and clock controlled ciphers. Moreover, practical cipher structures like Welch Gong (WG-7) also need to be tested against DFT attack with a view to reduce its complexity.

1.3 Objectives

The objectives of this thesis are to develop and implement DFT attack on vulnerable symmetric ciphers. To improve the efficiency of already applied DFT attack against WG family of ciphers. Represent a cipher output by its equivalent polynomial representation for evaluation of its algebraic structure/spectrum. Finding low degree annihilators for applying fast selective DFT attack and represent them in an equivalent polynomial form. To use annihilators in Fast Selective DFT attack to recover the initial state/key.

1.4 Research Methodology and Achieved Goals

The research work has been divided into three main phases. In the first phase, detailed study and literature review has been carried out related to the DFT attack. A strong theoretical concept has been built regarding the working of the DFT attack. In the second phase, the implementation of the DFT attack has been carried out and WG-7 has been thoroughly investigated with new modified methods proposed as well. MAPLE has been used for the testing because it was found suitable for the development of the thesis. In the last phase, DFT attack has been tested against Toy Block Cipher, Alternating Step Generator and Shrinking Generator.

WG-7 Cipher has been found vulnerable against DFT attack after application of an annihilator. WG-8 and WG-16 Cipher have been found secure due to non availability of annihilator. However, the Toy Block Cipher, Alternating Step Generator and Shrinking Generator have been found secure against DFT attack.

1.5 Thesis Organization

The thesis is organized as follows. Chapter 1 introduces the topic with the problem statement and objectives of thesis. Chapter 2 reviews the literature on DFT Attack. Chapter 3 describes the DFT attack versions with examples. Chapter 4 give details regarding the application of the attack on non-linear filtered sequence generator ciphers. Chapter 5 give details regarding the application of the attack on clock controlled ciphers and block ciphers. Chapter 6 concludes the thesis with objectives achieved and proposed future work.

Chapter 2

Literature Review

2.1 Introduction

In this chapter evolution of DFT attack and its variations/improvement has been discussed in detail. Variations of DFT attack developed so far have also been compared. Moreover, DFT attack complexity on stream ciphers such as variants of Welch Gong (WG) cipher and Bluetooth cipher has been reviewed as well.

The chapter has been divided into five sections. Section 2.2 gives an account of stream cipher cryptanalysis. Section 2.3 briefly illustrates algebraic attack. Section 2.4 describes evolution of the DFT attack. Section 2.5 discusses spectral Immunity and relation with algebraic attack.

2.2 Stream Cipher Cryptanalysis

Linear feedback shift register (LFSR) sequences are widely used as basic functional blocks in key stream generators in stream cipher models due to their fast implementation in hardware as well as in software in some cases. Examples include filtering sequence generators, combinatorial sequence generators, clock controlled sequence generators, and shrinking generators. For an LFSR based stream cipher, the initial state of the LFSR serve as a cryptographic key in each communication session. The goal of an attack is to recover the key from some known bits of the keystream. Consequently, the remaining bits of the keystream used in that session can be recovered and can be used in subsequent communication by only changing the known IV each time. There are many proposed attacks on LFSR based stream ciphers in the literature. The assumption taken for DFT attack is always known plain-text attack. The attacker has always access to keystreams generated by the cipher for analyzing the output. There are broadly two types of attack which can be performed by the attacker. These are key recovery attacks and distinguishing attack. The key recovery attacks tries to recover the key or initial state whereas the distinguishing attack tries to distinguish the keystream from the random one. This attack is applicable where the Boolean function used in the cipher is imbalanced.

2.3 Algebraic Attacks

Algebraic attacks have received a great attention in the cryptology community in the last decade. Algebraic cryptanalysis converts the problem of breaking the cipher into the system of equation and their subsequent solution to retrieve initial state. One can be sure of the security of the cipher if one is unable to solve the system of equation in the polynomial time. The algebraic attack relates the secret key and the output of the cipher. The system of equation carries all the information regarding the cipher and analyzing the system of these equations gives strong way for evaluating the cipher. There is huge potential for analysis just by seeing the published results by various authors [19][20][21][22][23][24]. These attacks consist of three steps: precomputation step to generate variable degree equations, substitution for establishing a system of low-degree equations from captured keystream bits and solving the system of equation to recover initial state.

2.4 Evolution of DFT Attack

The DFT attack evolved as an improvement of fast algebraic attack and algebraic attack. In 2003, Courtois [21] proposed the fast algebraic attack (FAA) on stream ciphers to step up the algebraic attack by determining the linear relations among the key stream bits. As compared to Algebraic attack that solves a system of equations by linearization and Gaussian elimination, the fast algebraic attack lessens the solving complexity by decreasing the total degree of the equation system. As a result, it thus reduces the number of monomials in the system and the essential number of keystream bits. The effectiveness of the pre-computation and replacement in the fast algebraic attack is further improved upon by Hawkes and Rose [23] for filtering sequence generators and Armknecht [22] for combination sequence generators with or without memory, respectively. Armknecht and Ars [24] introduced a variant of the FAA which minimized the number of required consecutive bits of the key stream, but the number of unknowns remains unchanged.

2.4.1 Rønjom-Helleseth New Attack

Rønjom and Helleseth [1] introduced the new attack, to recover the initial state of a filtering sequence generator. The attack is more efficient than the original algebraic attack and fast algebraic attack, but needs more keystream when evaluated against fast algebraic attack. They have proven the assumption that filter generator associated equation system behaviour is completely deterministic. An equation system is related to certain coefficient sequences and their minimal polynomial completely determines the linear structure of equation generated. Also it is proven that roots of the minimal polynomials have hamming weight found from the monomials degree in the filter function. The same degree monomials in consecutive equations generate the same polynomial. Then another polynomial can be computed that annihilates the coefficient sequence of nonlinear monomials. As a result, linear system of equation in n variables for initial states is generated which can be solved inconsequentially. The attack utilizes the properties of the coefficient sequences associated with the multivariate polynomial representation. The complexity of the pre-computation phase is $O(E(\log_2(E))^3)$, while the online complexity of the attack is $O(E)$ with data complexity of E keystream bits. This attack is most valuable in terms of the number of monomials in the equation system and normally linear in the linear complexity of the keystream.

Given $\{D_k\}$, the DFT of LFSR sequence {d} and assuming that l (consecutive) bits of $b(b_0, b_1 \ldots b_{l-1})$ are known. Since $b = d_{t+\tau}$.Then DFT of b can be found, since $B_k = \beta^k D_k$, $\beta \in F_2^n$, it is enough to find β. Recovering a key in the filtering sequence is to recover an initial state in the LFSR, which is equal to recover β. For the selective case, by applying h(x) to d, the nonzero DFT spectra of the resulting sequence is equal to a subset of the nonzero DFT spectra of d. Thus the linear complexity of $y_t = h(\alpha^k)D_k$ is less than or equal to the linear complexity of d. For required number of consecutive bits = LS(b). h(x) is the quotient of the minimal polynomial of b and the minimal polynomial of α. So, h(L) removes all DFT spectra except for D_1. It works if $D_1 \neq 0$.

Rønjom and Helleseth [2] have extended the same attack from [1] to the case of filter generators over finite fields F_m where $m = 2^n$, consisting of a primitive feedback polynomial over F_m generating a sequence that is filtered through a nonlinear Boolean function. The coefficient sequences are generated similar to the binary case; however, there are some basic distinctions. It is shown that filter generators over F_m for which only one word from the LFSR is chosen as input to a Boolean function, produce heavily disintegrated sequences and only a fraction of linear complexity is achieved from generation of these sequences. The first application on practical cipher structure has been against WG cipher. WG cipher has been introduced by Yasir et al.[16] as candidate of eSTREAM Project. The Cipher keystream generator consists of 11 stages LFSR over F_2^{29}. The feedback polynomial of the LFSR is primitive over F_2^{29} and produces a maximal length sequence over F_2^{29}. This m-sequence is filtered by a nonlinear WG transformation $F_{2^{29}} \rightarrow F_2$. With a pre-computation of 2^{82}, the fast algebraic attack on WG has complexity 2^{86} with access to approximately 2^{70} bits of keystream. The linear complexity of WG Cipher has been found by the designer as $2^{45.0415}$. The initial state in the WG Cipher can be found with a keybits requirement equivalent to $2^{45.0415}$. The pre-computation complexity is $O(2^{62})$ for generation of polynomial corresponding to the coefficient sequence of degree 2 and above. One interesting observation is made regarding flaw in design of WG Cipher for smaller linear complexity as Boolean function acts on bits in a single word. If all bits are utilized in the Boolean function the linear complexity is much higher to avoid any DFT attack. However, the designer [16] has restricted the keybits on a particular key to 2^{45} to avoid the attack and introduce a guess of keybits to launch DFT attack. In this case, the results presented prove that DFT attack is exceptionally fast in comparison to fast algebraic attacks.

Rønjom et al. [3] generalized the new attack by forming a system of linear equations over F_2^n instead of F_2. Instead of using coefficient sequences, filtered sequences with their respective trace representation have been utilized. The attacks widen the degree of choice when attacking combiner and filter generators and covers special cases in which the original attack might fail. It works better in the unlikely cases when the original attack needed some modifications. The complexity of the attack is fundamentally unchanged. Now h(x) is the quotient of the minimal polynomial of b and the minimal polynomial of α^k. So, h(L) removes all DFT spectra except for D_k for some k with $D_k \neq 0$ and gcd(k,N) = 1.

Rønjom et al. [5] described the attack exclusively in terms of matrices and linear algebra. Also is shown that any cipher based on a linear state machine can be attacked using this attack technique. Further, it has been proven that the binary filter generator can be represented in terms of recurring vector spaces with respect to an invariant matrix W. The matrix W is derived by enlarging the companion matrix of the LFSR and it is revealed that it represents the coefficient sequences. It is proved that the attack can be described entirely in terms of linear algebra and thus harmonize analysis in terms of the trace-representation over F_2^m.

Rønjom et al. [18] mentioned the same attack on the combiner generator. The subspace annihilator’s polynomials are not restricted to ground field and polynomials over the extension field are utilized for efficiency. This attack is applied to combiner generators consisting of LFSRs with relative prime periods. The attack always remains linear in the linear complexity of keystream sequence.

Yiyuan Luo et al. have applied the DFT attack while designing stream cipher WG-7[9]. The attack version applied for the attack is Ronjom and Helleseth new attack on binary filtering generators over F_2^m. The complexity of the attack has been found out to be $2^{29.5}$ keystream bits after a pre-computation with a complexity of $O(2^{39.5})$. If the attacker obtains the keybits less than 2^{25}, the attacker has to guess $2^{23.5}$ unknown bits to launch the DFT attack. The authors have concluded that best attack against the WG-7 is the exhaustive search as attacker cannot obtain 2^{24} consecutive keystream bits.

WG-8 is a lightweight variant of the well-known Welch-Gong (WG) stream cipher family with 80-bit secret key and 80-bit initial vector (IV), which can be regarded as a nonlinear filter generator over finite field F_2^8. The stream cipher WG-8 consists of a 20-stage LFSR with the feedback polynomial l(x) followed by a WG-8 transformation module with decimation d = 19, and operates in two phases, namely an initialization phase and a running phase. Xinxin Fan et al. have applied the DFT attack while designing stream cipher WG-8[27]. The attack version applied for the attack is Ronjom and Helleseth new algebraic attack on binary filtering generators over F_2^m. The complexity of the attack has been found out to be $2^{33.32}$ keystream bits after a pre-computation with a complexity of O($2^{48.49}$). They concluded that best against the WG-7 is the exhaustive search as attacker cannot obtain $2^{33.32}$ consecutive keystream bits due to exchange 32-bits random information.

WG-16 is an efficient variant of the well-known Welch-Gong (WG) stream cipher family with 128-bit secret key and 128-bit initial vector (IV). The stream cipher WG-16 consists of a 32-stage LFSR with the feedback polynomial l(x) followed by a WG-16 transformation module with decimation d = 1057. Therefore, it can be regarded as a nonlinear filter generator over finite field $F_2{}^{16}$. WG-16 operates in two phases, including an initialization phase and a running phase. Xinxin et. al. [28] has applied the DFT attack while designing stream cipher WG-16. The attack version applied for the attack is Ronjom and Helleseth new algebraic attack on binary filtering generators over $F_2{}^m$. The complexity of the attack has been found out to be $2^{79.046}$ keystream bits after a pre-computation with a complexity of O($2^{97.96}$). They concluded that best against the WG-7 is the exhaustive search as attacker cannot obtain $2^{79.046}$ consecutive keystream bits due to exchange of information in 4G-LTE network.

2.4.2 Selective DFT Attack

Gong [7] showed a fast computation of DFT using the selective DFT algorithm. The same is helpful in simplifying selective DFT attack by fast calculation of any desired cosets DFT without going through the whole sequence DFT. Gong also introduced reference pairs to relate and simplify the application of selective DFT attacks. The selective DFT attack presented in [7] is the final version. Moreover, Gong alongwith Bo-Zhu also presented a method for application of DFT attack against Block Cipher and Hash Functions. The same concept has been tested on Toy cipher in Chapter 5.

2.4.3 Fast Selective DFT Attack

A new variation is introduced known as the fast selective DFT attack by Gong et. al. [6]. It is strongly related to the fast algebraic attacks in the literature. However, the variation is more efficient than other known methods for the case when the captured number of consecutive bits of a filter generator is less than the linear complexity of the sequence. The new attack version imposes a new condition for the design of cryptographic well-built Boolean functions, known as the spectral immunity of the sequence/Boolean function.

DFT attack has been applied to a version of Bluetooth encryption algorithm E_0 by Jingjing Wang et al.[8]. The keystream generator E_0 is part of the Bluetooth specifications [17] for wireless communications and consists of 4 regularly clocked LFSRs of lengths 25, 31, 33 and 39, respectively, yielding key bits of 128 bits. The attack operates by shifting and adding sequences to replace computations in higher order field. The attack is an improvement of the original attack launched by Helleseth et al. [1] due to requirement of the pseudorandom sequence succession and growing of complexity with the degree of the finite field where the discrete Fourier transform is done by solving equations in the finite field of some high degree with shifting and adding sequences. The attack complexity computation bases on the method mentioned in [1]. The results published in Chinese language for Shanghai Jiaotong University Journal in 2012. Moreover, success probability of the DFT attack [8][26] is described as $1-2^{-\varphi(2^n-1)}$ which is better than Ronjom-Helleseth attack[1] with $1-2^{-n}$.

Wang et al. [25] studied both the annihilator and the spectral immunity found the essential and ample condition for the spectral immunity which is lowest spectral weight of the annihilator of the sequence, measurement of the security of the cipher against DFT attack. They concluded that decision for existence of low spectral weight relations is difficult for DFT attack than Algebraic attack and cannot be a periodic sequence as annihilator for another sequence. Moreover, spectral immunity is bounded by one half period of the sequence for recovering any keystream. The evolution of the attack has been summarized in table 2.1.

Table 2.1 Summary of DFT Attack Evolution

Attack Evolution	**# required consecutive bits**	**# unknowns in equations**	**Degree(h)**	**Solvable**
Rønjom-Helleseth (06)	LS(**b**)	v	LS(**b**) − v not applicable if $D_1 = 0$	Solves a system of equations over F_2 in v unknowns, solve for all v unknowns
Rønjom-Gong-Helleseth (07)	LS(**b**)	v	LS(**b**) − v not applicable if gcd(k,N) ≠ 1	Solves a system of equations over F_2^n in v unknowns, obtaining one unknown is sufficient
New Case of Selective DFT	LS(**b**)	v	LS(**b**) − v not applicable if $\lvert C_k \rvert \neq v$	Solves a system of equations over F_2^n in v unknowns, obtaining one unknown is sufficient

2.5 Spectral Immunity and relation with Algebraic Attack

In this section spectral immunity has been defined which is analogous to Algebraic immunity. A relation with algebraic/Fast Algebraic attack is also discussed in this section.

2.5.1 Spectral Immunity

The algebraic immunity of a Boolean function has received an enormous interest in research on the efficiency of algebraic cryptanalysis. The spectral immunity of binary periodic sequences originally defined by Gong et. al.[6]. The spectral immunity is similar to the algebraic immunity, but instead of investigating the degree of the algebraic normal form (ANF) of a particular function, the spectral immunity depends on the linear complexity of the function. Hence, while algebraic immunity is a basic property of a Boolean function, the spectral immunity depends on the combination of a particular function with a basis generator. Various research has been carried out to ascertain the upper bound on Spectral Immunity [25][29][30]. Mostly proven fact is that spectral immunity is upper bounded by one half of the period of the cipher. From the bounds it is in need of attention that a univariate DFT attack involves fewer unknowns than in the multivariate case. In particular, the univariate representation provides a more natural origin for analyzing stream ciphers defined on a primary cyclic group.

2.5.2 Relation with Algebraic/Fast Algebraic Attacks

The selective DFT attack is always efficient than FAA in terms of requirement of successive key stream bits and the number of keystream bits. Moreover, it will work where the employed Boolean functions have high algebraic immunity. Selective DFT Attack works by multiplying boolean sequence $b_{\mathbf{t}}$ by a sequence, **e** = $\{e_t\}$, having the linear span less than the linear span of the sequence b_0, b_1,... to recover the key from the relation $g_t = b_t e_t$, t = 0, 1, ... The complexity of FAA is to solve a system of linear equations over $F_{2.}$ DFT Algorithm 2 needs to solve a system of linear equations over $F_2{}^n$ which can be converted into a system of linear equations over F_2

with fewer variables. The number of nonzero DFT spectra of $\{e_t\}$ remains constant always for all the shifts, which is in turn the linear complexity of e. In comparison to Algebraic attack, DFT attack always flourishes due to requirement of less number of key bits than Algebraic attack.

2.6 Summary

DFT attack is more successful on ciphers having a Non-linear filter function design in place. The evolution of DFT attack has been discussed in detail which gives account of various improvements made so far by the designer’s. A detailed review of the stream ciphers that have been tested and evaluated against the DFT attack is given in the chapter. The chapter also includes some important implementations of the DFT attack.

Chapter 3

Discrete Fourier Transform Attack

3.1 Introduction

The DFT attack is a key recovery attack in which recovers an initial state in a filtering sequence generator by reducing the number of unknowns in the system of linear equations to the degree of LFSR by an increase in complexity in precomputation/offline phase. The major complexity is in terms of forming a system of linear equations over $F_2{}^n$ with n unknowns instead of linear equations over F_2, where n is the degree of the LFSR. It subsequently reduces the number of the required consecutive bits of the filtering sequence to the linear complexity of the sequence. The required algebraic relations are converted into DFT form (Trace form) and then using captured bits equal to linear span of the cipher and associated variables initial state is recovered. The attack can also be performed when captured bits are less than linear span of the cipher using annihilators.

The Chapter 3 is divided into 3 sections. Section 3.2 contains the terminologies. Section 3.3 explains the theory of DFT attack including both types of DFT attacks alongwith their complexity calculations.

3.2 Terminologies

Following definitions are derived from work by Guang Gong [7]. All the work has been carried out in field based on characteristics of 2.

3.2.1 Definition 3.1

Let $c_j = (c_0, c_1, \ldots., c_{N-1})$, where $c_j \in F_2$ and represent binary sequence of length N. Since c_j is discrete, its Fourier transform is called discrete Fourier transform (DFT). It is also defined in terms of a finite field $GF(2^n)$, denoted as F_{2^n}, where n is the smallest number such that N | 2^n - 1. Let α be an element in F_{2^n} with order N. The Discrete Fourier Transform (DFT) of {c_j} is defined by

$$C_k = \sum_{j=0}^{N-1} c_j\, \alpha^{-jk}, k = 0,1 \ldots. N-1 \tag{3.1}$$

The inverse DFT, denoted as IDFT, is given by

$$c_j = \sum_{k=0}^{N-1} C_k\, \alpha^{jk}, j = 0,1 \ldots N-1 \tag{3.2}$$

The sequence {C_k} is called a DFT spectral sequence of c (with respect to α) or DFT spectra in short. The DFT spectral value C_k, in general, is an element in the extension field F_{2^n}, while {c_j} is a binary sequence.

3.2.2 Definition 3.2

Let $C(x) = \sum_{k=0}^{N-1} C_k\, x^k$ .Then c_j = C(α^j). C(x) can be written as

$$C(x) = \sum_k Tr_1^{n_k}(C_k x^k) \quad \&$$

$$c_j = \sum_k Tr_1^{n_k}(C_k \alpha^{jk}), j = 0,1, \ldots.., N-1 \tag{3.3}$$

Where the k's are coset leaders modulo N, n_k | n is the size of C_k, and $Tr_1^{n_k}(x)$ is a trace function from $F_{2^{n_k}}$ to F_2. This is termed as a trace representation of {c_j}.

3.3 Theory of DFT Attack

In this section, the basic concept of DFT attack is described for two situations. In one situation, captured data bits are equal to linear complexity of the cipher and in second, captured data bits are less than the linear complexity of the cipher. Both the situations have been described in Algorithms 1 and 2 respectively.

3.3.1 Algorithm 1 (Selective DFT Attack Algorithm) m=l(u)

This section describes the selective DFT attack algorithm. g(x) will be referenced as the characteristic polynomial of the LFSR which is also primitive. f(x) will be a boolean function in n variables, $x = (x_0, x_1, \dots, x_{n-1})$. Let {b_t} be an output sequence of the LFSR and {u_t} be the output of the filter function f(x) ,whose elements are given by u_t=f(b_t), b_t= (b_t, b_{t+1}, b_{t+2},……., b_{t+n-1}), t = 0,1,............,2^n-2.

3.3.1.1 Off-line computation

In offline computation first step is to compute the reference pair (o,s): Let {o_t} be generated with the initial state, o_t=Tr(α^t), t=0,1,2,3……., where $\mathrm{Tr}(x) = x + x^2 + \cdots + x^{2^{n-2}}$. . Then generate s_t = f(o_t), o_t = (o_t, o_{t+1}, $o_{t+2,}$…., o_{t+n-1}); t = 0, 1, …… ,n-1. The second step is to compute the selective filter d(x) which involves the computation of g(x), the minimal polynomial of s. The same can be obtained by running the BM algorithm to {s_t}, we get the minimal polynomial

$$g(x) = c_0x^L+c_{L-1}\, x^{L-1}+\dots +c_Lx^0 \tag{3.4}$$

Next step is to find the first k such that gcd (k, 2^n-1) = 1 and g(α^k) = 0, and compute k^{-1} mod 2^n -1. k are the cyclotomic cosets leaders of the LFSR period. For this k, $c_t = o_{tk}$ is computed, where t = 0,1,….., 2^n - 1. Applying the BM algorithm to {c_t}, g_k(x) is generated which helps to find d(x) as per equation 3.5:

$$d(x) = \frac{g(x)}{g_k(x)} = \sum_{i=0}^{l} c_i x^i \tag{3.5}$$

Where l =L-m, L is the linear span and m=captured data bits.

Fourth step is to compute S_k by the selective DFT algorithm in that set $M = \begin{pmatrix} c_0 & \cdots & c_{n-1} \\ \vdots & \ddots & \vdots \\ c_{n-1} & \cdots & c_{2n-2} \end{pmatrix}$. This matrix is called a circulant matrix generated by {c_t} and computes the time convolution of d(x) and {s_t}:

$$v_t = \sum_{i=0}^{l} d_i s_{i+t}, t = 0,1, \dots\dots, n-1 \quad (3.6)$$

Fifth step is to solve the following system of m linear equations in n variables$(x_0, x_1, \dots, x_{n-1})$ and calculate V and S_k:

$$\mathrm{M}\begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{pmatrix} = \begin{pmatrix} v_0 \\ v_1 \\ \vdots \\ v_{n-1} \end{pmatrix} \quad (3.7)$$

$$\mathrm{V} = \sum_{i=0}^{n-1} x_i \alpha^{ik} \quad (3.8)$$

$$S_k = V\,(d\,(\alpha^k))^{-1} \quad (3.9)$$

3.3.1.2 On-line computation

With the same $\{c_t\}$, M, and d(x) as those for computing S_k, U_k is computed in the online computation. Equation 3.10 computes the time convolution of d(x) and $\{u_t\}$:

$$\mathrm{w_t} = \sum_{i=0}^{l} d_i u_{i+t}, t = 0,1, \dots\dots, n-1 \quad (3.10)$$

Then system is solved for the unknowns $(x_0, x_1, \dots, x_{n-1})$ in the system of the linear equations by equation 3.11:

$$\mathrm{M}\begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{pmatrix} = \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_{n-1} \end{pmatrix} \quad (3.11)$$

$$\mathrm{W} = \sum_{i=0}^{n-1} x_i \alpha^{ik} \quad (3.12)$$

$$U_k = W{*}d(\alpha^k)^{-1} \quad (3.13)$$

$$\beta = (S_k^{-1}\, U_k)^{k^{-1}} \quad (3.14)$$

Finally the initial state is computed by using β calculated in equation 3.14 as

$$b_t = \mathrm{Tr}(\beta\alpha^t),\ t = 0,\ 1,\ \dots\dots,\ n-1 \quad (3.15)$$

3.3.1.3 Selective DFT Attack Example

The LFSR shown in Figure 3.1 has 5 stages and characteristic polynomial is $x^5 + x^3 + 1$ defined over F_2^5. Let $\alpha \in F_2^5$ satisfy $\alpha^5 + \alpha^3 + 1 = 0$ and filter function is defined as:

$$f = x_0 + x_0x_1 + x_0x_2 + x_0x_1x_2 + x_3 + x_1x_3 + x_2x_3 + x_1x_2x_3 + x_4 + x_0x_4 + x_1x_4 + x_0x_1x_4 + x_2x_4 + x_0x_2x_4 + x_1x_2x_4 + x_0x_3x_4 \quad (3.16)$$

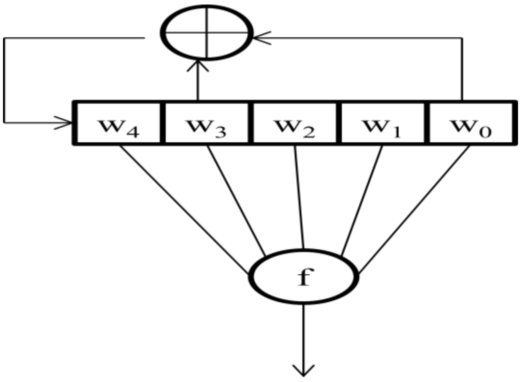


Figure 3.1 LFSR with Non-Linear Filter Generator

The 20 bits captured by the adversary to perform DFT attack are u_i: 11000100000001000000.

In offline computation, the reference pair (o,s) with $\{o_t\}$ is generated by characteristic polynomial with initial state as $o_t = Tr(\alpha^t)\,, t = 0,1,2,3,4 \Rightarrow (o_0, o_1, o_2, o_3, o_4) = (1,0,0,0,0)$. Keeping this initial state of the LFSR, values generated for sequence $\{o_t\}$ and $\{s_t\}$ are: 1000010101110110001111100110100 and 1110000101100010000000100000000 respectively.

Selective filter d(x) is calculated next. There is a need to find g(x) the minimal polynomial of s. BM algorithm is run to $\{s_t\}$ to get the equation 3.17 with a linear span of 20 owing to degree of the equation:

$$g(x) = x^{20} + x^{16} + x^{15} + x^{13} + x^{10} + x^9 + x^8 + x^7 + x^4 + x^3 + 1 \quad (3.17)$$

The coset leaders 1, 3, 5, 7, 11 and 15 are relatively prime with 31, so upon checking $g(\alpha) \neq 0\, and\, g(\alpha^3) = 0$. So the selected value of k = 3. Then computation of c$_t$ is carried out by decimation of sequence o$_t$: $c_t = o_{3t},\ t = 0,1, \dots 2n - 1$, where n=5 and $c_0, \dots., c_9 =$ (1001001100). Applying BM algorithm to c$_{t,}$ equation 3.18 generates:

$$g_3(x) = x^5 + x^3 + x^2 + x + 1 \tag{3.18}$$

Selective filter is computed by using equation 3.5, equation 3.17 and 3.18 as:

$$d(x) = x^{15} + x^{13} + x^{12} + x^{11} + x^9 + x^7 + x^5 + x^4 + x^3 + x + 1 \tag{3.19}$$

$$d(\alpha^3) = \alpha^{18} and\ T = (d(\alpha^3))^{-1} = \alpha^{13} \tag{3.20}$$

DFT coefficient for S$_3$ is calculated by generation of circulant Matrix M from c$_t$ as:

$$M = \begin{pmatrix} c_0 & c_1 & c_2 & c_3 & c_4 \\ c_1 & c_2 & c_3 & c_4 & c_5 \\ c_2 & c_3 & c_4 & c_5 & c_6 \\ c_3 & c_4 & c_5 & c_6 & c_7 \\ c_4 & c_5 & c_6 & c_7 & c_8 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix} \tag{3.21}$$

Time convolution of d(x) and {s$_t$} is calculated using equation 3.6 as $(v_0, v_1, v_2, v_3, v_4) = (0,0,0,1,0)$. Solving the system of equation in accordance with equation 3.7 computes the value of x$_i$ as $(x_0, x_1, x_2, x_3, x_4) = (0,1,0,0,1)$. The value of V is computed using equation 3.8 as α^{27} and the value of S$_3$ is computed using equation 3.9 as α^9.

U$_3$ is calculated online by first computing time convolution of d(x) and {s$_t$} using equation 3.10 as $(w_0, w_1, w_2, w_3, w_4) = (0,1,0,0,0)$. Solving the system of equation in accordance with equation 3.11 computes the value as$(x_0, x_1, x_2, x_3, x_4) = (1,0,1,1,0)$. The value of W is computed by using equation 3.12 is α^{23} and the value of U$_3$ is computed by using equation 3.13 is α^5. β is calculated using equation 3.14 as

$\beta = (\alpha^{22}.\alpha^{5})^{3^{-1} mod\, 31} = \alpha^{9}$. The initial state of the LFSR using equation 3.15 is recovered as (1, 0, 1, 1, 1).

3.3.1.4 Complexity Calculations

The preprocessing phase is a one-time effort and its complexity is $O(l(s)[n\,(\log n)^2 + (\log l(s))^3 + \eta(n)])$ and $\eta(n) = n \log_2 n \log_2 \log_2 n$. Where l(s) is linear span of the captured bits and n is the cryptosystem initial state bits. The complexity for the online phase of the attack is calculated as $O(l(s) + n_k \log(n_k)\eta(n_k))$. The Preprocessing or precomputations phase complexity for section 3.3.1.3 example is O($2^{8.5}$). The complexity for online phase is O($2^{4.7}$).

3.3.2 Algorithm 2 Fast Selective DFT Method for m < l(u)

This algorithm recovers the scalar factor β where the number of observed consecutive bits of a filter generator is less than the linear complexity of the sequence. The fast selective method is used in DFT attack to extract the shift β to recover the initial state of the cipher. The bits where the output sequence output one are filtered and their relative positions used to develop the system of equations. It is used to replace β with the initial state of the cipher in terms of polynomial to develop the systems of equations for solution. The trace is also carried out to find the elements of subfield F_2 from F_2^n. Then the value of variables is found to recover β to recover initial state.

3.3.2.1 Off-line computation

Select a sequence g = {g_t} and h = {$u_t.g_t$} which satisfy the following condition: |D_g U D_h| < l(u). Where D_g = {k |G_k≠0, k is a coset leader mod D}. Then compute p(x) from the known keystream bits u using BM Algorithm. Then compute the following polynomials:

$$\mathrm{q(x)} = \prod_{k\in D_h \setminus D_g} p_k(x) \tag{3.22}$$

$$\mathrm{q}(\alpha^k) = \prod_{k\in D_h \setminus D_g} p_k(\alpha^k) \tag{3.23}$$

Denoting q(x) by equation 3.24

$$q(x) = \sum_{i=0}^{r} c_i x^i \tag{3.24}$$

3.3.2.2 On-line computation

For $t = 0,1, \ldots., l(g) - 1$, using the known bits $u_0, \ldots\ldots., u_{m-1}$, compute $f_t(\alpha^k)$ for $k\in D_g$, where

$$f_t(x) = \sum_{i=0}^{r} c_i u_{i+t} x^i \tag{3.25}$$

Then applying q(L) to h results in equation 3.26:

$$\sum_{k\in \mathrm{Dg}} Tr(\beta_k(G_k f_t(\alpha^k) + H_k q(\alpha^k))\alpha^{tk}) = 0 \tag{3.26}$$

$$z_t = \sum_{k\in \mathrm{Dg}} Tr(\beta_k G_k f_t(\alpha^k)\alpha^{tk}) = 0 \tag{3.27}$$

Thereafter the coefficient matrix is computed by replacing $G_k\alpha^k$ with

$$G_k\alpha^k = x_0 + x_1\alpha + x_2\alpha^2 + \ldots\ldots. + x_{n-1}\alpha^{n-1} \tag{3.28}$$

$$\mathrm{A} = \begin{pmatrix} z_0 & \cdots & z_{n-1} \\ \vdots & \ddots & \vdots \\ z_{l(g)-1} & \cdots & z_{(n-1)(l(g)-1)} \end{pmatrix} \tag{3.29}$$

$$\mathrm{A}\begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{3.30}$$

The independent system of equations is generated which produces a unique solution. To recover bits after initialization stage, above system of $l(g) - 1$ linear equations over F_2 with $l(g) - 1$ variables are required to be solved.

If h=0, then this is the same case as Algebraic attack. All k are replaced with t where keystream u_t is equal to 1. Then there is no need to calculate $f_t(x)$. Function g is used to perform the rest of the process.

$$z_t = \sum_{k\in \mathrm{Dg}} Tr(G_k x) = 0, x = \alpha^k\beta \tag{3.31}$$

If there is a k∈D_g with gcd (k,D) = 1, then return β = ($β^k$)k’ , where k’ = k^{-1} (mod D),otherwise, return {$β^k$ | k∈D_g}. In Algorithm 2, the number of required consecutive bits from u is at the most l(h) + l(g).

3.3.2.3 Fast Selective DFT Attack Example

The LFSR shown in Figure 3.2 has 5 stages and generating polynomial for F_2^5 is $x^5 + x^3 + 1$. Let $\alpha \in F_2^5$ satisfy $\alpha^5 + \alpha^3 + 1 = 0$, and filter function is defined as:

$$f = x_0 + x_0x_1 + x_0x_2 + x_0x_1x_2 + x_3 + x_1x_3 + x_2x_3 + x_1x_2x_3 + x_4 + x_0x_4 + x_1x_4 + x_0x_1x_4 + x_2x_4 + x_0x_2x_4 + x_1x_2x_4 + x_0x_3x_4 \quad (3.32)$$

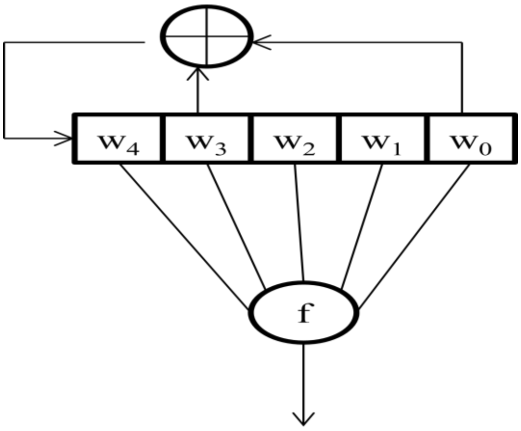


Figure 3.2 LFSR with Non-Linear Filter Generator

The bits captured by the adversary to perform fast selective DFT attack are u_i: 11000100000001000000000111. The initial state of the LFSR is:

$$w_0 = Tr(\beta)\,,\ w_1 = Tr(\alpha\beta)\,, w_2 = Tr(\alpha^2\beta)\,, w_3 = Tr(\alpha^3\beta)\,, w_4 = Tr(\alpha^4\beta) \quad (3.33)$$

The trace representation of {u_i} is

$$U(x) = Tr(\alpha^{27}x + \alpha^9x^3 + \alpha^{14}x^7 + \alpha^7x^{11}) \quad (3.34)$$

To launch this attack there is need to find the relation that satisfies this condition of U(x). G(x) = 0. From computer search, annihilator found is mentioned in equation 3.35 as:

$$g = x_2 + x_0x_2 + x_1x_2 + x_0x_3 + x_1x_3 + x_1x_4 + x_3x_4 \quad (3.35)$$

The trace form of the annihilator is:

$$G(x) = Tr(\alpha^{29}x^5) \tag{3.36}$$

To find out the initial state, β needs to be found. For this purpose captured keystream bits are being utilized. Because u$_5$= u$_{13}$= u$_{22}$= u$_{23}$= u$_{24}$=1, $Tr(\alpha^{29}x^5) = 0$ for $x = \alpha^5\beta, \alpha^{13}\beta, \alpha^{22}\beta, \alpha^{23}\beta, \alpha^{24}\beta$.

$$\begin{pmatrix} Tr(\alpha^{23}\beta^5) \\ Tr(\alpha\beta^5) \\ Tr(\alpha^{15}\beta^5) \\ Tr(\alpha^{20}\beta^5) \\ Tr(\alpha^{25}\beta^5) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{3.37}$$

β^5 is replaced as

$$\beta^5 = x_0 + \alpha x_1 + \alpha^2 x_2 + \alpha^3 x_3 + \alpha^4 x_4 \tag{3.38}$$

System of equation is generated using equation 3.38 as:

$$\begin{pmatrix} Tr(\alpha^{23}(x_0 + \alpha x_1 + \alpha^2 x_2 + \alpha^3 x_3 + \alpha^4 x_4)) \\ Tr(\alpha(x_0 + \alpha x_1 + \alpha^2 x_2 + \alpha^3 x_3 + \alpha^4 x_4)) \\ Tr(\alpha^{15}(x_0 + \alpha x_1 + \alpha^2 x_2 + \alpha^3 x_3 + \alpha^4 x_4)) \\ Tr(\alpha^{20}(x_0 + \alpha x_1 + \alpha^2 x_2 + \alpha^3 x_3 + \alpha^4 x_4)) \\ Tr(\alpha^{25}(x_0 + \alpha x_1 + \alpha^2 x_2 + \alpha^3 x_3 + \alpha^4 x_4)) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{3.39}$$

From the linearity of the trace function, following system of equation is developed:

$$\begin{pmatrix} Tr(\alpha^{23}) & Tr(\alpha^{24}) & Tr(\alpha^{25}) & Tr(\alpha^{26}) & Tr(\alpha^{27}) \\ Tr(\alpha) & Tr(\alpha^{2}) & Tr(\alpha^{3}) & Tr(\alpha^{4}) & Tr(\alpha^{5}) \\ Tr(\alpha^{15}) & Tr(\alpha^{16}) & Tr(\alpha^{17}) & Tr(\alpha^{18}) & Tr(\alpha^{19}) \\ Tr(\alpha^{20}) & Tr(\alpha^{21}) & Tr(\alpha^{22}) & Tr(\alpha^{23}) & Tr(\alpha^{24}) \\ Tr(\alpha^{25}) & Tr(\alpha^{26}) & Tr(\alpha^{27}) & Tr(\alpha^{28}) & Tr(\alpha^{29}) \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

By evaluation of the trace function, system is solved with following variables as

$$\begin{pmatrix} 0 & 0) & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

The non zero solution is

$$\begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Substituting the value in equation 3.38, value of the β is attained. Which is

$$\beta = \alpha^{14}$$

Hence, the recovered initial state from equation 3.33 is w_0=1, w_1=1, w_2=1, w_3=0, w_4=1.

3.3.2.4 Complexity Calculations

The preprocessing phase is a one-time effort like Algebraic Attack or Fast Algebraic Attack. The complexity for the online phase of the attack is calculated for case where product of original function sequence and Annihilator sequence is equal to zero. Then computation of $\beta_k G_k \alpha^{tk}$ has a complexity of $O(2\eta(n-1))$ exclusive-OR operations in F_2. The system of equations can be efficiently solved with complexity of $l(c)\log(l(c)\,\eta l(c))$.The total complexity is $O(2\eta(n-1) + l(c)\log l(c)\,\eta(l(c))))$. The section 3.3.2.3 example total complexity is O($2^{6.5}$). It requires 9 keystream bits for solving 5 independent equations to recover the initial state.

3.4 Summary

In this chapter the DFT attack description and methodology has been described. The DFT attack complete process has been followed by taking examples for both algorithms for better understanding. The actual procedure has been carried out by going through algorithms and example. At the end, the complexity of the DFT attack is calculated with a brief description.

Chapter 4

Discrete Fourier Transform Attack on WG-7

4.1 Introduction

In this chapter, details of DFT attack on WG-7 are included. The maple version 15 has been used to generate the results as per DFT attack methods mentioned in the literature [6][7]. DFT attack has not been carried out on WG-7 cipher except comments made on complexity of DFT attack by the designer's [9]. The complexity calculation has been based on the algorithm specified in [2].

This section is organized as following: In Section 4.2, specification of WG-7 has been described in detail. Section 4.3 gives a small account of algebraic and fast algebraic attack against WG-7. Section 4.4 contains the results of the DFT attack against WG-7 and Section 4.5 contains the modified DFT attack methods against WG-7.

4.2 WG-7

WG-7 is a synchronous stream cipher, designed by Yiyuan Luo, Qi Chai, Guang Gong and Xuejia Lai in 2010[9]. The cipher has two steps: the initialization step and the keystream generation step. WG-7 has 80-bit key and 81-bit initial value (IV). The cipher is designed as light weight cipher for RFID tags application.

4.2.1 Specifications

WG-7 is designed to generate up to 2^{24} bits of key stream from an 80-bit key length and an 81-bit initialization vector. The parameters are shown in Table 4.1. The internal state of 161 bits $[s_1, \ldots, s_{161}]$ is divided into one LFSRs of length 23, containing 7 bit word each. The cipher consists of a 23 stage LFSR over F_{2^7} and a

WG linear transformation. The finite field is defined by primitive polynomial $g(x) = x^7 + x + 1$. The characteristic polynomial is primitive over F_{2^7} and is given by f(x) = $x^{23} + x^{11} + \alpha$, where α is root of g(x). It has the ideal two-level autocorrelation property and offers better security against algebraic attack. The non-linear transformation is defined by equation 4.1:-

$$\mathrm{WG7(x)} = \mathrm{Tr}(x^3 + x^9 + x^{21} + x^{57} + x^{87}), x \in F_{2^7} \qquad (4.1)$$

Table 4.1 Parameters of WG-7

Key Length	**IV Length**	**Internal State**
80-bit	81-bit	161-bit

The structure of the cipher is shown in Figure 4.1.

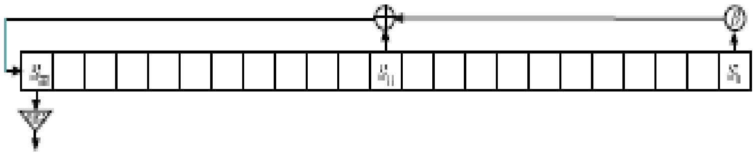


Figure 4.1 WG-7

4.3 Application of Algebraic and Fast Algebraic Attack on WG-7

Mohammad Ali Orumiehchiha et. al.[9] launched algebraic attack and fast algebraic attacks to recover the initial state of the cipher. The degree of the filter function has been reduced by the introduction of annihilators for both algebraic and fast algebraic attack. The algebraic attack has been predicted with time complexity of $2^{54.36}$ and memory complexity of $2^{19.38}$. Fast algebraic attack has been predicted with time complexity of $2^{26.73}$, data complexity of $2^{19.38}$, memory complexity of $2^{14.66}$ and pre-computation of $2^{26.87}$.

4.4 Application of Discrete Fourier Transform Attack on WG-7

This section presents various versions of the DFT attack launched against WG-7 to check their efficacy.

4.4.1 Application of Selective DFT Attack on WG-7

Selective DFT attack cannot be applied on practical ciphers as linear complexity is equal to complete period of the cipher. The WG-7 cipher has the period equal to 2^{161}. Although the linear complexity has reduced due to the design of cipher by a factor as mentioned in Ronjom et al. [2], still it is enough to restrict the selective DFT attack.

4.4.2 Application of Helleseth et al. [2] New Attack on the Cipher

Ronjom and Helleseth [2] proposed a variation of algebraic attack on binary filter generators. The attack make use of E keystream bits with a complexity O(E), where E is linear complexity of the keystream $E = \sum_{j=1}^{e} \binom{n}{j}$ and e is the degree of the boolean function f. The pre-computation complexity of the attack is $O(E(Log_2E)^3)$. For WG-7, the attack requires $2^{25.5}$ keystream bits with a complexity of $O(2^{25.5})$ with a precomputations complexity of $O(2^{39.5})$. If the length of keystream captured is less than 2^{25} then the rest of bits $2^{25.5} - 2^{25} > 2^{23}$ have to be guessed by the attacker to launch this attack. The DFT attack has been rejected in the last assessment made in [9] that attacker cannot obtain cannot obtain 2^{24} bits in succession thereby it is expensive than brute force attack. Ronjom et. al.[2] new attack cannot be applied using the annihilator due to requirement of keybits in succession and only 1's in captured stream can be converted into 0. The 0's in the stream have to be guessed for two possible values (0 or 1). This guess work adds more complexity in the DFT attack.

4.4.3 Application of Fast Selective DFT Attack using Annihilator

The DFT attack mentioned in [6] has been extended to WG-7 Cipher using the annihilator discovered in [10]. The annihilator is discovered for launching of the algebraic attack against the cipher with an algebraic immunity of 3. The algebraic normal form (ANF) of the filter function is mentioned as equation 4.2:

$$g(y_1,\ldots,y_7) = y_1 + y_1y_3 + y_2y_3 + y_4 + y_1y_4 + y_2y_4 + y_1y_2y_4 + y_3y_4 + y_1y_3y_4 + y_1y_2y_3y_4 + y_1y_3y_5 + y_4y_5 + y_1y_2y_4y_5 + y_1y_2y_3y_4y_5 + y_6 + y_2y_6 + y_1y_2y_6 + y_1y_2y_3y_6 + y_1y_2y_4y_6 + y_1y_2y_3y_4y_6 + y_1y_5y_6 + y_3y_5y_6 + y_1y_4y_5y_6 + y_3y_4y_5y_6 + y_7 + y_2y_7 + y_1y_2y_7 + y_2y_3y_7 + y_1y_4y_7 + y_1y_2y_4y_7 + y_1y_2y_3y_4y_7 + y_5y_7 + y_1y_5y_7 + y_1y_3y_5y_7 + y_1y_2y_3y_5y_7 + y_2y_4y_5y_7 + y_2y_3y_4y_5y_7 + y_6y_7 + y_1y_2y_6y_7 + y_1y_3y_6y_7 + y_1y_2y_3y_6y_7 + y_2y_4y_6y_7 + y_1y_3y_4y_6y_7 + y_2y_3y_4y_6y_7 + y_5y_6y_7 + y_2y_5y_6y_7 + y_1y_2y_5y_6y_7 + y_2y_3y_5y_6y_7 + y_1y_4y_5y_6y_7 + y_3y_4y_5y_6y_7. \tag{4.2}$$

4.4.3.1 Transformation of Annihilator into DFT Form

The Annihilator is transformed from Polynomial form into DFT form by using maple platform. The algebraic form obtained from [10] is mentioned as equation 4.3:

$$h(y_1,\ldots,y_7) = 1 + y_1 + y_3 + y_1y_2y_3 + y_4 + y_1y_4 + y_2y_4 + y_1y_2y_4 + y_3y_4 + y_1y_3y_4 + y_2y_3y_4 + y_1y_3y_5 + y_4y_5 + y_1y_4y_5 + y_3y_4y_5 + y_6 + y_1y_6 + y_2y_6 + y_1y_2y_6 + y_3y_6 + y_2y_3y_6 + y_7 + y_3y_7 + y_1y_3y_7 + y_2y_3y_7 + y_4y_7 + y_2y_4y_7 + y_3y_4y_7 + y_3y_5y_7 + y_4y_5y_7 + y_6y_7 + y_1y_6y_7 + y_2y_6y_7 + y_3y_6y_7. \tag{4.3}$$

The DFT form of annihilator has been obtained by using the method mentioned in [3]. The same is repeated here for clarity by using the example.

Example 1: A filter generator consisting of an m-sequence v = $\{v_t\}$ has been designed which is produced by

$$h(x) = x^3 + x + 1 \in F_2 \tag{4.4}$$

and filtered through this boolean function

$$f(x_0, x_2) = x_0x_2 \tag{4.5}$$

with deg(f) = 2. Let (e_0,e_1) = (0,2) are the tapping positions from the register such that the keystream sequence a_t is given by

$$a_t = f_t(s_0, s_1, s_2) = f\left(s_{t+e_0}, s_{t+e_1}\right) = s_t s_{t+2} \tag{4.6}$$

The cyclotomic cosets modulo 3 are

C_1 = {1, 2, 4}

C_3 = {3, 6, 5}

Now {B_k} is computed for k= {1,3}. For k = 1, value of v = (v_0, v_1, v_2) is required that satisfies the condition $\sum_{i=0}^{2} v_i 2^i \equiv 1(\text{mod } 7)$ and $\sum_{i=0}^{2} v_i = 2$. There is only one v that satisfies these conditions, which is v = (0, 0, 2), so Equation 4.7 is computed:-

$$\mathrm{B}_1 = \alpha^{4\mathrm{e}_0+4\mathrm{e}_1} \tag{4.7}$$

$$\mathrm{B}_1 = \alpha^8 = \alpha$$

For k = 3, value of v = (v_0, v_1, v_2) is required that satisfies the condition $\sum_{i=0}^{2} v_i 2^i \equiv 3(\text{mod } 7)$ and $\sum_{i=0}^{2} v_i = 2$. There is only one v that satisfies these conditions, which is v = (1, 1, 0), so Equation 4.8 is computed:-

$$\mathrm{B}_3 = \alpha^{\mathrm{e}_0+2\mathrm{e}_1} + \alpha^{2\mathrm{e}_0+\mathrm{e}_1} \tag{4.8}$$

$$\mathrm{B}_3 = \alpha^4 + \alpha^2 = \alpha$$

The sequence can be written as

$$a_t = \sum_{k\in T(2)} Tr_1^{n_k}(\mathrm{B_k}x^k) \tag{4.9}$$

$$a_t = Tr_1^3(\mathrm{B}_1x + \mathrm{B}_3x^3)$$

$$a_t = Tr_1^3(\alpha x + \alpha x^3)$$

The method is automated by using the Maple code mentioned in CD attached with the thesis. Each degree relations has been evaluated separately and generated by $\boldsymbol{x^7 + x + 1}$ as the base polynomial is represented using cyclotomic cosets of $GF(2^7)$. The cyclotomic cosets of $GF(2^7)$ are 1, 3, 5, 7, 9, 11, 13, 15, 19, 21, 23, 27, 29, 31, 43, 47, 55 and 63:

$B_1 = \alpha^{97} + \alpha^{96} + \alpha^{69} + \alpha^{67} + \alpha^{66} + \alpha^{65} + \alpha^{35} + \alpha^{33} + \alpha^{6} + \alpha^{5} + \alpha^{4} + \alpha^{3} + 1$

(4.10)

$B_3 = \alpha^{76} + \alpha^{75} + \alpha^{73} + \alpha^{72} + \alpha^{69} + \alpha^{68} + \alpha^{66} + \alpha^{65} + \alpha^{17} + \alpha^{16} + \alpha^{15} +$

$\alpha^{14} + \alpha^{12} + \alpha^{11} + \alpha^{10} + \alpha^{7} + \alpha^{6} + \alpha^{5} + \alpha^{3} + \alpha^{2}$ (4.11)

$B_5 = \alpha^{89} + \alpha^{87} + \alpha^{85} + \alpha^{82} + \alpha^{81} + \alpha^{79} + \alpha^{71} + \alpha^{67} + \alpha^{66} + \alpha^{65} + \alpha^{29} +$

$\alpha^{27} + \alpha^{26} + \alpha^{25} + \alpha^{23} + \alpha^{18} + \alpha^{17} + \alpha^{16} + \alpha^{15} + \alpha^{11} + \alpha^{10} + \alpha^{6} + \alpha^{3} + \alpha^{2}$

(4.12)

$B_7 = \alpha^{36} + \alpha^{32} + \alpha^{29} + \alpha^{28} + \alpha^{26} + \alpha^{25} + \alpha^{22} + \alpha^{21} + \alpha^{20} + \alpha^{18} + \alpha^{14} +$

$\alpha^{13} + \alpha^{10} + \alpha^{8} + \alpha^{6} + \alpha^{4}$ (4.13)

$B_9 = \alpha^{113} + \alpha^{107} + \alpha^{105} + \alpha^{104} + \alpha^{100} + \alpha^{98} + \alpha^{97} + \alpha^{91} + \alpha^{89} + \alpha^{88} + \alpha^{85} +$

$\alpha^{84} + \alpha^{81} + \alpha^{80} + \alpha^{77} + \alpha^{76} + \alpha^{75} + \alpha^{73} + \alpha^{69} + \alpha^{67} + \alpha^{66} + \alpha^{65} + \alpha^{53} +$

$\alpha^{51} + \alpha^{49} + \alpha^{45} + \alpha^{41} + \alpha^{38} + \alpha^{33} + \alpha^{32} + \alpha^{31} + \alpha^{30} + \alpha^{29} + \alpha^{28} + \alpha^{20} +$

$\alpha^{18} + \alpha^{17} + \alpha^{15} + \alpha^{12} + \alpha^{9} + \alpha^{7} + \alpha^{6} + \alpha^{5} + \alpha^{4} + \alpha^{3} + \alpha^{2}$ (4.14)

$B_{11} = \alpha^{60} + \alpha^{58} + \alpha^{57} + \alpha^{55} + \alpha^{54} + \alpha^{52} + \alpha^{48} + \alpha^{47} + \alpha^{45} + \alpha^{41} + \alpha^{39} +$

$\alpha^{38} + \alpha^{37} + \alpha^{36} + \alpha^{35} + \alpha^{34} + \alpha^{33} + \alpha^{32} + \alpha^{31} + \alpha^{30} + \alpha^{28} + \alpha^{26} + \alpha^{24} +$

$\alpha^{23} + \alpha^{22} + \alpha^{21} + \alpha^{20} + \alpha^{19} + \alpha^{18} + \alpha^{17} + \alpha^{15} + \alpha^{13} + \alpha^{12} + \alpha^{11} + \alpha^{7} + \alpha^{4}$

(4.15)

$B_{13} = \alpha^{70} + \alpha^{69} + \alpha^{68} + \alpha^{67} + \alpha^{65} + \alpha^{62} + \alpha^{60} + \alpha^{58} + \alpha^{56} + \alpha^{55} + \alpha^{53} +$

$\alpha^{52} + \alpha^{46} + \alpha^{45} + \alpha^{44} + \alpha^{40} + \alpha^{36} + \alpha^{35} + \alpha^{34} + \alpha^{33} + \alpha^{32} + \alpha^{26} + \alpha^{23} +$

$\alpha^{22} + \alpha^{20} + \alpha^{19} + \alpha^{18} + \alpha^{17} + \alpha^{12} + \alpha^{10} + \alpha^{7} + \alpha^{6}$ (4.16)

$B_{19} = \alpha^{108} + \alpha^{106} + \alpha^{105} + \alpha^{103} + \alpha^{101} + \alpha^{98} + \alpha^{94} + \alpha^{93} + \alpha^{92} + \alpha^{90} +$

$\alpha^{88} + \alpha^{86} + \alpha^{85} + \alpha^{84} + \alpha^{82} + \alpha^{81} + \alpha^{79} + \alpha^{78} + \alpha^{76} + \alpha^{74} + \alpha^{72} + \alpha^{71} +$

$\alpha^{70} + \alpha^{68} + \alpha^{67} + \alpha^{66} + \alpha^{64} + \alpha^{61} + \alpha^{56} + \alpha^{53} + \alpha^{52} + \alpha^{47} + \alpha^{46} + \alpha^{45} +$

$$\alpha^{42} + \alpha^{37} + \alpha^{36} + \alpha^{35} + \alpha^{34} + \alpha^{32} + \alpha^{31} + \alpha^{30} + \alpha^{25} + \alpha^{24} + \alpha^{23} + \alpha^{22} + \alpha^{21} + \alpha^{20} + \alpha^{19} + \alpha^{18} + \alpha^{17} + \alpha^{16} + \alpha^{14} + \alpha^{10} + \alpha^{7} + \alpha^{4} \tag{4.17}$$

$$B_{21} = \alpha^{118} + \alpha^{117} + \alpha^{116} + \alpha^{115} + \alpha^{114} + \alpha^{112} + \alpha^{110} + \alpha^{108} + \alpha^{107} + \alpha^{106} + \alpha^{105} + \alpha^{103} + \alpha^{102} + \alpha^{101} + \alpha^{98} + \alpha^{94} + \alpha^{91} + \alpha^{89} + \alpha^{84} + \alpha^{82} + \alpha^{81} + \alpha^{75} + \alpha^{74} + \alpha^{73} + \alpha^{72} + \alpha^{70} + \alpha^{67} + \alpha^{64} + \alpha^{62} + \alpha^{61} + \alpha^{59} + \alpha^{53} + \alpha^{51} + \alpha^{49} + \alpha^{44} + \alpha^{43} + \alpha^{42} + \alpha^{41} + \alpha^{39} + \alpha^{36} + \alpha^{35} + \alpha^{34} + \alpha^{33} + \alpha^{28} + \alpha^{27} + \alpha^{24} + \alpha^{16} + \alpha^{13} + \alpha^{12} + \alpha^{11} + \alpha^{7} + \alpha^{6} \tag{4.18}$$

Combining all the cyclotomic cosets to represent the sequence using equation 4.9 for T=127, the DFT form for Annihilator of WG-7 is mentioned in Equation 4.19 as:

$$g = \alpha^{3}x + \alpha^{10}x^{3} + \alpha^{30}x^{5} + \alpha^{78}x^{7} + \alpha^{113}x^{9} + \alpha^{19}x^{11} + \alpha^{31}x^{13} + \alpha^{33}x^{19} + \alpha^{97}x^{21} \tag{4.19}$$

4.4.3.2 Results of the Discrete Fourier Transform Attack

Applying the attack mentioned in [6] and section 3.3.2 with the use of Equation 4.19 annihilator recovers 7 bits of keybits out of 161 bits by solving independent relations. The method is automated by using the Maple code mentioned in CD attached with the thesis. The independent relations are mentioned in Table 4.2:

Table 4.2 Equations Generated for Fast Selective DFT Attack on WG-7

Indexes	Equation
4	$\alpha^{3}\beta+\alpha^{10}\beta^{3}+\alpha^{30}\beta^{5}+\alpha^{78}\beta^{7}+\alpha^{113}\beta^{9}+\alpha^{19}\beta^{11}+\alpha^{31}\beta^{13}+\alpha^{33}\beta^{19}+\alpha^{97}\beta^{21}+1$
7	$\alpha^{6}\beta+\alpha^{19}\beta^{3}+\alpha^{45}\beta^{5}+\alpha^{99}\beta^{7}+\alpha^{13}\beta^{9}+\alpha^{52}\beta^{11}+\alpha^{70}\beta^{13}+\alpha^{90}\beta^{19}+\alpha^{33}\beta^{21}+1$
10	$\alpha^{9}\beta+\alpha^{28}\beta^{3}+\alpha^{60}\beta^{5}+\alpha^{120}\beta^{7}+\alpha^{40}\beta^{9}+\alpha^{85}\beta^{11}+\alpha^{109}\beta^{13}+\alpha^{20}\beta^{19}+\alpha^{96}\beta^{21}+1$
11	$\alpha^{10}\beta+\alpha^{31}\beta^{3}+\alpha^{65}\beta^{5}+\beta^{7}+\alpha^{49}\beta^{9}+\alpha^{96}\beta^{11}+\alpha^{122}\beta^{13}+\alpha^{39}\beta^{19}+\alpha^{117}\beta^{21}+1$
15	$\alpha^{11}\beta+\alpha^{34}\beta^{3}+\alpha^{70}\beta^{5}+\alpha^{7}\beta^{7}+\alpha^{58}\beta^{9}+\alpha^{107}\beta^{11}+\alpha^{8}\beta^{13}+\alpha^{58}\beta^{19}+\alpha^{11}\beta^{21}+1$
17	$\alpha^{9}\beta+\alpha^{28}\beta^{3}+\alpha^{60}\beta^{5}+\alpha^{120}\beta^{7}+\alpha^{40}\beta^{9}+\alpha^{85}\beta^{11}+\alpha^{109}\beta^{13}+\alpha^{20}\beta^{19}+\alpha^{96}\beta^{21}+1$

21	$\alpha^{12}\beta+\alpha^{37}\beta^{3}+\alpha^{75}\beta^{5}+\alpha^{14}\beta^{7}+\alpha^{67}\beta^{9}+\alpha^{118}\beta^{11}+\alpha^{21}\beta^{13}+\alpha^{77}\beta^{19}+\alpha^{32}\beta^{21}+1$
23	$\alpha^{16}\beta+\alpha^{49}\beta^{3}+\alpha^{95}\beta^{5}+\alpha^{42}\beta^{7}+\alpha^{103}\beta^{9}+\alpha^{35}\beta^{11}+\alpha^{73}\beta^{13}+\alpha^{26}\beta^{19}+\alpha^{116}\beta^{21}+1$
37	$\alpha^{26}\beta+\alpha^{79}\beta^{3}+\alpha^{18}\beta^{5}+\alpha^{112}\beta^{7}+\alpha^{66}\beta^{9}+\alpha^{18}\beta^{11}+\alpha^{76}\beta^{13}+\alpha^{89}\beta^{19}+\alpha^{72}\beta^{21}+1$
49	$\alpha^{40}\beta+\alpha^{121}\beta^{3}+\alpha^{88}\beta^{5}+\alpha^{83}\beta^{7}+\alpha^{65}\beta^{9}+\alpha^{45}\beta^{11}+\alpha^{4}\beta^{13}+\alpha^{101}\beta^{19}+\alpha^{112}\beta^{21}+1$
57	$\alpha^{27}\beta+\alpha^{82}\beta^{3}+\alpha^{23}\beta^{5}+\alpha^{119}\beta^{7}+\alpha^{75}\beta^{9}+\alpha^{29}\beta^{11}+\alpha^{89}\beta^{13}+\alpha^{108}\beta^{19}+\alpha^{93}\beta^{21}+1$
60	$\alpha^{68}\beta+\alpha^{78}\beta^{3}+\alpha^{101}\beta^{5}+\alpha^{25}\beta^{7}+\alpha^{63}\beta^{9}+\alpha^{99}\beta^{11}+\alpha^{114}\beta^{13}+\alpha^{125}\beta^{19}+\alpha^{65}\beta^{21}+1$
73	$\alpha^{58}\beta+\alpha^{48}\beta^{3}+\alpha^{51}\beta^{5}+\alpha^{82}\beta^{7}+\alpha^{100}\beta^{9}+\alpha^{116}\beta^{11}+\alpha^{111}\beta^{13}+\alpha^{62}\beta^{19}+\alpha^{109}\beta^{21}+1$
95	$\alpha^{80}\beta+\alpha^{114}\beta^{3}+\alpha^{34}\beta^{5}+\alpha^{109}\beta^{7}+\alpha^{44}\beta^{9}+\alpha^{104}\beta^{11}+\alpha^{16}\beta^{13}+\alpha^{99}\beta^{19}+\alpha^{63}\beta^{21}+1$
105	$\alpha^{52}\beta+\alpha^{30}\beta^{3}+\alpha^{21}\beta^{5}+\alpha^{40}\beta^{7}+\alpha^{46}\beta^{9}+\alpha^{50}\beta^{11}+\alpha^{33}\beta^{13}+\alpha^{75}\beta^{19}+\alpha^{110}\beta^{21}+1$
107	$\alpha^{102}\beta+\alpha^{53}\beta^{3}+\alpha^{17}\beta^{5}+\alpha^{9}\beta^{7}+\alpha^{115}\beta^{9}+\alpha^{92}\beta^{11}+\alpha^{48}\beta^{13}+\alpha^{9}\beta^{19}+\alpha^{17}\beta^{21}+1$
111	$\alpha^{91}\beta+\alpha^{20}\beta^{3}+\alpha^{89}\beta^{5}+\alpha^{59}\beta^{7}+\alpha^{16}\beta^{9}+\alpha^{98}\beta^{11}+\alpha^{32}\beta^{13}+\alpha^{54}\beta^{19}+\alpha^{40}\beta^{21}+1$
116	$\alpha^{69}\beta+\alpha^{81}\beta^{3}+\alpha^{106}\beta^{5}+\alpha^{32}\beta^{7}+\alpha^{72}\beta^{9}+\alpha^{110}\beta^{11}+\beta^{13}+\alpha^{17}\beta^{19}+\alpha^{86}\beta^{21}+1$
117	$\alpha^{46}\beta+\alpha^{12}\beta^{3}+\alpha^{118}\beta^{5}+\alpha^{125}\beta^{7}+\alpha^{119}\beta^{9}+\alpha^{111}\beta^{11}+\alpha^{82}\beta^{13}+\alpha^{88}\beta^{19}+\alpha^{111}\beta^{21}+1$
120	$\alpha^{24}\beta+\alpha^{73}\beta^{3}+\alpha^{8}\beta^{5}+\alpha^{98}\beta^{7}+\alpha^{48}\beta^{9}+\alpha^{123}\beta^{11}+\alpha^{50}\beta^{13}+\alpha^{51}\beta^{19}+\alpha^{30}\beta^{21}+1$
123	$\alpha^{36}\beta+\alpha^{109}\beta^{3}+\alpha^{68}\beta^{5}+\alpha^{55}\beta^{7}+\alpha^{29}\beta^{9}+\alpha\beta^{11}+\alpha^{79}\beta^{13}+\alpha^{25}\beta^{19}+\alpha^{28}\beta^{21}+1$
125	$\alpha^{64}\beta+\alpha^{66}\beta^{3}+\alpha^{81}\beta^{5}+\alpha^{124}\beta^{7}+\alpha^{27}\beta^{9}+\alpha^{55}\beta^{11}+\alpha^{62}\beta^{13}+\alpha^{49}\beta^{19}+\alpha^{108}\beta^{21}+1$

For rest of the key bits (154) the recovery method mentioned in [2] will be used. The same will have a negligible impact in terms of decreasing the requirement of keybits. The complexity calculations are appended in the section below: The keybits requirement is $2^{25.10}$ for online phase. The preprocessing phase requires keybits $2^{39.15}$ and storage on the order of $2^{25.10}$ with complexity on the order of $2^{25.10}$.

4.4.3.3 Complexity of the attack

The comparisons of complexity of the attack in [9] and Combination of DFT attack and Fast Selective DFT Attack is appended in the Table 4.3. It is evident that this combinational attack is costly than attack complexity presented by Yiyuan et al. [9].

Table 4.3 Complexities Comparisons of Fast Selective DFT Attack with Earlier Version

Complexity	DFT Attack[9]	Combination of Helleseth New attack and Fast Selective DFT Attack
Pre-Computation(Offline)	O($2^{39.5}$)	O($2^{39.15}$)
Data	$2^{25.5}$	$2^{25.10}$
Storage	$2^{39.51}$	$2^{39.16}$
Computational(Online)	O($2^{29.5}$)	O($2^{29.10}$)

4.5 Modification in Application of Fast Selective DFT Attack on WG-7

The application of Fast Selective DFT attack [6] on WG-7 cipher doesn't achieve considerable decrease in complexity of the attack due to structure of the cipher (WG-7). As the structure of the word oriented cipher hinders in computation of the DFT for the complete period (2^{161} -1) and absence of the polynomial which completely defines the period.

To address the issue of significant decrease in complexity of the DFT attack, a primitive polynomial will be derived to define the complete period of the WG-7 cipher before application of filter function. The polynomial derived as a result will be used to calculate DFT form of the annihilator and original function for subsequent use in two methods. Method-1 will use annihilator function to recover the initial keybits of the cipher. Method-2 will use original filter function for application of the fast selective DFT attack. To further illustrate this procedure, a simulated cipher similar in structure to WG-7 has been used. The methods are explained in detail in subsequent sections:-

4.5.1 Fast Selective DFT Attack using Annihilator

Two phases of the attack will be utilized to perform this method, offline and online. The relevant details of these methods are as following:

4.5.1.1 Precomputation (Offline) Phase

In precomputations phase, polynomial form of the WG-7 cipher without filter generator will be computed. For this purpose, two outputs of the cipher will be taken simultaneously cipher one without filter function and second with annihilator from equation 4.3. The cipher will be run with initial state set as (0, 0,…..,α^{128}). The bits for complete period (2^{161}) of the cipher will be generated. The bits generated without filter function (equation 4.1) will be XORed to calculate the primitive polynomial of the cipher using known BM algorithm. The polynomial will be primitive in GF (2^n) where n= 161.

The primitive polynomial will then be used to compute DFT coefficients (Trace form) of the annihilator by using the fast algorithm of computing DFT (FDFT) [7]. The output of the cipher with annihilator will be utilized for the purpose. Cyclotomic cosets leaders for the GF(2^{161}) will be identified to represent the trace form of the annihilator. The trace form will then be used in online phase for launching fast selective DFT attack as described by Gong et. al.[6] and in section 3.3.2. The requirement of storage will increase on the O(2^{162}). The major computation of the attack will be carried out in this phase and it is a onetime effort.

4.5.1.2 Online Computation Phase

Significant decrease in linear span of the cipher will be achieved by the introduction of the annihilator which also reduces requirement of linear independent equations. The system of equation for the captured bits will be generated by the method as mentioned in section 3.3.2.3 to recover the initial state of the filter generator (WG-7). The system of equation will be solved as already discussed in section 3.3.2.2 for recovering initial state of the cipher. Complexity of the attack is mentioned in the Table 4.5.

4.5.1.3 Application of DFT Attack on Simulated Filter Generator

A filter generator has been introduced to simulate the proposed method in section 4.5.1. The relevant details are given subsequently:-

4.5.1.3.1 Specification of Filter Generator

Filter generator is designed to generate up to 2^{10}-1 bits of key stream from a 10 bit key length. The internal state of 10 bits $[s_1, \dots, s_{10}]$ is divided into one LFSRs of length 2, containing 5 bit word each. The cipher consists of a 2 stage LFSR over F_{2^5} and a linear transformation. The finite field is defined by primitive polynomial $g(x) = x^5 + x^3 + 1$. The characteristic polynomial is primitive over F_{2^5} and is given by f(x) = $x^2 + x + \alpha^7$, where α is root of g(x). The non-linear transformation is defined by equation 4.20:-

$$h = x_0 + x_0x_1 + x_0x_2 + x_0x_1x_2 + x_3 + x_1x_3 + x_2x_3 + x_1x_2x_3 + x_4 + x_0x_4 + x_1x_4 + x_0x_1x_4 + x_2x_4 + x_0x_2x_4 + x_1x_2x_4 + x_0x_3x_4 \quad (4.20)$$

4.5.1.3.2 Offline Mode Computation

In offline mode, two outputs of the word oriented LFSR is taken. The period of the cipher is 2^{10}-1=1023, where n=mk=10, m=5 and k=2. One output is without its output filter function for the complete period. The other is with the use of annihilator mentioned as equation 4.21. Both the outputs are stored with a storage requirement of 2^{11}.

$$k = x_2 + x_0x_2 + x_1x_2 + x_0x_3 + x_1x_3 + x_1x_4 + x_3x_4 \qquad (4.21)$$

The output of the LFSR word is XORed in order to convert it from F_2^5 to F_2. If all the bits can be represented as $[x_0, \dots, x_9]$, then XORed function is performed on the bits as $x_0 + x_1 + x_2 + x_3 + x_4$. Then it is used as input to BM algorithm to generate the primitive polynomial as equation 4.22:

$$l(x) = x^{10} + x^9 + x^8 + x^4 + x^3 + x^2 + 1 \qquad (4.22)$$

After this step, the DFT form or trace form for the annihilator is generated as per algorithm specified in Section 4.4.3.1. The same can be re-verified by using the output of the cipher with annihilator and calculating DFT for their cyclotomic cosets leaders:

[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 69, 71, 73, 75, 77, 79, 83, 85, 87, 89, 91, 93, 95, 99, 101, 103, 105, 107, 109, 111, 115, 117, 119, 121, 123, 125, 127, 147, 149, 151, 155, 157, 159, 165, 167, 171, 173, 175, 179, 181, 183, 187, 189, 191, 205, 207, 213, 215, 219, 221, 223, 231, 235, 237, 239, 245, 247, 251, 253, 255, 341, 343, 347, 351, 363, 367, 375, 379, 383, 439, 447, 479, 495, 511]

The final trace form of the annihilator is mentioned as equation 4.23:

$$\mathrm{K(x)} = \mathrm{Tr}(\alpha^{66}x + \alpha^{858}x^3 + \alpha^{693}x^5 + \alpha^{429}x^9 + \alpha^{429}x^{17}), x \in F_{2^{10}} \qquad (4.23)$$

4.5.1.3.3 Online Mode Computation

In online computation, the equation 4.23 will be used to recover the initial state or bits of the cipher. The bits captured by the adversary are now used to perform fast selective DFT attack. The initial state of the LFSR can be represented as:

$w_0 = Tr(\beta)$, $w_1 = Tr(\alpha\beta)$,$w_2 = Tr(\alpha^2\beta)$,$w_3 = Tr(\alpha^3\beta)$,$w_4 = Tr(\alpha^4\beta)$,$w_5 = Tr(\alpha^5\beta)$, $w_6 = Tr(\alpha^6\beta)$,$w_7 = Tr(\alpha^7\beta)$,$w_8 = Tr(\alpha^8\beta)$,$w_9 = Tr(\alpha^9\beta)$ (4.24)

To find out the initial state, β needs to be computed and captured keystream bits will be utilized for the purpose. All bits index 'i' where bits value is 1 will be used to form system of equation on $F_2{}^{10}$. The equation 4.23 will be substituted for $x = \alpha^i\beta$ to form equation as

$$Tr\left(\alpha^{66}\alpha^i\beta + \alpha^{858}\alpha^{3i}\beta^3 + \alpha^{693}\alpha^{5i}\beta^5 + \alpha^{429}\alpha^{9i}\beta^9 + \alpha^{429}\alpha^{17i}\beta^{17}\right) = 0 \quad (4.25)$$

Then all corresponding β^n is replaced as

$$\beta^n = x_0 + \alpha x_1 + \alpha^2 x_2 + \alpha^3 x_3 + \alpha^4 x_4 + \alpha^5 x_5 + \alpha^6 x_6 + \alpha^7 x_7 + \alpha^8 x_8 + \alpha^9 x_9 \quad (4.26)$$

System of equation is generated using equation 4.26 as:

$$Tr(\alpha^{66}\alpha^i(x_0 + \alpha x_1 + \alpha^2 x_2 + \alpha^3 x_3 + \alpha^4 x_4 + \alpha^5 x_5 + \alpha^6 x_6 + \alpha^7 x_7 + \alpha^8 x_8 + \alpha^9 x_9)) + Tr(\alpha^{858}\alpha^{3i}(x_0 + \alpha x_1 + \alpha^2 x_2 + \alpha^3 x_3 + \alpha^4 x_4 + \alpha^5 x_5 + \alpha^6 x_6 + \alpha^7 x_7 + \alpha^8 x_8 + \alpha^9 x_9)) + Tr(\alpha^{693}\alpha^{5i}(x_0 + \alpha x_1 + \alpha^2 x_2 + \alpha^3 x_3 + \alpha^4 x_4 + \alpha^5 x_5 + \alpha^6 x_6 + \alpha^7 x_7 + \alpha^8 x_8 + \alpha^9 x_9)) + Tr(\alpha^{429}\alpha^{9i}(x_0 + \alpha x_1 + \alpha^2 x_2 + \alpha^3 x_3 + \alpha^4 x_4 + \alpha^5 x_5 + \alpha^6 x_6 + \alpha^7 x_7 + \alpha^8 x_8 + \alpha^9 x_9)) + Tr\left(\alpha^{429}\alpha^{17i}(x_0 + \alpha x_1 + \alpha^2 x_2 + \alpha^3 x_3 + \alpha^4 x_4 + \alpha^5 x_5 + \alpha^6 x_6 + \alpha^7 x_7 + \alpha^8 x_8 + \alpha^9 x_9)\right) = 0 \quad (4.27)$$

From the linearity of the trace function, system of equation is solved to extract values of {x_0,..x_9}. Which is finally substituted to get value of β to recover the initial state. The complexity of attack is mentioned in Table 4.4.

4.5.2 Fast Selective DFT Attack with Original Function

Two phases of the attack will be utilized to perform this method, offline and online. The relevant details of these methods are as following:

4.5.2.1 Precomputations (Offline) Phase

In precomputations phase, polynomial form of the WG-7 cipher without filter generator will be computed. For this purpose, two outputs of the cipher will be taken simultaneously cipher one without filter function and other with filter function from equation 4.2. The cipher will be run with initial state set as (0, 0,…..,α^{128}). The bits for complete period (2^{161}-1) of the cipher will be generated. The bits generated without filter function will be XORed to calculate the primitive polynomial of the cipher using known BM algorithm. The polynomial will be primitive in GF (2^n) where n= 161.
The primitive polynomial will then be used to compute DFT coefficients (Trace form) of the annihilator by using the fast algorithm of computing DFT (FDFT) [7]. The output of the cipher with original function will be utilized for the purpose. Cyclotomic cosets leaders for the F_2^{161} will be identified to represent the trace form of the original function. The trace form will then be used in online phase for launching fast selective DFT attack as described by Gong et. al.[6] and section 3.3.2. The requirement of storage will increase 2^{162}. The major computation of the attack will be carried out in this phase and it is a onetime effort.

4.5.2.2 Online Computation Phase

The system of equation for the captured bits will be generated by the method as mentioned in section 3.3.2.2 to recover the initial state of the filter generator (WG-7). The system of equation will be solved as already discussed in section 3.3.2.2 for recovering initial state of the cipher. The complexity of the attack is mentioned in Table 4.5.

4.5.2.3 Application of DFT Attack on Simulated Filter Generator

The filter generator introduced in the section 4.5.1.3 has been used to simulate the proposed method in section 4.5.2. The trace representation of $\{h_i\}$ is:

$$H(x) = Tr(\alpha^{462}x + \alpha^{858}x^3 + \alpha^{330}x^5 + \alpha^{363}x^7 + \alpha^{594}x^9 + \alpha^{528}x^{11} + \alpha^{66}x^{13} + \alpha^{429}x^{17} + \alpha^{693}x^{19} + \alpha^{264}x^{21} + \alpha^{858}x^{25} + \alpha^{363}x^{69} + \alpha^{528}x^{73}) \qquad (4.28)$$

The method will remain the same as mentioned in 4.5.1.3 with exception of using equation 4.28 instead of equation 4.23 to recover the initial state. The complexity of attack is mentioned in Table 4.4.

Table 4.4 Comparisons of Key Recovery Attacks on Simulated Filter Generator

Complexity	**DFT Method**	**Algebraic Attack**	**Fast Selective DFT method-1**	**Fast Selective DFT method-2**
Pre-Computation(Offline)	O($2^{16.15}$)	-	O($2^{16.8}$)	O($2^{18.6}$)
Data	$2^{7.48}$	$2^{5.79}$	2^{5}	$2^{7.2}$
Computational(Online)	O($2^{11.25}$)	O($2^{12.752}$)	O($2^{9.3}$)	O($2^{11.15}$)

4.6 Comparison of Key Recovery Attacks on WG-7

This table compares the various key recovery attack complexity results of predicted DFT attack by Yiyuan et. al.[9], Algebraic/FAA attack by Orumiehchiha et. al.[10] and proposed fast selective DFT attack with or without annihilator. It is evident by comparison of the complexities that proposed fast selective DFT attack with annihilator (method-1) is efficient in online phase and keybits requirement than Yiyuan et. al. [9] predicted DFT attack, however, precomputation phase and storage is costly than the former. The method-1 is also efficient in online phase and keybits requirement than Algebraic/FAA attack but costly in precomputation and storage. The DFT attack without annihilator (method-2) is of the same complexity as the Yiyuan et. al. [9] predicted DFT attack except the precomputation phase which is costly for the method-2.

Table 4.5 Comparisons of Key Recovery Attacks on WG-7

Complexity	**DFT Method[9]**	**Algebraic Attack [10]**	**Fast Algebraic attack[10]**	**Fast Selective DFT method-1**	**Fast Selective DFT method-2**
Pre-Computation(Offline)	O($2^{39.5}$)	-	$2^{26.87}$	O(2^{41})	O(2^{45})
Data	$2^{25.5}$	$2^{19.38}$	$2^{19.38}$	2^{18}	$2^{25.15}$
Computational(Online)	O($2^{29.5}$)	$2^{54.36}$	$2^{26.73}$	O($2^{23.5}$)	O($2^{29.75}$)

4.7 Summary

In this chapter, new results of Discrete Fourier Transform attack against the stream cipher WG-7 have been presented which is an improvement over the previous attack. First a method has been presented which recovers few bits of the initial value. After this method two more methods have been presented which display significant improvement over the previous methods.

Chapter 5

Discrete Fourier Transform Attack on Clock Controlled and Block Cipher

5.1 Introduction

In this chapter, DFT attacks are launched against clock controlled and block cipher to check its efficacy. This evaluation has never been carried out since inception of the attack; however, a brief method in literature is available in [7]. For clock controlled cipher evaluation is carried out against Alternating Step Generator and Shrinking Generator. For block cipher evaluation Toy cipher is used. The main reason to employ the small structures for the development of the attack is to first check the applicability of the attack with encouraging results and later develop the attack on similar practical structure of stream/block cipher.

Section 5.2 describes the construction of Alternating step generator and the evaluation of the generator against DFT attack. Section 5.3 describes the construction of Shrinking generator and the evaluation of the generator against DFT attack and Section 5.4 describes the construction of CTC and the evaluation of the cipher against DFT attack.

5.2 DFT Attack against the Alternating Step Generator

Alternating Step Generator, a stream cipher introduced in [13]. It is shown in Figure 5.1.

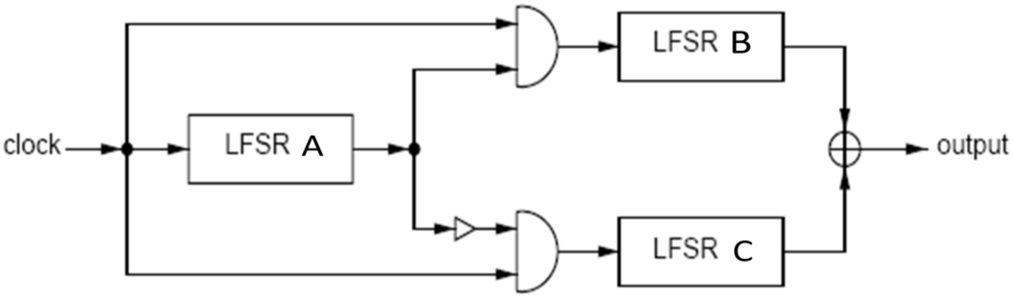


Figure 5.1 Alternating Step Generator

5.2.1 Specifications

To simulate attack on the alternating Step Generator a three LFSR of different lengths relatively prime to each other are used. The parameters are summarized in Table 5.1.

Table 5.1 Parameters of Alternating Step Generator

Parameters	**LFSR A**	**LFSR B**	**LFSR C**
Lengths	2	3	5
Polynomial	$x^2 + x + 1$	$x^3 + x + 1$	$x^5 + x^3 + 1$

5.2.2 Keystream Generation

LFSR A is controlling the clocking of the other two LFSR. When the output of LFSR A is one it triggers LFSR B and when the output of LFSR A is zero it triggers the LFSR C. The output is taken as XOR of the LFSR B and C registers. The system output can be represented by the equation:

$$z_t = B_t^m + C_t^n \tag{5.1}$$

Where m is the length of LFSR B and n is the length of LFSR C. The period of the Alternating step generator is defined as

$$pd = (2^l - 1)(2^m - 1)(2^n - 1) \tag{5.2}$$

Which comes out to be 651 for l=2, m=3 and n=5. The system of equation to relate all the LFSR's as defined in [11] are:

$$z^t \oplus z^{t+1} = (B_m^t \oplus B_{m-1}^t)A_l^t + (C_n^t \oplus C_{n-1}^t)(A_l^t \oplus 1) \quad (5.3)$$

Multiplying both sides of the equation 5.3 with $(A_l^t + 1)$ gives

$$(z^t \oplus z^{t+1})(A_l^t \oplus 1) = (C_n^t \oplus C_{n-1}^t)(A_l^t \oplus 1) \quad (5.4)$$

Multiplying both sides of the equation 5.3 with A_l^t gives

$$(z^t \oplus z^{t+1})A_l^t = (B_m^t \oplus B_{m-1}^t)A_l^t \quad (5.5)$$

5.2.3 Application of Fast Selective DFT Attack

Fast Selective DFT Attack has been performed on Alternate Step Generator using the method by Gong et. al. in [6]. The attack is launched in Galois extension field to carryout computations of the multivariate variables generated from the equations given by Al-Hinai et al. in [11]. The implementation is carried out on maple version 15. The maple code to automate the process is mentioned in CD attached with the thesis. Each LFSR is represented by its trace representation and substituting the same trace form in Equation 5.5 gives following equation:

$$Tr_1^2(x)(Tr_1^3(x + \alpha x) \oplus 1) = 0 \quad (5.6)$$

Equation 5.6 is applicable where output bits are all one for $z^t \oplus z^{t+1} = 1$. Now x is replaced with $\alpha^v\beta$ for $Tr_1^2(x)$, where v is the index position of t when $z^t \oplus z^{t+1} = 1$. Then β is replaced with $x_0 + \alpha x_1$. Similarly, x is be replaced with $\alpha^v\gamma$ for $Tr_1^3(x)$, where v is the index position of t when $z^t \oplus z^{t+1} = 1$. Then γ is replaced with $y_0 + \alpha y_1 + \alpha^2 y_2$. The trace form is transformed to find the elements of subfield F_2 from F_2^n.The equations mentioned in Table 5.2 have been generated for 128 captured bits and evaluated to find the initial state:

Table 5.2 Equation for Alternating Step Generator

Indexes	**Equation**	**Indexes**	**Equation**
1	$x_0+y_1x_0+y_2x_0+x_1+y_1x_1+y_2x_1$	54	$x_1+y_2x_1$
2	$x_0+y_0x_0+y_1x_0+y_2x_0$	55	$x_0+y_1x_0+x_1+y_1x_1$
3	$x_1+y_0x_1+y_1x_1$	56	$x_0+y_0x_0+y_2x_0$
5	$x_0+y_2x_0$	60	$x_1+y_0x_1$
7	$x_0+y_0x_0+y_2x_0+x_1+y_0x_1+y_2x_1$	63	$x_1+y_0x_1+y_2x_1$
10	$x_0+y_0x_0+y_1x_0+x_1+y_0x_1+y_1x_1$	64	$x_0+y_2x_0+y_1x_0+x_1+y_2x_1+y_1x_1$
11	$x_0+y_0x_0$	65	$x_0+y_2x_0+y_1x_0+y_0x_0$
12	$x_1+y_2x_1$	66	$x_1+y_1x_1+y_0x_1$
13	$x_0+y_1x_0+x_1+y_1x_1$	68	$x_0+y_2x_0$
17	$x_0+y_0x_0+y_1x_0$	70	$x_0+y_2x_0+y_0x_0+x_1+y_2x_1+y_0x_1$
18	$x_1+y_0x_1$	73	$x_0+y_1x_0+y_0x_0+x_1+y_1x_1+y_0x_1$
20	$x_0+y_1x_0$	74	$x_0+y_0x_0$
21	$x_1+y_0x_1+y_2x_1$	75	$x_1+y_2x_1$
22	$x_0+y_1x_0+y_2x_0+x_1+y_1x_1+y_2x_1$	76	$x_0+y_1x_0+x_1+y_1x_1$
23	$x_0+y_0x_0+y_1x_0+y_2x_0$	81	$x_1+y_0x_1$
24	$x_1+y_0x_1+y_1x_1$	83	$x_0+y_1x_0$
28	$x_0+y_0x_0+y_2x_0+x_1+y_0x_1+y_2x_1$	84	$x_1+y_2x_1+y_0x_1$
29	$x_0+y_1x_0+y_2x_0$	85	$x_0+y_2x_0+y_1x_0+x_1+y_2x_1+y_1x_1$
31	$x_0+y_0x_0+y_1x_0+x_1+y_0x_1+y_1x_1$	87	$x_1+y_1x_1+y_0x_1$
32	$x_0+y_0x_0$	91	$x_0+y_2x_0+y_0x_0+x_1+y_2x_1+y_0x_1$
33	$x_1+y_2x_1$	94	$x_0+y_1x_0+y_0x_0+x_1+y_1x_1+y_0x_1$
34	$x_0+y_1x_0+x_1+y_1x_1$	96	$x_1+y_2x_1$
39	$x_1+y_0x_1$	97	$x_0+y_1x_0+x_1+y_1x_1$
42	$x_1+y_0x_1+y_2x_1$	98	$x_0+y_2x_0+y_0x_0$
43	$x_0+y_1x_0+y_2x_0+x_1+y_1x_1+y_2x_1$	102	$x_1+y_0x_1$
44	$x_0+y_0x_0+y_1x_0+y_2x_0$	104	$x_0+y_1x_0$
45	$x_1+y_0x_1+y_1x_1$	105	$x_1+y_2x_1+y_0x_1$

47	$x_0+y_2x_0$	106	$x_0+y_2x_0+y_1x_0+x_1+y_2x_1+y_1x_1$
49	$x_0+y_0x_0+y_2x_0+x_1+y_0x_1+y_2x_1$	108	$x_1+y_1x_1+y_0x_1$
50	$x_0+y_1x_0+y_2x_0$	110	$x_0+y_2x_0$
52	$x_0+y_0x_0+y_1x_0+x_1+y_0x_1+y_1x_1$	112	$x_0+y_2x_0+y_0x_0+x_1+y_2x_1+y_0x_1$
53	$x_0+y_0x_0$	113	$x_0+y_2x_0+y_1x_0$

The multivariate equations developed as result of Fast selective DFT attack is solved by guessing values of $x_0\, and\, x_1$ to find the value of $y_0\,,\ y_1\, and\, y_2$. These values have been used to extract the shift β and γ to recover the initial state of the Alternating Step Generator. There are total four guess values due to polynomial x^2+x+1 being used to generate the states of LFSR.

The results are not so much encouraging for the further development of attack because no unique solution is possible of the equation system. Guess work is required which adds complexity to extract the shift β and γ to recover the initial state of the Alternating Step Generator. The cost and complexity is more than the known generic methods like algebraic attack and correlation attack described in [11]. The difficulty being experienced is to separate linear and non-linear part of the cipher to accomplish successful results in application of DFT attack.

5.2.4 Application of Selective DFT Attack

Selective DFT Attack has been performed on Alternate Step Generator using the method by Gong in [7] and Section 3.3.1. The implementation carried out on maple version 15. The maple code to automate the process is mentioned in CD attached with the thesis. The minimal polynomial is formed as following with a linear span of 49:

$$g(x) = x^{49} + x^{42} + x^{21} + x^{14} + 1 \tag{5.7}$$

However, the issue which arises is to find the root of the Equation 5.7 and has been checked on all values relatively prime to period of Alternative Step Generator. No value which is relatively prime to 651 is found as root or zero of equation 5.7 which renders this attack useless against clock controlled ciphers of this pattern. The further development of Selective DFT attack is stopped due to this impediment.

5.3 DFT Attack against the Shrinking Generator

Shrinking Generator, a stream cipher introduced in [15]. It is shown in Figure 5.2.

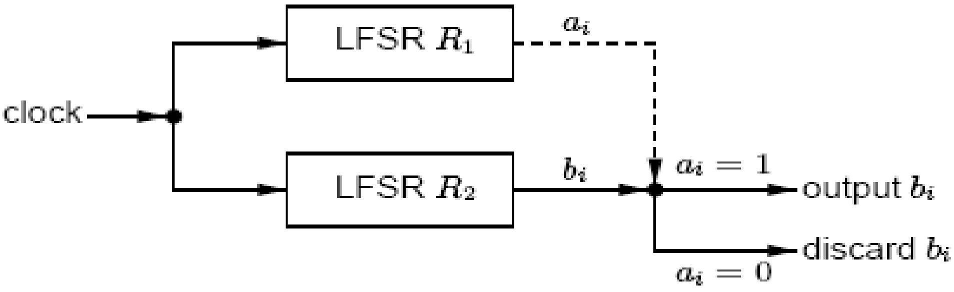


Figure 5.2 Shrinking Generator

5.3.1 Specifications

The Shrinking Generator used here has two LFSR of lengths relatively prime to each other. The parameters are summarized in Table 5.3.

Table 5.3 Parameters of Shrinking Generator

Parameters	**LFSR R1**	**LFSR R2**
Lengths	2	3
Polynomial	$x^2 + x + 1$	$x^3 + x + 1$

5.3.2 Keystream Generation

LFSR R1 is controlling the output of a second LFSR R2. Registers R1 and R2 are clocked. If the output of R1 is 1, the output bit of R2 forms part of the keystream. If the output of R1 is 0, the output bit of R2 is discarded. Where m is the length of LFSR R_1 and n is the length of LFSR R_2. The period of the Shrinking generator is defined as

$$pd = 2(2^n - 1) \tag{5.8}$$

Which comes out to be 14 for m=2 and n=3.

5.3.3 Application of Selective DFT Attack

Selective DFT Attack has been performed on Shrinking Generator using the method by Gong in [7] and Section 3.3.1. The implementation carried out on maple version 15. The maple code to automate the process is mentioned in CD attached with the thesis. The minimal polynomial of **s** is formed with a linear span of 6:

$$g(x) = x^6 + x^4 + 1 \tag{5.9}$$

However, the issue which arises is to find the root of the Equation 5.9 and has been checked on all values relatively prime to period of Shrinking Generator. No value which is relatively prime to 14 has been found as root or zero of Equation 5.9 which renders this attack useless against clock controlled ciphers of this pattern. The further development of Selective DFT attack is stopped due to this impediment.

5.3.4 Application of Helleseth et al. New Attack

Helleseth et al. new attack [1] has been performed on Shrinking Generator to check its efficacy after failure of selective DFT attack. The implementation carried out on maple version 15. The maple code to automate the process is mentioned in CD attached with the thesis. The minimal polynomial is taken from Equation 5.9.

However, the problem posed by the equation 5.9 that it generates all the monomials of degree one as output of LFSR R$_2$. Equation 5.9 eliminates all degree one relations and hence renders this attack useless against clock controlled ciphers of this pattern. The further development of the attack is stopped due to this impediment. In other words it can be said that the difficulty of launching Helleseth et al. New Attack [1] against Shrinking Generator is the non availability of polynomial for generating monomials of degree 2 and higher.

5.3.5 Application of Fast Selective DFT Attack

Fast Selective DFT Attack has been performed on Shrinking Generator using the method by Gong et. al. in [6] and Section 3.3.2. The attack is launched where the Galois extension field has been employed to carryout computations of the multivariate variables. The variables have been generated from the following the system of equation to relate all the LFSR’s:

$$z^t = (B_m^t)A_l^t \tag{5.10}$$

The implementation carried out on maple version 15. The maple code to automate the process is mentioned in CD attached with the thesis. Each LFSR is represented by its trace representation and substituting the same trace form in Equation 5.10 gives Equation 5.11:

$$Tr_1^2(x)Tr_1^3(x) = z^t \tag{5.11}$$

Equation 5.11 will be applicable where $Tr_1^2(x)$ is equal to one. Now x will be replaced with $\alpha^v\beta$ for $Tr(x)$, where v is the index position of t when $Tr_1^2(x) = 1$. Then β will be replaced with $x_0 + \alpha x_1$. Similarly, x will be replaced with $\alpha^v\gamma$ for $Tr_1^3(x)$, where v is the index position of t when $Tr_1^2(x) = 1$. Then γ will be replaced with $y_0 + \alpha y_1 + \alpha^2 y_2$. The trace form is transformed to find the elements of subfield

F_2 from F_2^n.The equations mentioned in Table 5.4 have been generated for 7 captured bits and evaluated to find the initial state:

Table 5.4 Equation for Shrinking Generator

Indexes	Equation
1	x_1y_0=0
3	x_0y_1=1
4	$x_1(y_0+y_2)$=1
6	$x_0(y_0+y_1+y_2)$=0
7	$x_1(y_0+y_1)$=1
9	x_0y_2=1
10	x_1y_1=1

The multivariate equations developed as result of Fast selective DFT attack is solved by guessing values of x_0 *and* x_1 to find the value of y_0, y_1 *and* y_2. These values are then used to extract the shift γ to recover the initial state of the Shrinking Generator. There are total three guess values due to polynomial x^2+x+1 being used to generate the states of LFSR.

The results are not so much encouraging for the further development of attack because of more complexity than the known generic methods like algebraic attacks due to guess work involved. The difficulty being experienced is to separate linear and non-linear part of the cipher to accomplish successful results in application of DFT attack. Also the base polynomial selection for calculation of DFT of the sequence is difficult due to presence of 2 x LFSRs primitive polynomials.

5.4 DFT Attack against the Toy Block Cipher

The toy block cipher considered for the purpose of evaluation of DFT attack is a basic Substitution-Permutation Network (SPN). A simplified structure is being evaluated to get a firsthand knowledge on the applicability of the DFT Attack. If the applicability of DFT attack is proved on Toy cipher then practical cipher can be considered for finding vulnerability against this attack.

5.4.1 TC: A Toy Block Cipher

The Toy Cipher takes a 16-bit input block and the block is processed by repeating the basic operations of round four times. Each round consists of substitution, a transposition of the bits (permutation of the bit positions), and key mixing. The basic structure is presented by Feistel in 1973[14] and these basic properties also found in AES and DES as well. It is shown in Figure 5.3. The maple code to automate the process is mentioned in CD attached with the thesis.

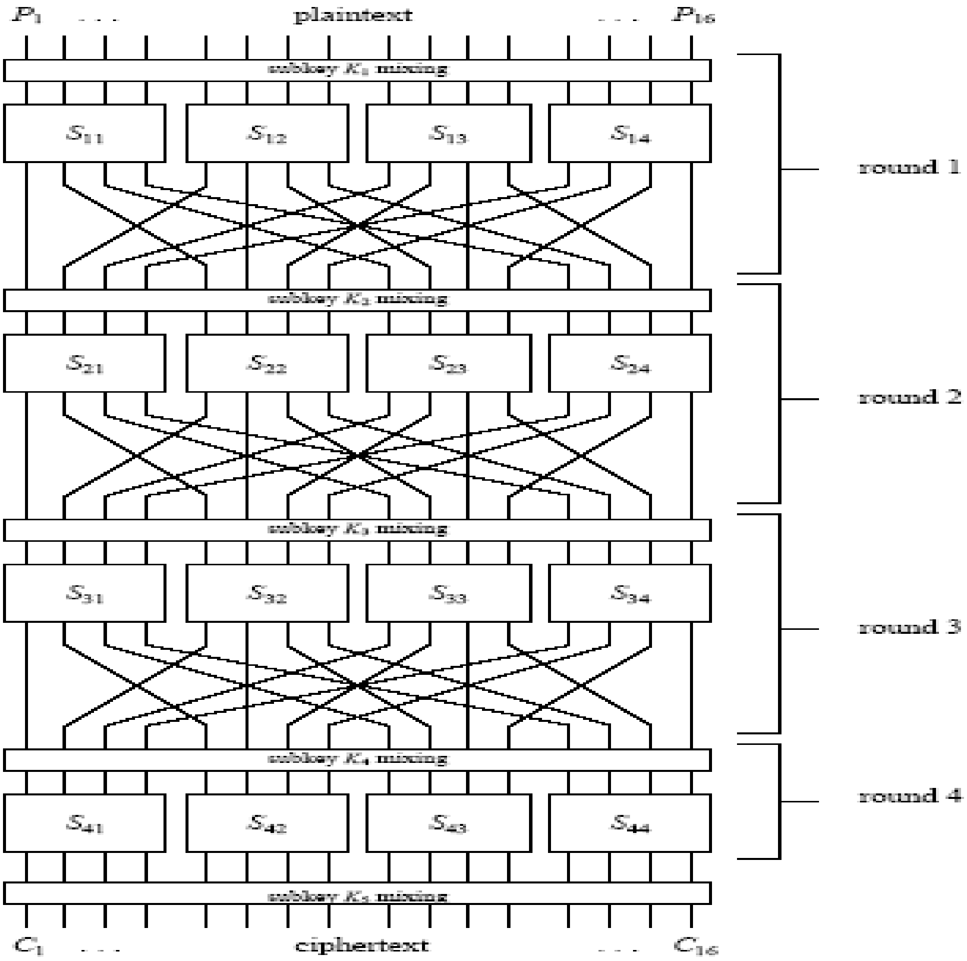


Figure 5.3 Substitution-Permutation Network

5.4.2 Specifications

The Toy Cipher has a Feistel structure having a block size of 16-bit, subkey length of 16-bit and 4 rounds. The parameters are summarized in Table 5.5.

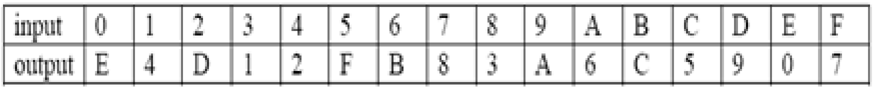
Table 5.5 Parameters of Toy Cipher

Block Size	**Key Length**	**Number of Rounds**
16-bit	64-bit	4

5.4.3 S-Box Representation

In this cipher, the 16-bit data block is split into four 4-bit sub-blocks. Each sub-block forms an input to a 4×4 S-box (a substitution with 4 input and 4 output bits), which can be easily implemented with a table lookup of sixteen 4-bit values, indexed by the integer represented by the 4 input bits as shown in Table 5.6. The most fundamental property of an S-box is that it is a nonlinear mapping, i.e., the output bits cannot be represented as a linear operation on the input bits.

Table 5.6 S-box Representation

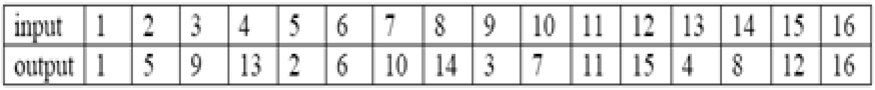


input	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
output	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7

5.4.4 P-Box Representation

Each S-box output is combined to make sixteen input bits. The these bits are permuted to get sixteen output bits, which can be easily implemented with a table lookup of sixteen bit values, indexed by the position represented by each input bit as shown in Table 5.7. In round 4 there is no permutation after substitution.

Table 5.7 P-box Representation



input	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
output	1	5	9	13	2	6	10	14	3	7	11	15	4	8	12	16

5.4.4.1 Round Function *F*

Figure 5.3 illustrates the round function F in detail consisting of an XOR operation followed by a Substitution and Permutation Network. The function F is defined as under:

$$: \quad (X, K_i) \quad \rightarrow \quad C = P(S(X \oplus K_i))$$

The last round doesnot has permutation after substitution. The output of S-Box is taken as Cipher text.

5.4.5 Key Scheduling

Key scheduling for the purpose of evaluation of DFT is considered to be an independent key generation for creation of each round subkey. Each round subkey distribution is shown below:

$$K \leftarrow [k_0, \dots, k_{63}]$$

$$K_1 \leftarrow [k_0, \dots, k_{15}]$$

$$K_2 \leftarrow [k_{16}, \dots, k_{31}]$$

$$K_3 \leftarrow [k_{32}, \dots, k_{47}]$$

$$K_4 \leftarrow [k_{48}, \dots, k_{63}]$$

5.4.6 Decryption Algorithm

Decryption algorithm is simply the inverse of encryption algorithm and subkeys used in the reverse order. There is no need to further explain the decryption algorithm here as it is not used in the DFT attack [7].

5.4.7 Application of DFT on Toy Cipher

The attack described in [7] is applied on the toy cipher for recovery of Key K. This method mainly revolves around the assumption that plaintext is formed in such a way that it can be regarded as output of LFSR. The attacker has the liberty to generate as many ciphertexts as he wants using the same key K for initial captured ciphertext. Attacker has obtained two ciphertexts and obtained plaintext corresponding to one of the ciphertext. The key remains same for both sets of ciphertexts. The plaintext are generated by LFSRs and padded by zero to complete the required input bits. The plaintexts/ciphertexts are shift equivalent of each other. The unknown plaintext will be found from two ciphertexts and one plaintext However, the unknown is the key K which will be recovered using captured and further generated ciphertexts in DFT attack. The method is repeated here for completeness:

All two sequences $\{a_t\}$ and $\{b_t\}$ are generated by LFSR. The LFSR sequence is selected such as its characteristic polynomial degree is less than length of m bits key K. The Encryption process $E_K(x)$ is regarded as filtering function for a filter generator. Using $E_K(x)$, $s_t=E_K(a_t,0)$, $a_t=(a_t,a_{t+1},\ldots,a_{t+n-1})$, $u_t=E_K(b_t,0)$, $b_t=(b_t,b_{t+1},\ldots,b_{t+n-1})$ are generated. Where 0 is the zero vector of dimensional m-n by padding m-n bits to both sequences $\{a_t\}$ and $\{b_t\}$. Then $u_t = s_{t+\tau}$, and $U_k = \alpha^{k\tau} S_k$, where $\{s_t\}$ and $\{u_t\}$ represent the corresponding component sequences in their respective output vector sequences. As $b_t = a_{t+\tau}$, the Key K will be recovered from the DFT of $\{S_k\}$ and $\{U_k\}$ as each component is a function of K.

5.4.8 Test Vectors

Test vectors for Toy Cipher in binary notation with the corresponding key are shown in Table 5.8.

Table 5.8 Test Vectors

PT	CT	Key-1
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0	0,0,1,1,0,1,1,0,0,0,0,0,1,1,1,0	K_1:1,1,1,1,1,1,0,0,0,0,0,1,0,0,0,0
		K_2:1,1,0,0,0,1,0,1,0,0,1,1,1,1,0,1
		K_3:0,0,0,1,1,1,0,0,1,0,0,1,0,1,1,0
		K_4:1,1,1,0,1,1,0,0,1,1,0,1,0,1,0,1
Cosets-Leaders	**DFT**	
0	1	
1	1	
3	$\alpha + \alpha^2$	
5	0	
7	$1+\alpha^3$	

5.5 Results of the DFT Attack

The selective DFT attack as mentioned in Section 3.3.1 and by Gong [7] is developed and implemented on the cipher. The maple code to automate the process is mentioned in CD attached with the thesis. The degree of selected LFSR is 4 and the polynomial selected as $x^4 + x + 1$. Value of chosen plaintext $\{a_t\}$ and $\{b_t\}$ with corresponding ciphertexts $\{s_t\}$ and $\{u_t\}$ respectively generated by the Encryption function $E_K(x)$ is shown in Table 5.9. DFT Spectra for S_k and U_k are shown in Table 5.10 and Table 5.11 respectively.

The results are not encouraging as it doesn’t retrieve the keybits from the cipher yet the method devised to retrieves the plaintext from the ciphertext if the plaintext becomes the shifted version of the LFSR output. The limitation of retrieving the plaintext is that it must be generated by a linear feedback shift register and arranged in the output of LFSR and rest padded by the zero bits. The result proved to be not of great interest as is formatted in an unrealistic way and lots of unrealistic assumptions required to carry out the DFT attack.

The suggested attack against block cipher is not effective due to recovery of plaintext bits on unrealistic assumption about the target plaintext format. The block cipher DFT attack doesn’t target the symmetric key which is the aim in stream cipher version of the DFT attack.

Table 5.9 LFSR Sequences for Toy Cipher

<table>
<tr><th>a_t</th><th>b_t</th><th>Key</th></tr>
<tr><td>0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0</td><td>0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0</td><td rowspan="19">K_1:1,1,1,1,1,1,0,0,0,0,0,1,0,0,0,0
K_2:1,1,0,0,0,1,0,1,0,0,1,1,1,1,0,1
K_3:0,0,0,1,1,1,0,0,1,0,0,1,0,1,1,0
K_4:1,1,1,0,1,1,0,0,1,1,0,1,0,1,0,1</td></tr>
<tr><td>0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0</td><td>1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0</td></tr>
<tr><td>0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0</td><td>1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0</td></tr>
<tr><td>1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0</td><td>1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0</td></tr>
<tr><td>0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0</td><td>1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0</td></tr>
<tr><td>0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0</td><td>0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0</td></tr>
<tr><td>1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0</td><td>0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0</td></tr>
<tr><td>1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0</td><td>0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0</td></tr>
<tr><td>0,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0</td><td>1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0</td></tr>
<tr><td>1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0</td><td>0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0</td></tr>
<tr><td>0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0</td><td>0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0</td></tr>
<tr><td>1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0</td><td>1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0</td></tr>
<tr><td>1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0</td><td>1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0</td></tr>
<tr><td>1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0</td><td>0,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0</td></tr>
<tr><td>1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0</td><td>1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0</td></tr>
<tr><td colspan="2">0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0</td></tr>
<tr><th>s_t</th><th>u_t</th></tr>
<tr><td>0,0,0,0,0,0,1,1,0,1,1,0,1,0,1,1
$1+x^{16}$</td><td>1,1,1,0,1,1,0,0,0,1,0,1,1,0,0,1
$1+x^{16}$</td></tr>
<tr><td>0,0,0,0,1,0,1,1,0,0,1,0,0,0,1,1</td><td>0,0,1,1,0,0,1,1,1,0,0,0,1,1,0,0</td></tr>
</table>

$x^{14}+x^{12}+x^{10}+x^8+x^6+x^4+x^2+1$	$1+x^{16}$	
0,1,1,0,0,1,0,0,1,0,0,0,1,1,1,0 $1+x^{16}$	0,0,0,0,0,0,0,1,0,1,0,0,1,0,1,1 $1+x^{16}$	
1,0,1,1,1,0,0,1,0,1,0,1,1,1,0,0 $1+x^{16}$	0,0,1,1,1,1,1,0,1,0,1,1,0,1,0,1 $x^{15}+x^{14}+x^{13}+x^{12}+x^{11}+x^{10}+x^9+x^8+$ x^7+ $x^6+ x^5+x^4+ x^3+x^2+x+1$	
1,1,1,0,0,1,0,0,1,0,0,1,1,0,1,1 $1+x^{16}$	0,0,1,1,0,1,1,0,0,0,0,0,1,1,1,0 $1+x^{16}$	
0,0,1,1,0,0,0,1,0,1,0,1,1,1,0,0 $1+x^{16}$	0,0,0,0,0,0,1,1,0,1,1,0,1,0,1,1 $1+x^{16}$	
0,0,1,0,1,0,0,0,0,1,1,1,1,1,0,1 $x^{10}+x^8+ x^2+1$	0,0,0,0,1,0,1,1,0,0,1,0,0,0,1,1 $x^{14}+x^{12}+x^{10}+x^8+x^6+x^4+x^2+1$	
0,0,1,0,0,1,0,1,1,1,1,1,0,1,0,0 $x^{15}+x^{14}+x^{13}+x^{12}+x^{11}+x^{10}+x^9+x^8$ $+$ $x^7+x^6+ x^5+x^4+ x^3+x^2+x+1$	0,1,1,0,0,1,0,0,1,0,0,0,1,1,1,0 $1+x^{16}$	
0,0,0,0,0,0,1,1,0,1,0,0,1,0,0,1 $1+x^{16}$	1,0,1,1,1,0,0,1,0,1,0,1,1,1,0,0 $1+x^{16}$	
0,0,0,0,0,0,0,1,0,1,1,0,1,0,0,1 $1+x^{16}$	1,1,1,0,0,1,0,0,1,0,0,1,1,0,1,1 $1+x^{16}$	
1,1,1,0,1,1,0,0,0,1,0,1,1,0,0,1 $1+x^{16}$	0,0,1,1,0,0,0,1,0,1,0,1,1,1,0,0 $1+x^{16}$	
0,0,1,1,0,0,1,1,1,0,0,0,1,1,0,0 $1+x^{16}$	0,0,1,0,1,0,0,0,0,1,1,1,1,1,0,1 $x^{10}+x^8+ x^2+1$	
0,0,0,0,0,0,0,1,0,1,0,0,1,0,1,1 $1+x^{16}$	0,0,1,0,0,1,0,1,1,1,1,1,0,1,0,0 $x^{15}+x^{14}+x^{13}+x^{12}+x^{11}+x^{10}+x^9+x^8+$ x^7+ $x^6+ x^5+x^4+ x^3+x^2+x+1$	
0,0,1,1,1,1,1,0,1,0,1,1,0,1,0,1	0,0,0,0,0,0,1,1,0,1,0,0,1,0,0,1	

$x^{15}+x^{14}+x^{13}+x^{12}+x^{11}+x^{10}+x^{9}+x^{8}$ + $x^{7}+x^{6}+x^{5}+x^{4}+x^{3}+x^{2}+x+1$	$1+x^{16}$	
0,0,1,1,0,1,1,0,0,0,0,0,1,1,1,0 $1+x^{16}$	0,0,0,0,0,0,0,1,0,1,1,0,1,0,0,1 $1+x^{16}$	
1,1,0,1,0,1,1,0,1,0,0,0,1,0,0,0 $1+x^{16}$		

Table 5.10 DFT Spectra S_k

s_t	**0**	**1**	**3**	**5**	**7**
0,0,0,0,0,0,1,1,0,1,1,0,1,0,1,1	1	$\alpha + \alpha^2 + \alpha^3$	$1+\alpha + \alpha^2 + \alpha^3$	$\alpha + \alpha^2$	$1+\alpha^3$
0,0,0,0,1,0,1,1,0,0,1,0,0,0,1,1	0	α^2	$1 + \alpha^2$	1	$\alpha + \alpha^2 + \alpha^3$
0,1,1,0,0,1,0,0,1,0,0,0,1,1,1,0	1	$\alpha + \alpha^2$	$\alpha + \alpha^2$	1	$1 + \alpha^2 + \alpha^3$
1,0,1,1,1,0,0,1,0,1,0,1,1,1,0,0	1	$1 + \alpha + \alpha^3$	α^2	$1 + \alpha + \alpha^2$	α^2
1,1,1,0,0,1,0,0,1,0,0,1,1,0,1,1	1	1(0)	$1 + \alpha^2$	0	$1 + \alpha^3$
0,0,1,1,0,0,0,1,0,1,0,1,1,1,0,0	1	α^2	$1 + \alpha^2 + \alpha^3$	1	1
0,0,1,0,1,0,0,0,0,1,1,1,1,1,0,1	0	$1 + \alpha + \alpha^2$	$1+\alpha$	$\alpha + \alpha^2$	$\alpha + \alpha^2$
0,0,1,0,0,1,0,1,1,1,1,1,0,1,0,0	0	$1+\alpha^3$	$1 + \alpha + \alpha^2$	$\alpha + \alpha^2$	$1 + \alpha^2$
0,0,0,0,0,0,1,1,0,1,0,0,1,0,0,1	1	$\alpha + \alpha^3$	$\alpha + \alpha^2$	$1 + \alpha + \alpha^2$	α^2
0,0,0,0,0,0,0,1,0,1,1,0,1,0,0,1	1	$\alpha + \alpha^2$	α^3	1	$\alpha + \alpha^3$
1,1,1,0,1,1,0,0,0,1,0,1,1,0,0,1	1	$\alpha + \alpha^3$	1	$\alpha + \alpha^2$	$\alpha + \alpha^3$
0,0,1,1,0,0,1,1,1,0,0,0,1,1,0,0	1	$\alpha + \alpha^3$	$1+\alpha^3$	1	α^3
0,0,0,0,0,0,0,1,0,1,0,0,1,0,1,1	1	α	1	0	$1 + \alpha + \alpha^2$
0,0,1,1,1,1,1,0,1,0,1,1,0,1,0,1	0	$\alpha + \alpha^3$	$1+\alpha + \alpha^2 + \alpha^3$	$\alpha + \alpha^2$	$1+\alpha$
0,0,1,1,0,1,1,0,0,0,0,0,1,1,1,0	1	1	$\alpha + \alpha^2$	0	$1+\alpha^3$

Table 5.11 DFT Spectra U_k

u_t	**0**	**1**	**3**	**5**	**7**
1,1,1,0,1,1,0,0,0,1,0,1,1,0,0,1	1	$\alpha + \alpha^3$	1	$\alpha + \alpha^2$	$\alpha + \alpha^3$

0,0,1,1,0,0,1,1,1,0,0,0,1,1,0,0	1	$\alpha + \alpha^3$	$1 + \alpha^3$	1	α^3
0,0,0,0,0,0,0,1,0,1,0,0,1,0,1,1	1	α	1	0	$1 + \alpha + \alpha^2$
0,0,1,1,1,1,1,0,1,0,1,1,0,1,0,1	0	$\alpha + \alpha^3$	$1+\alpha + \alpha^2 + \alpha^3$	$\alpha + \alpha^2$	$1 + \alpha$
0,0,1,1,0,1,1,0,0,0,0,0,1,1,1,0	1	1	$\alpha + \alpha^2$	0	$1 + \alpha^3$
0,0,0,0,0,0,1,1,0,1,1,0,1,0,1,1	1	$\alpha + \alpha^2 + \alpha^3$	$1+\alpha + \alpha^2 + \alpha^3$	$\alpha + \alpha^2$	$1 + \alpha^3$
0,0,0,0,1,0,1,1,0,0,1,0,0,0,1,1	0	α^2	$1 + \alpha^2$	1	$\alpha + \alpha^2 + \alpha^3$
0,1,1,0,0,1,0,0,1,0,0,0,1,1,1,0	1	$\alpha + \alpha^2$	$\alpha + \alpha^2$	1	$1 + \alpha^2 + \alpha^3$
1,0,1,1,1,0,0,1,0,1,0,1,1,1,0,0	1	$1 + \alpha + \alpha^3$	α^2	$1 + \alpha + \alpha^2$	α^2
1,1,1,0,0,1,0,0,1,0,0,1,1,0,1,1	1	1	$1+\alpha^2$	0	$1 + \alpha^3$
0,0,1,1,0,0,0,1,0,1,0,1,1,1,0,0	1	α^2	$1 + \alpha^2 + \alpha^3$	1	1
0,0,1,0,1,0,0,0,0,1,1,1,1,1,0,1	0	$1 + \alpha + \alpha^2$	$1+\alpha$	$\alpha + \alpha^2$	$\alpha + \alpha^2$
0,0,1,0,0,1,0,1,1,1,1,1,0,1,0,0	0	$1 + \alpha^3$	$1 + \alpha + \alpha^2$	$\alpha + \alpha^2$	$1 + \alpha^2$
0,0,0,0,0,0,1,1,0,1,0,0,1,0,0,1	1	$\alpha + \alpha^3$	$\alpha + \alpha^2$	$1 + \alpha + \alpha^2$	α^2
0,0,0,0,0,0,0,1,0,1,1,0,1,0,0,1	1	$\alpha + \alpha^2$	α^3	1	$\alpha + \alpha^3$

5.6 Attack Complexities

The attack complexities for the Toy Cipher are calculated in Table 5.12.

Table 5.12 DFT Attack Complexities for Toy Cipher

Cipher Complexity	Same Key	Different Key
Offline Phase	2^{16}	$2^{16.5}$
Online Phase	2^{8}	2^{9}
Total Complexity	2^{17}	$2^{17.25}$

5.7 Summary

In this chapter, the structure of Clock Controlled Ciphers i.e., Alternating Step and Shrinking Generator has been explained with low degree polynomials. Then DFT attack variations have been launched to determine their efficacy. Then structure of Toy Cipher has been explained and two DFT attacks carried out using same and different keys to check practical significance of the attack.

Chapter 6

Conclusion and Future Work

6.1 Introduction

In this chapter, the thesis has been concluded and possible way forward for the future is defined with regard to Application of DFT Attack.

6.2 Conclusion

DFT Attack is a relatively newer technique of cryptanalysis and its application on different new ciphers is important in terms of its efficacy. DFT attack has been tested on Alternate Step Generator [13], Shrinking Generator [15], Toy Block cipher [14] and practical cipher WG-7[9]. This thesis has thoroughly investigated the security of the practical stream cipher WG-7 by applying the Helleseth et al. New Attack [1], [2], selective and fast selective DFT attack [6], [7]. A proposed fast selective DFT attack using annihilator has been applied to considerably reduce the complexity of DFT attack on the cipher.

The predicted key recovery attack employing Yiyuan Luo et al. DFT attack [9] recovers the key with keybits requirements of about $2^{25.5}$ and online computation of about O($2^{29.5}$). Whereas the key recovery attack employing proposed fast selective DFT attack recovers the keys with keybits requirements of about 2^{18} and online computation of about O($2^{23.5}$). The presented results indicate that WG-7 stream cipher is not secure against DFT attack. The offline computation is high as it requires generation and storage of bits in O(2^{161}). The proposed DFT attack has also been compared against Algebraic and FAA attack [10] and found more efficient in terms of keybits and online computation. Algebraic attack requires keybits of about 2^{20} and online computation of about O($2^{54.36}$), whereas FAA requires keybits of about $2^{19.38}$ and online computation of about O($2^{26.73}$). Also WG-7 cipher is found vulnerable mainly due to structure of the cipher. The cipher spectral immunity can be increased by using words from other registers to act as input to nonlinear filter function rather than using single word input.

This thesis establishes the fact on the basis of application of DFT attack against alternate step generator, shrinking generator and toy block cipher that these structures are not vulnerable. This can be interpreted more clearly that all clock controlled stream cipher and block cipher structures are secure against DFT Attacks. The DFT attack can only work efficiently on Non-linear filter generators and LFSR combiner generator ciphers, which are relatively easy to attack because of their structure.

6.3 Future Work

There is a need to develop software to calculate linear complexity of the cipher. The software should test each cipher to find its linear complexity and can be easily used by cryptographers and researchers for testing and evaluation purposes. Moreover, there is need to improve the DFT calculation method [7] to make the DFT attack more efficient. WG-8 and WG-16 cipher security needs to be tested like WG-7 as they belong to the same cipher family and their design may be vulnerable to DFT attack with the use of the annihilator.

6.4 Summary

In this chapter, the thesis has been concluded and future work is proposed which mainly focuses on effectiveness of DFT attacks against all type of symmetric ciphers.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] S. Rønjom and T. Helleseth, “A New Attack on the Filter Generator” , IEEE Transactions on Information Theory, vol. 53, no. 5, pp. 1752–1758, 2007.

[2] S. Rønjom and T. Helleseth, “Attacking the Filter Generator over GF(2^m)” , in Arithmetic of Finite Fields, First International Workshop, WAIFI 2007, Madrid, Spain, June 21-22, 2007, Proceedings, Madrid, Spain, 2007, pp. 264–275.

[3] G. Gong. Sondre Rønjom and T. Helleseth, “On Attacks on Filtering Generators using Linear Subspace Structures”, in Sequences, Subsequences, and Consequences, International Workshop, SSC 2007, Los Angeles, CA, USA, May 31 - June 2, 2007, Revised Invited Papers, Los Angeles, CA, USA, 2007, pp. 204–217.

[4] Sondre Rønjom, Guang Gong, and Tor Helleseth, “A Survey of Recent Attacks on the Filter Generator”, in: Serdar Boztas and Hsiao-Feng Lu, eds., Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, Lecture Notes in Computer Science, volume 4851, pp. 7-17, Springer-Verlag, Berlin, 2007.

[5] Sondre Rønjom and Tor Helleseth, “The Linear Vector Space Spanned by the Nonlinear Filter Generator”, in Sequences, Subsequences, and Consequences, Lecture Notes in Computer Science, volume 4893, pp. 169-183, Springer-Verlag, Berlin, 2007.

[6] T. Helleseth. Guang Gong, Sondre Rønjom and H. Hu, “Fast Discrete Fourier Spectra Attacks on Stream Ciphers,” IEEE Transactions on Information Theory, vol. 57, no. 8, pp. 5555–5565, 2011.

[7] G. Gong. (2011), “A Closer Look at Selective DFT Attacks”. Department of Electrical and Computer Engineering; University of Waterloo. Waterloo, Ontario, Canada. [Online]. Available: http://cacr.uwaterloo.ca/techreports/2011/cacr2011-35.pdf .

[8] J. Wang and K. Chen, “Improvement of Discrete Fourier Transform Attack,” Journal of shanghai Jiaotong University (Science), vol. 46, no. 2, pp. 285–288, 2012.

[9] G. G. Yiyuan Luo, Qi Chai and X. Lai, “A Lightweight Stream Cipher WG-7 for RFID Encryption and Authentication”, in Proceedings of the Global Communications Conference. GLOBECOM 2010, 6-10 December 2010, Miami, Florida, USA, 2010, pp. 1–6.

[10] J. Pieprzyk. Mohammad Ali Orumiehchiha and R. Steinfeld, “Cryptanalysis of WG-7: a Lightweight Stream Cipher”, Cryptography and Communications, vol. 4, no. 3-4, pp. 277–285, 2012.

[11] Sultan Zayid Mohammed Al-Hinai, “Algebraic Attacks on Clock-Controlled Stream Ciphers”. PhD thesis, Queensland University of Technology, 2007.v

[12] Cryptool 2 Cryptography for Everybody. [Online]. Available: http://www.cryptool.org/en/cryptool2

[13] C.Gunther. “Alternating Step Generators Controlled by deBruijin Sequences”. In D.Chaum and W. Price, editors, Advances in Cryptology-EUROCRYPT 87, volume 304 of Lecture Notes in Computer Science, pages 5-14. Springer-Verlag, 1988.

[14] H. Feistel, "Cryptography and Computer Privacy", Scientific American, vol. 228, no. 5, pp. 15-23, 1973.

[15] W. Meier and O. Staffelbach, ”The Self-Shrinking Generator”, Advances in Cryptology: Proceedings of Eurocrypt 94, Lecture Notes in Computer Science, Springer-Verlag, vol. 950, pp. 205-214, 1994.

[16] Y. Nawaz and G. Gong, “WG: A Family of Stream Ciphers with Designed Randomness Properties”, Inf. Sci., vol. 178, no. 7, pp. 1903–1916, 2008.

[17] Bluetooth Specification v1.1, 1999. http://www.bluetooth.com/. 12, 39.

[18] Tor Helleseth, Michal Hojsík, and Sondre Rønjom, “Algebraic Attacks on Filter and Combiner Generators”, in Enhancing Cryptographic Primitives with Techniques from Error Correcting Codes, NATO Science for Peace and Security Series, Sub-Series D: Information and Communication Security, volume 23, pp. 39-48, IOS Press, Amsterdam, 2009.

[19] N. Courtois and W. Meier, “Algebraic Attacks on Stream Ciphers with Linear Feedback”, Advances in Cryptology-Eurocrypt’2003, Lecture Notes in Computer Science, vol. 2656, pp. 345-359, Springer, 2003.

[20] F. Armknecht and M. Krause, “Algebraic Attacks on Stream Combiners with Memory”, Advances in Cryptology-CRYPTO 2003, Lecture Notes in Computer Science, vol. 2729, pp. 162-176, Springer-Verlag, 2003.

[21] N. Courtois, “Fast Algebraic Attacks on Stream Ciphers with Linear Feedback”, Advances in Cryptology-Crypto’2003, Lecture Notes in Computer Science, vol. 2729, pp. 176-194, Springer-Verlag, 2003.

[22] F. Armknecht, “Improving Fast Algebraic Attacks”, Proceedings of Fast Software Encryption 2004, Lecture Notes in Computer Science, vol. 3017, pp. 65-82, Springer-Verlag, 2004.

[23] P. Hawkes and G. G. Rose, “Rewriting Variables: the Complexity of Fast Algebraic Attacks on Stream Ciphers”, Advances in Cryptology-Crypto 2004, Lecture Notes in Computer Science, no. 3152, pp. 390-406, Springer-Verlag, 2004.

[24] FF. Armknecht and G. Ars, “Introducing a New Variant of Fast Algebraic Attacks and Minimizing their Successive Data Complexity”, Mycrypt 2005 (International Conference on Cryptology in Malaysia), Lecture Notes in Computer Science, vol. 3715, pp. 16-32, 2005.

[25] K. Chen. Jingjing Wang and S. Zhu, “Annihilators of Fast Discrete Fourier Spectra Attacks,” in Advances in Information and Computer Security -7th

International Workshop on Security, IWSEC 2012, Fukuoka, Japan, November 7-9, 2012. Proceedings, Fukuoka, Japan, 2012, pp. 182–196.

[26] K. Chen. Jingjing Wang, Xiangxue Li and. Zhang, “Attack based on Direct Sum Decomposition against the Nonlinear Filter Generator”, in Progress in Cryptology - AFRICACRYPT 2012 - 5th International Conference on Cryptology in Africa, Ifrance, Morocco, July 10-12, 2012. Proceedings, Ifrane, Morocco, 2012, pp. 53–66.

[27] K. M. Xinxin Fan and G. Gong, “WG-8: A Lightweight Stream Cipher for Resource-Constrained Smart Devices”, ICST Trans. Security Safety, vol. 3, p. e4, 2015. [Online]. Available: http://dx.doi.org/10.4108/sesa.2.3.e4

[28] Xinxin Fan and Guang Gong. "Specification of the Stream Cipher WG-16 Based Confidentiality and Integrity Algorithms". University of Waterloo, Waterloo, ON, Canada, Tech. Rep. CACR 6 (2013): 2013.

[29] Herbert, Vincent. "Des Codes Correcteurs Pour Sécuriser l'information Numérique". PhD diss., Université Pierre et Marie Curie-Paris VI, 2011.

[30] Wu, Di, Wenfeng Qi, and Huajin Chen. "On the Spectral Immunity of Periodic Sequences Restricted to Binary Annihilators". *Designs, Codes and Cryptography* (2014): 1-13.

[31] S.W. Golomb, G. Gong. “Signal Design for Good Correlation” For Wireless Communication, Cryptography and Radar. Cambridge University Press, Cambridge (2005)