SECURITY THROUGH ELLIPTIC CURVES FOR WIRELESS NETWORKS IN MOBILE DEVICES

Dr. EKAMBARAM. KESAVULU REDDY

SECURITY THROUGH ELLIPTIC CURVES FOR WIRELESS NETWORKS IN MOBILE DEVICES

The Thesis Submitted to

SRI VENKATESWARA UNIVERSITY

In Partial Fulfillment for the award of the degree of

DOCTOR OF PHILOSOPHY

IN

COMPUTER SCIENCE

By

EKAMBARAM. KESAVULU REDDY

Under the guidance of

Prof. P. Govindarajulu

DEPARTMENT OF COMPUTER SCIENCE
SRI VENKATESWARA UNIVERSITY
TIRUPATI-517502

JULY 2011

To the major candles in my Life

My Parents

Smt. Ekambaram .Rangamma And Sri. Ekambaram. Chenga Reddy

About

The entire book is research-oriented and developing the various investigations related to Side-Channel Attacks to Simple Power Analysis Attacks. In view of this book we develop the algorithms as well as implementing the programming approach. We develop a new cryptosystem for Simple Power Analysis Attacks comparing with RSA cryptosystems. This book is suitable for Post-graduate in Engineering when adopting the syllabus by various universities.

BOOK ORGANIZATION

The chapters are planned in a systematic way. In order to developed the gap between existing and present system. Each concept is explained in easy to understand manner supported with analysis. This book contains six chapters.

The First chapter presents the evolution of Elliptic Curve Cryptography and related topic of elliptic curve cryptosystems.

The chapter 2 reviews the related work on side channel attacks on Elliptic Curve cryptosystems. This review focused on introductory information on side channel attacks and **known methods for protection against side channel attacks with brief effective assessment.**

The chapter 3 presents different methods that are used to compute the scalar multiplication (ECSM), which is the core operation of ECCs, and the various costs associated with them. Further an **analytic result for Major and Minor collisions, probability of occurrence as well as effect of the fixed sequence window** method are discussed in this chapter.

The chapter 4 deals with the **Modified Montgomery Inverse algorithm which is resistant to SPA attacks and Montgomery Inverse algorithm**, which is used to

eliminate the number of iterations of the main loop that directly leaks the value of the function f. Further it is proved mathematically that f is uniformly distributed with a significant reduction.

In chapter 5, some background on S/MIME and the protocol summarization and the performance advantages to be obtained by using elliptic curves in the wireless environment are discussed. An application of elliptic curves in identity based encryption that may help to launch deployment of a public key infrastructure, is presented. We present the performance of encryption and decryption time increases from bits 256 to 2048 depending upon different platforms. The encryption and decryption performance time increases or decreases depend upon the type of messages. **We provide the performance analysis between EKR Modified Montgomery Inversion Algorithm and Fast Exponentiation**

In chapter 6, the performance of ECC applications on various mobile devices, and suitability of PKC scheme of Elliptic Curve Cryptosystems for constrained environment are discussed. Further PKC scheme effectiveness and suitability are discussed with reference to RSA and DSA. **We develop a new cryptosystem i.e EKR Cryptosystem to resistance against Simple Power Analysis Attacks with one Public key and one Private Key to resistance against to Simple power Analysis Attacks** in Side-Channel Attacks in elliptic curve cryptosystems based on EKR Modified Montgomery inversion algorithm

In Chapter 7 A brief conclusions including from First chapters to Sixth Chapter

ACKNOWLEGEMENTS

All praise and glory of the Lord Venkateswaraswamy for supporting and guiding me every day of my life and throughout all my work.

I confiscate this opportunity to express my sincere gratitude to my Research supervisor, Prof.P.Govindarjulu for his ceaseless encouragement, understanding, guidance and his genuine knowledge.

I thank all the professors who have taught me and enriched my mind with their precious knowledge, especially **Professors in the department, Head, Department of computer science, Chairman, Board of Studies in Computer Science.** I also thank my colleagues their suggestions and comments, especially Teaching and Non-Teaching.

I thank to my parents for their love and support, and care and to encourage to completion of my research work. I thank my aunts to encourage and help to completion of my research work.

Finally, I will be thankful to my wife ***Mrs .E. Gouri Sree and my children E.Digvijay Kumar Reddy, E. Spandhana Reddy*** for bearing with me every difficulty I faced and for helping me overcome it.

(E. KESAVULU REDDY)

ABSTRACT

The basic principle is "A Function easy to evaluate but its inverse is Infeasible unless a secret key is known".

Some implementations of cryptographic algorithms often leak "*side channel information*." Side channel information includes power consumption, electromagnetic fields and timing to process. **Side channel attacks, which use side channel information leaked from real implementation of cryptographic algorithms** .The security of system lies in extracting k from points P and Q on elliptic curve. **These attacks are two types. They are Simple Power Analysis Attacks (SPA) and Differential Power Attacks (DPA). These attacks monitor the power consumption or the Electromagnetic emanations of a device example smart cards or mobile devices. The attacker's goal is to retrieve partial or full information about a long-term key that is employed in several Elliptic Curve Scalar Multiplication (ECSM) executions.**

We analyze and study **Major and Minor Collisions** and provide an analytic **result for their probability of occurrence as well as effect of the fixed sequence window** method.

We present performance comparisons of 163 bit equivalent 1024 bit RSA on elliptic curve exponentiation *for general curves. Also we provide the* ***performance analysis between EKR Modified Montgomery Inversion algorithm to Fast Exponentiation.***

We implement a secret key to avoid retrieving the valuable information by the attacker through Simple Power Analysis Attacks. The secret key t is proposed as a prime integer in E Kesavulu Reddy modified Montgomery inversion algorithm based on window method between 1 to n **and t is a number of pre-computed values. The developer or user to assigned the value of secret key t.**

We propose a new cryptosystem based on EKR Modified Montgomery Inversion algorithm with one public key and one private key to resist against to Simple Power Analysis Attacks. **The new cryptosystem allows the cryptographic information like RSA cryptosystems and also resist against to simple power analysis attacks.**

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

IFP	**Integer Factorization Problem**
DLP	**Discrete logarithm Problem**
ECDLP	**Elliptic Curve Discrete Logarithm Problem**
ECC	**Elliptic Curve Cryptosystems**
RSA	**Rivest- Shamir-Addle man Algorithm**
SPA	**Simple Power Analysis Attacks**
DPA	**Differential Power Analysis Attacks**
ECSM	**Elliptic Curve Scalar Multiplication**
ECDH	**Elliptic Curve Diffe-Hellman key exchange**
DSA	**Digital Signature Algorithm**
ECDSA	**Elliptic Curve Digital Signature Algorithm**
SCA	**Side-Channel Attacks**
DES	**Data Encryption Standard**
SEMD	**Single-Exponent, Multiple-Data Attack**
MESD	**Multiple-Exponent, Single-Data Attack**
ZEMD	**Zero-Exponent, Multiple-Data Attack**
EKR Modified Montgomery Inversion	**E.Kesavulu Reddy Modified Montgomery Inversion**
AES	**Advanced Encryption Standard**
DH	**Diffe-Hellman Key Exchange**
MIME	**Multipurpose Internet Mail Extensions**
S/MIME	**Secure Multipurpose Internet Mail Extensions**
CA	**Certificate Authority**
NIST	**National Institute of Standard Technology**

NAF	**Non-Adjacent Form**
ID Based Encryption	**Identity-Based Encryption**
BDH	**Bilinear Diffe-Hellman**
PKC	**Public Key Cryptography**
QS	**Quadratic Sieve**
NFS	**Number of field sieve**
FIPS	**Federal Information Processing Standards**
PB	**Polynomial Bases**
NB	**Normal Bases**
EKR Cryptosystem	**E.Kesavulu Reddy Cryptosystem**

CHAPTER 1

1. INTRODUCTION

1.1. Introduction to cryptography

In 1976 white Field, Diffe and Martin Hellman introduced the concept of public key cryptography. These public key cryptosystems are unsecure, broken and some of them are also impractical. Today the following types of systems are considered as secure and efficient

1. Integer Factorization Problem (IFP):
2. Discrete logarithm Problem (DLP):
3. Elliptic Curve Discrete Logarithm Problem (ECDLP)

1.2. Elliptic curve cryptosystems

In 1985 Vector Miller [Miller] and Neal Kolbitz [Kolbitz] independently proposed Elliptic Curve Cryptosystems (ECC). Since 1985 Elliptic Curve Cryptography has received intense security from cryptographers, mathematicians and computer scientists around the world. Elliptic curve cryptography has no weaknesses and highly secure than public key systems.

Elliptic curve Discrete Logarithm problem is the best algorithm to solve the underlying mathematical problems which requires more exponential time than RSA and DSA. The sub-exponential time algorithm in RSA and DSA are based on Integer Factorization Problem (IFP) and Discrete Logarithm Problem (DLP). For this reason, the elliptic curve cryptography offers security equivalent to RSA and DSA while using smaller key sizes. The exponentiation algorithm with secret key based cryptosystems such as RSA and DLP suffers Simple Power Analysis Attacks (SPA) and Differential Power Analysis Attacks (DPA).

The key size and computing power increases in Elliptic Curve Cryptosystems compared to the other public key cryptosystems. The benefits of this higher –strength per bit include:

- Higher speeds
- Lower Power Consumption
- Bandwidth Savings
- Storage efficiency
- Smaller certificates

These advantages are particularly beneficial in applications where bandwidth, processing capacity and power availability or storage is constrained. Such applications include

- Chip cards
- Electronic Commerce
- Web Servers
- Cellular Telephones
- Pagers

To implementing the cryptosystems on smart cards and personal wireless devices enough memory not required for the security of keys. First, we need to check the resistance of devices against side channel attacks. Side channel include execution time [42] power consumption [43], [47], [48], electromagnetic emanations [24],[25] and computational errors due to faults in hardware[27].

1.3. Motivation

The basic principles of Power Analysis Attacks on elliptic curve cryptosystems have been discussed in [18]. There are basically two things can be done to counteract both Simple Power Analysis Attacks (SPA) and (First Order) Differential Power Analysis Attacks (DPA). First one has to randomize the expressions of calculated points. This can be done by using randomized projective coordinates. SPA and DPA are based on monitoring the power consumption of cryptographic token while executing an algorithm that manipulates the secret key. The traces of the measured power are analyzed to obtain significant information about the key. In SPA, a single power trace can reveal large features of the algorithm being executed such as the iterations of the loop. However the cryptosystem

specific operations such as point Doubling and Adding in elliptic curve cryptosystems can be identified [28]. In order to resist SPA attack, the steps of the algorithm need to be uniform across different executions.

On the other hand even if SPA attack does not apply the DPA attacks rely on first collecting several power traces for executions of the algorithm using the same key. If the traces are processed the various statistical tools to identify the features of the processed data at specific instant of the cryptographic algorithms. Information about the processed data can directly lead to processing the specific data at this specific instant.

Oswald [38] presents the SPA attacks an elliptic curve point multiplication algorithm using markov models. Oswald Also demonstrates the attack on an Addition Subtraction algorithm which is similar to one described by Morain et.al [56]. In [55] Moller proposes a method of elliptic curve point multiplication that can be shown to provide security against side channel attacks. The algorithm was a special signed-digit encoding to ensure that point doubling and point additions occur in uniform pattern.

Joy and Quisquater [37] investigate the Hessian parameterization of elliptic curves as a step towards resistance against such attacks in the context of Elliptic Curve Cryptography. In [4], Billet and Joy use an extended form of the Jacobi Quadratic and derived a Unified Addition Formulae for adding or doubling points with only 13 field multiplications. This is the fastest known addition law for elliptic curves whose order is multiples of 2.

In [70], Sakai and Sakurai have introduced a new side channel attack on exponent recoding and proposed new algorithm for side channel attacks on exponent recoding. The exponent recodings are based on width-w NAF or Fractional Window and represents the unsigned / signed fractional window.

In [29], MAMYIYA & MIYAJI proposes for DPA in Elliptic Curve Cryptosystems has been improved to the Reined Power Analysis (RPA), which exploits a special point with zero value and reveals secret key. An elliptic curve has a special point (0,y),(x,0). RPA utilizes such a feature that the

power consumption of '0' is distinguishable from with a non-zero element. RPA is generalized to Zero-Value Register attack (ZRA) by [29]. ZRA utilizes a special future need a lot of each different operations stored in auxiliary registers, one of which happens to become zero.

In [30], Kim and Takagi proposed the security of n γ pairing over binary fields in the context of side channel attacks. In [46], Medwed and Xavier Standeart proposes a fresh Re-keying scheme that is specially suited for challenge-response protocols such as used to authenticate tags. They also evaluate the resistance against fault and side channel analysis and introduce a simple architecture for VSLI and implementing RFID tags. The Re-keying scheme provides better security for side channel and fault attacks for low-cost devices. In [47], Messerges, Dabbish and Sloan presents power analysis techniques used to attack DES are reviewed and analyzed. Also they propose noise ratio approach which helps to design smart card solutions and these are secure against power analysis attacks.

In [3], Bayam and ORS propose a RSA cryptosystem on hardware and modified it to be resistant against DPA attacks. First implementing modular exponentiation and realized with montgomery modular multiplication on RSA cryptosystem which is resistant to power analysis attacks. In Meuenaer and Xavier Standaert [25], propose a new countermeasure to protect the symmetric encryption of application messages against DPA attacks. In further they propose stealthy node compromises should be considered when securing wireless sensor networks. Node capture is considered as one of the most critical issue in the security of wireless sensor networks.

In [62], Santosh Ghosh and Mukhopadhyay propose a powerful tool for side channel analysis retrieving secrets embedded in cryptographic devices such as smart cards. They also propose an Efficient Division Algorithm protected against Simple Side channel Analysis.

In [41], Kirtane and Rangan proposed a secure implementation of Hensel Lifting and adopted the existing implementations of modular exponentiation. The CRT-RSA decryption algorithms to make them secure against major side channel attacks as well as compatible with each other. Regazzonj and Eisenbarth propose a Fault detection circuiting which is added to protect it against fault injection

attacks and analyze the resistance of the modified device to power attacks. They focus on only one component in the cryptographic device namely the S-Box in the AES and Kasumi Ciphers.

In [40], Koschuch and Grobschadi implement a Light-Weight Secure Sockets Layer protocol with a focus on small code size and low memory usage. They integrate a generic public key crypto library into this SSL stack to support elliptic curve cryptography over arbitrary prime and binary fields.

The reader can refer for recent surveys on these attacks and their countermeasures [38],[37],[4],[29],[30],[46],[31],[3],[25],[39],[62],[42],[16] and[11]).

1.4. Proposed problem

Several authors studied different techniques for avoiding the side channel attacks in elliptic curve cryptosystems. In particular Nevine Maurice Ebied's side channel attacks in Elliptic curve Cryptosystems and analysis of different algorithms proposed in her thesis entitled "key Randomization Countermeasures to Power Analysis Attacks on Elliptic Curve Cryptosystems. Further, the actual value of the key processed either a bit level or a digit level has been discussed. The Randomization is performed before the Elliptic Curve Scalar Multiplication (ECSM) execution for which an efficient SPA-resistant algorithm is used. A candidate key value randomization method is the Key Splitting.

In Key Splitting methods there are two problems in the above thesis i.e.

1. **Analytical result for minor collisions has not been discussed for the occurrence as well as the effect of the Fixed-sequence recoding them.**
2. **The distribution of function f resulting from the Almost Montgomery Inverse algorithm over the integers range [1, 2^m-1].**

The present thesis deals with the study on minor collisions and to provide Analytical results for their probability of occurrences as well as the effect of Fixed-Sequence recoding on them. **Further study the distribution of f resulting from the Almost Montgomery Inverse algorithm over the integer's range [1, 2^m-1]** and its relation to modulus.

To analyze and provide the advantages of elliptic curve cryptography for Wireless security and **compare the bit level of security from EKR Modified Montgomery inversion algorithm to Fast exponentiation**. To analyze and provide the performance of elliptic curves and **develop a new EKR cryptosystem with one public key and private key to resistant to simple power analysis through proposed EKR Modified Montgomery Inversion algorithm.**

CHAPTER. 2.
OVERVIEW OF THE SIDE -CHANNEL ATTACKS

2.1. Introduction

Smart cards are one of the major application fields of cryptographic algorithms, and may contain sensitive data, such as RSA private key. Some implementations of cryptographic algorithms often leak "*side channel information*." Side channel information includes power consumption, electromagnetic fields and timing to process. Side channel attacks, which use side channel information leaked from real implementation of cryptographic algorithms, were first introduced by Kocher [55], [18]. Side channel attacks can be often much more powerful than mathematical cryptanalysis

In recent years, security has become an important issue in the design of computer systems as more critical services are provided over the Internet and many networked devices communicate through wireless channels. Normally cryptographic algorithms are used to provide basic security functions such as confidentiality, authentication, and digital signature. Therefore their security is vital to any security mechanisms or protocols.

The security of cryptographic algorithms such as block ciphers and public-key algorithms relies on the secrecy of the key. Traditionally, when cryptanalysists examine the security of a cryptographic algorithm, they try to recover the secret key by observing the inputs and outputs of the algorithm. Assuming this type of attack models, cryptologists have made commonly-used cryptographic algorithms secure against such attacks. However, a real computing device not only generates the outputs specified in algorithms but also inevitably produces some other information such as timing and power. These types of information, called side-channel information, can be exploited in side-channel attacks to retrieve secret keys. Side channel attacks have successfully broken many algorithms.

Mobile devices and sensor nodes that work in the field, not protected by physical security mechanisms, are more vulnerable to side-channel attacks. Among all side channel attacks, power analysis, which exploits the power consumption of a cryptographic system, can be carried out easily. It

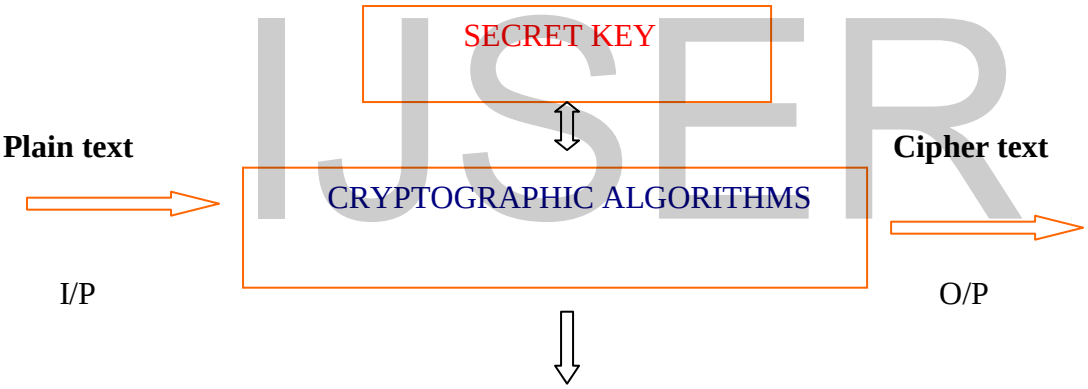
is very effective in breaking cryptographic algorithms [8], [2], [76]. In this type of attacks, adversaries learn what operations are performed and what data are processed by analyzing the power traces of computations. They can then figure out part or all of the bits in the secret key. A lot of work has been done on the countermeasures against power analysis attacks. Many countermeasures studied software implementations, trying to make the power consumption of a crypto system either random or identical for different keys [8], [71].

The software countermeasures usually only work for specific algorithms and have large performance overhead. Very often, the countermeasures are found vulnerable to more advanced attacks. For example, randomized automata [71] for Elliptic Curve Cryptography operations are found not secure and can be broken by many new attacks [24], [17], [3], [72]. There are some hardware countermeasures [37], [44], [49]. The use of self-timed dual-rail logics is proposed in [37] to provide protection to power analysis attacks. Dynamic and differential logic is also employed in [49].

Both the two logic styles can make the power consumption of logic gates independent of the data values. One of the drawbacks of these methods is large area and power overhead. The method described in [44] compensates the power consumption of the system with voltage and frequency scaling techniques and an analog current injection circuit. However, besides large power overhead, frequency scaling affects the performance of software and may make the system vulnerable to timing attacks. It should be pointed out that memory security is not the primary focus of these above techniques. Some other hardware countermeasures are at the architectural level [38], [87]. They randomize either the register renaming [38] or the issue of instructions in the instruction window [87] to make the power analysis attacks more difficult. However, these methods may not fit well with the low-end processors, which typically do not have register renaming mechanism or large instruction window to support out-of-order execution.

2.2. Side-channel attacks

An encryption device is perceived as a unit that receives plaintext Input and produces cipher text output and vice-versa. Attacks were earlier based on either knowing the cipher text or both or on the ability to define what plaintext is to be encrypted and then seeing the results of the encryption. To day, it is known that encryption devices have additional output and often additional inputs which are not the plaintext or the ciphertext. Encryption devices producing timing information that is easily measurable, radiation of various sorts power consumption statistics and more. Often the encryption device also has additional "Unintentional "inputs such as make use of some or all of this information, along with other cryptanalytic techniques, to recover the key the device using.



`

Timing Attacks

Fault Injection Attacks

Electromagnetic Attacks

Fig.2.1. Basic level of Side-Channel Attack

2.2.1. Security problem and countermeasures

Several countermeasures to power analysis attacks have been suggested that alter the internal operation of the device under attack. So that the secret information content in the power analysis was leaked through the internal changes. Due to this type of changes the value of signature utilization is reduced. While these countermeasures may be effective. They require modification to the internal

structure of the processing blocks. The basic information are passed from PC through PT(Power Tracer) to the smartcard(Electronic applications) to trace outs the encryption form of analysis in Power Tracer(PT).Power Tracer send the trigger signal to the digital oscilloscope[55]. The Digital oscilloscopes process these power data and transmit them to PC by network cable .Finally power data will be displayed by tracers in cryptanalysis software which was installed in PC. Data power is displayed in digital oscilloscope (Fig.2.2 shows).

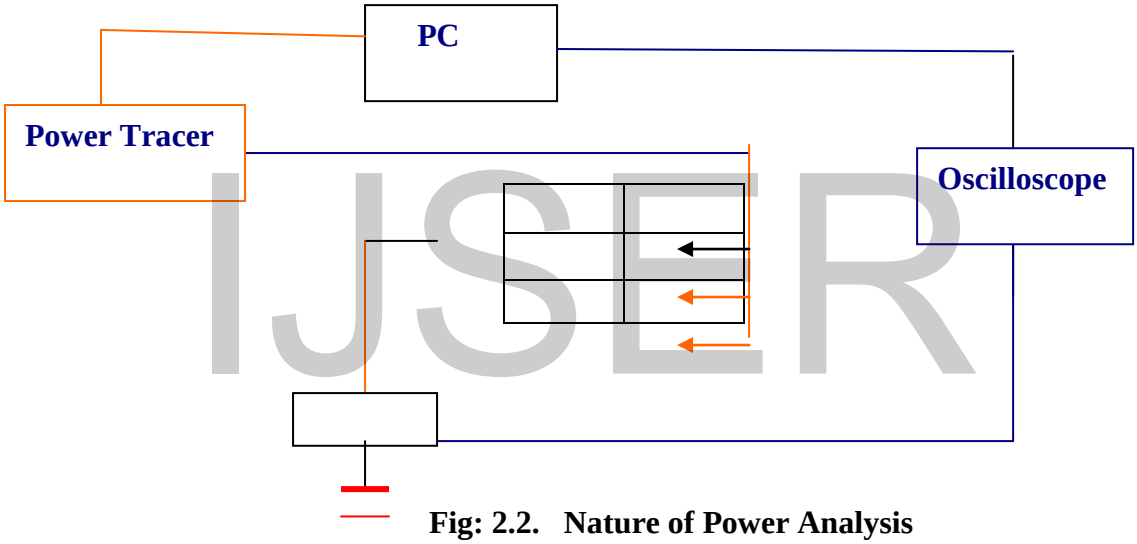


Fig: 2.2. Nature of Power Analysis

Side Channel Analysis techniques are of based on the amount of time required for the attack and the analysis depends upon the type of attack. The most common types of attacks of Side Channel Information are

1. Timing Attacks.
2. Simple Power Analysis Attacks
3. Differential Power Analysis Attacks.

2.2.1.1. Timing attacks

Timing attacks are based on measuring the time it takes for a unit to perform operations. This information can lead to information about the secret keys. For example, by carefully measuring the amount of time required to perform private key operations, an attacker might find fixed Diffe-Hellman exponents, factor RSA keys and break other cryptosystems [51]. If a unit is vulnerable, the attack is computationally simple and often requires only known cipher text.

2.2.1.1.1. Countermeasures against Timing Attacks

2.2.1.1.1.1. Adding Delays

The most obvious way to prevent timing attacks is to make all operations take exactly the same amount of time. Unfortunately this often difficult. If a time is used to delay returning results until a pre-specified time factors such as the system responsiveness or power consumption may still change when the operation finishes in a way that can be detected [52].According to [52] fixed time implementations are likely to be slow. Many performance optimizations can not be used since all operations must take as long as slowest operations. The number of samples required increases roughly as the sequence of the timing noise [52]. So random delays can make the attack a bit more difficult, but still possible.

2.2.1.1.1.2. Timing Equalization of Multiplication and Squaring

The time taken by the unit for the performance of multiplication and the performance of exponentiation actions should be set to be similar. Due to this quality an attacker will not be able to learn if, when and how many multiplications are made and how many exponentiations. This technique prevents timing attacks against the exponentiation operations. Those are performed as part of symmetric encryption and which are subject to the most common attacks.

Cryptosystems often take slightly different amount of time to process different outputs. Performance characteristics typically depend on both the encryption key and the input data plaintext or cipher text. However as shown in[51] attacks exist which can exploit timing measurements from vulnerable systems, to find the entire Secret key.

Timing measurements are fed into a statistical model that can provide the guessed key bit with some degree of certainty. Computing the variances is easy and provides a good way to identify correct exponent bit guesses. The numbers of samples needed to gain enough information to allow the recovery of the key are determined by the properties of the signal and noise. More samples are required for more noise generally, error correction techniques increase the memory and processing requirements for the attack, but can greatly reduce the number of samples required[42].

2.2.1.1.1.3. Remote Timing Attacks

For over two decades, timing attacks have been an active area of research within applied cryptography. These attacks exploit cryptosystem or protocol implementations that do not run in constant time. When implementing an elliptic curve cryptosystem with a goal to provide side-channel resistance, the scalar multiplication routine is a critical component. In such instances, one attractive method often suggested in the literature is Montgomery's ladder that performs a fixed sequence of curve and field operations. Billy Bob Brumley and Nicola Tuveri [5] describe timing attack vulnerability in Open SSL's ladder implementation for curves over binary fields. We use this vulnerability to steal the private key of a TLS server where the server authenticates with ECDSA signatures. Using the timing of the exchanged messages, the messages themselves, and the signatures, we mount a lattice attack that recovers the private key.

2.2.1.2. Simple Power Analysis (SPA) Attacks

Simple Power Analysis is generally based on looking at the visual representation of the power consumption of a unit performs an encryption operation. Simple Power Analysis is a technique that involves direct interpretation of power consumption measurements collected during cryptographic operations. SPA can yield information about device's operation as well as key material.

The attacker directly observes a system's power consumption. The amount of power consumed varies depending on the microprocessor instruction performed. Large features such as DES rounds, RSA operations etc may be identified, since the operations performed by the microprocessor vary

significantly during different part of these operations. Similarly many DES implementation have visible differences within permutations and shifts and can thus be broken using SPA [52].

Because SPA can reveal the sequence of instructions executed. It can be used to break cryptographic implementations in which the execution path depends on the data being processed such as DES key schedule, DES permutations, comparisons, multipliers and exponentiations. Most cryptographic units are tested and found to be vulnerable to SPA attacks, though according to [52] it is not difficult to design a system that should not be vulnerable to such attacks.

2.2.1.2.1. Countermeasures against Simple power Analysis Attacks

2.2.1.2.1.1. Power Consumption Balancing

Power consumption balancing techniques should be applied when possible dummy registers and gates should be added on which is useless. When ever an operation is performed in hardware, a complementary operation should be performed on a dummy element to assure that the total power consumption of the unit remains balanced according to some high value. such techniques by which the power consumption is constant and independent on input and key bits, presents all sorts of power consumption attacks such as SPA..

2.2.1.2.1.2. Hessian Elliptic Curves

In [55], Liardet and Smart suggest to represent elliptic curves as the intersection of two quadrics in P3 as a means to protect against side-channel attacks. Considering the special case of an elliptic curve whose order is divisible by 4 [55] they observe that the same algorithm can be used for adding and doubling points with 16 multiplications (see also [8] . Using [46] the proposed Hessian parameterization, only 12 multiplications are necessary for adding or doubling points. The Hessian parameterization gives thus a 33% improvement over the Jacobi parameterization. Another advantage of the Hessian parameterization is that points are represented with fewer coordinates, which results in substantial memory savings.

2.2.1.2.1.3. Lightweight SSL Implementation Resistant to Side-Channel Attacks

Ever-growing mobility and ubiquitous wireless Internet access raise the need for secure `communication with devices that may be severely constrained in terms of processing power, memory capacity and network speed. We [49] describe a lightweight implementation of the Secure Sockets Layer (SSL) protocol with a focus on small code size and low memory usage.

We [49] integrated a generic public-key crypto library into this SSL stack to support elliptic curve cryptography over arbitrary prime and binary fields. Furthermore, we aimed to secure the SSL handshake against side-channel attacks (in particular simple power analysis) by eliminating all data-dependent or key-dependent branches and memory accesses from the arithmetic operations and compare the resulting performance with an unprotected implementation.

Our lightweight SSL stack has only 6% of the code size and RAM requirements of Open SSL, but outperforms it in point multiplication over prime fields when no appropriate countermeasures against side-channel attacks are implemented.

With such countermeasures, however, the execution time of a typical SSL handshake increases by roughly 50%, but still completes in less than 160 msec on a 200 MHz PDA when using an elliptic curve over a 192-bit prime field.

2.2.1.2.1.4. A Simple Power Analysis Attack on the Serpent Key Schedule

Serpent was designed and submitted as an AES candidate proposal by Ross Anderson, EliBiham, and Lars Knudsen [2]. We describe an SPA attack on an 8-bit smart card implementation of the Serpent block cipher. Our attack uses measurements taken during an on-they key expansion together with linearity in the cipher's key schedule algorithm to drastically reduce the search time for an initial key. The results here show that Hamming weight measurements from 8-bit smart card implementations of Serpent's key schedule reveal enough side channel information to uniquely determine a 256-bit initial key in a few milliseconds.

More generally, we have shown that LFSRs may be very susceptible to SPA attacks and those algorithm designs can greatly accelerate side channel attacks. Kevin J. Compton¤ Brian Timm*y* Joel

VanLavenz [24] suspect that the attack presented here can be made error-robust, as well, since we use only the first 200 rows of the reduced row echelon matrix [A "/e"]. The remaining 328 rows could be used for error correction. Also, pre-shift Hamming weights could be used.

2.2.1.2.1.5. Exponent Recodings for the Exponentiation against Side Channel Attacks

SAKAI & SAKURAI [76] propose a new side channel attack, where exponent recodings for public key cryptosystems such as RSA and ECDSA are considered. The known side channel attacks and countermeasures for public key cryptosystems were against the main stage (square and multiply stage) of the modular exponentiation (or the point multiplication on an elliptic curve).

AKAI and SAKURAI are [76] compute the exponent recoding has to be carried out before the main stage. There are some exponent recoding algorithms including conditional branches, in which instructions depend on the given exponent value. Consequently exponent recoding can constitute an information channel, providing the attacker with valuable information on the secret exponent. The width-w NAF and the unsigned/signed fractional window representation are used to recover the secret exponent on attack for exponent recoding in proposed algorithms

2.2.1.2.1.6. Countermeasures On *ηT pairing* Over Binary Fields

Since many efficient algorithms for implementing pairings have been proposed such as ηT pairing and the Ate pairing, pairings could be used in constraint devices such as smart cards. They are [38] investigated the security of the ηT pairing on supersingular curves over characteristic two against timing attack, SPA and DPA. To avoid such attacks we proposed explicit algorithms of the ηT pairing using randomized projective coordinates. We demonstrated that the proposed method is the most efficient countermeasure compared with previous techniques. We further demonstrated that the proposed algorithms are secure against SPA, DPA, and RPA. However, the proposed algorithms could be susceptible to higher-order DPA attacks. Thus, a direction for further research would be to investigate security against higher-order DPA.

2.2.1.2.1.7. Register File Resistant to Power Analysis Attacks

We [87] propose RFRF, a register file that is resistant to power analysis attacks. By storing a flipped copy of data and adding a pre charge phase for write operations, RFRF has the same number of transitions on bit lines for all read or write operations. We also validate our method with simulations. The results show that the power trace of RFRF is independent of data values. When combined with similar solutions for other processor components, it is expected that the proposed method can strongly protect the system from power analysis attacks.

The overhead of RFRF mainly comes from the redundant register bank and additional circuitry for the pre charge phase in writes and for the data bypassing during back-to back write and read. However, the overhead is considered worthwhile as chip security is becoming a top priority in most embedded systems. Future efforts will also be spent on improving the cost-efficiency of the solution. To reduce the power overhead, the RFRF has two working modes where only security applications incur energy overhead for the security enhancement. It should also be pointed out that due to variations in process, voltages, and temperature, as well as differences in routing of the two register banks, the power consumption may not always be absolutely independent on data values.

However, this effect is considered difficult to exploit for the power analysis attacks. RFRF will also be studied for other purposes such as thwarting fault analysis attacks and improving the reliability of devices.

2.2.1.2.1.8. Sommer's SPA attack

In [58], Sommer has provided experimental evidence that the Hamming weight of secret data can be found from a single power trace of a smart card. She studied an algorithm written in the assembly language of the 8-bit processor. The algorithm consisted of a loop, where in each iteration only the data value was changed and was then moved from the accumulator to an internal register or written to output ports. She collected traces of the power consumption of the card where the sampling rate was 50 samples per card clock cycle.

2.2.1.2.1.9. Single-Exponent, Multiple-Data Attack (SEMD)

In this attack, it is assumed that the attacker can make the card exponentiate several random inputs with the secret key and with another known key. The attacker collects the power consumption traces of the exponentiations that use the secret key and computes their average. He repeats this procedure again but with the known key (111 . . . 1)2 in his attack. He then subtracts the two averaged signals. The resulting signal will show spikes in the iterations where the bits of the two keys differ. The portions of the averaged signals that are data dependent or where the bits of the exponents agree will approach 0. Messerges' countermeasure to this attack is to blind the exponent before every exponentiation.

2.2.1.2.1.10. Multiple-Exponent, Single-Data Attack (MESD)

In this attack it is assumed that the cards will exponentiate the same input, not necessarily known to the attacker, with several keys of the attacker's choice. The attacker first collects the average power trace of the exponentiation of the input with the secret key. Now, assuming that the attacker knows the first j − 1most significant or least significant, depending on the algorithm—bits of the key, he wants to attack the jth bit. He guesses that this bit is 1 and sets a new key equal to the bits that he knows concatenated with the guessed bit and arbitrary value for the remaining bits. He asks the card to exponentiate several times the constant input with that key and collects the average power trace. He repeats this step with the guessed bit reset to 0. He subtracts each of the collected averaged traces from the original one. For the correct guess, the resulting trace will approach zero through all iterations including iteration j, but for the wrong guess, the resulting trace will depict differences in iteration j.

2.2.1.3. Differential Power Analysis (DPA) Attacks.

Differential Power Analysis attacks are harder to prevent. These attacks are not visible. The keys information obtained from statistical analysis and Error-correction techniques. DPA usually consists of data collection and data-analysis stages that make extensive statistical functions for noise

filtering. It is also used for gaining additional information about the processes that the unit is performing.

In addition to large – scale power variations due to the instruction sequence, these effects correlated to the data values being manipulated. These variations tend to be smaller and are something overshadowed by measurement errors and other noise. In such cases it is still possible to break the systems using statistical functions tailored to the target algorithms. Because DPA automatically locality correlated regions in a device's power consumption. The attacker can target the implementation can be automated and little or no information.

To implement a DPA attack, an attacker first observes M encryption operation and captures power traces T[1 :: M] [1::K] containing k samples each. In addition the attacker records the cipher texts C [1::M]. Not necessary to known the plaintext. DPA analysis uses power consumption measurements and statistical methods to determine whether a key block guess k is correct [51]. Analyzing the outer DES operation first , using the resulting key to decrypt the cipher texts , and attacking the next DES sub key can be find Triple – DES keys . DPA can use known plaintext or known cipher text and find encryption or decryption keys [51].

Several improvements can be applied to the data collection and DPA analysis process to reduce the number of samples required or to circumvent countermeasures. For example it is helpful to apply correctness for the measurement variance, yielding the significance of the variations instead of their magnitude. One variant of this approach, automated template DPA, can find DES keys using fewer than 15 traces from most smart cards [51].

High-order DPA involves looking at power consumptions between several sub-operations of the encryption operation. The DPA techniques described above analyze information across a simple event between multiple cryptographic sub-operations. It mentioned that there is no known unit that vulnerable to Higher-order DPA techniques and is not vulnerable to DPA as well. In other words the precautions that are taken to prevent DPA should be ones against that work against High-order DPA as

well. According to [52] no systems are currently known that are resistant to DPA and are not resistant to high-Order DPA.

2.2.1.3.1. Countermeasures against differential power analysis attacks

2.2.1.3.1.1. Reduction of Signal Size

One approach to prevented DPA attacks is by reducing the signal sizes, such as by using constant execution path code, choosing operations that leak less information in this power consumption balancing Hamming weights and state transitions or by physically shielding the device. The signal size can not be reduce to zero when the attacker able to perform an infinite number of samples on DPA on the signal [42].

2.2.1.3.1.2. Addition of Noise

Another approach against DPA involves introducing noise into power consumption measurements. Like signal size reduction adding noise increase the number of samples required for an attack, possible to an unfeasibly large number.

2.2.1.3.1.3. DPA Resistant To Hardware Implementation of the RSA Cryptosystems

RSA [5] cryptosystem was implemented on hardware, and then modified to be resistant against Differential Power Analysis attacks by using the Randomized Table Window method. This is the first FPG realization of an algorithmic countermeasure which makes RSA resistant to power analysis attacks. Modular exponentiation is realized with Montgomery Modular Multiplication.

The Montgomery modular multiplier has been realized with Carry-Save Adders. Carry-Save representation has been used throughout the RSA encryption algorithm. The primarily implemented RSA architecture prevents the extraction of the secret key using Simple Power Analysis attacks. When comparing the protected implementation with the unprotected, it can be seen that the total time has increased by 24.2%, while the throughput has decreased by 19.5%. First hardware implementation of a RSA cryptosystem which is resistant to power analysis attacks.

Modular exponentiation is realized with Montgomery Modular Multiplication. The Montgomery modular multiplier has been realized with Carry-Save Adders. The primarily implemented RSA circuit's architecture prevents the extraction of the secret key using Simple Power Analysis attacks. In the second implementation of this work, the changes within the Randomized Table-Window Method (RT-WM) have been applied over the first implementation in order to have a DPA resistant implementation. This is the first hardware realization of RT-WM.

2.2.1.3.1.4. High-Level Simulation for Side Channel Attacks

In particular, differential power analysis (DPA) [88], [75] is very risky, because it cracks security codes by statistically processing the difference in electricity consumption and can be easily attacked. Therefore, it is important to verify DPA in the early stage of designing an algorithm. This study proposes a new simulator that can evaluate the resistance of DPA at the algorithm design level. Moreover, experiments proved the validity of the proposed simulator. Paper [82] noticed that the hamming weight of the medium value in a round of data encryption standard (DES) cryptogram reflected a bias of the transition probability due to nonlinearity of substitution-box (S-BOX). This result indicated that DPA simulation at the algorithm level could be performed using the hamming weight as a power consumption model.

However, this simulation can not distinguish between hardware architectures. Next, regarding the logic design phase, a paper [90] extracted the logic toggle information from the results of delay simulation. Then the paper simulated a CPA attack using the logic toggle information as power consumption model. Regarding studies on tamper-resistance verification of actual devices, such as field-programmable gate array (FPGA) and application-specific ICs (ASICs), papers [24[25] reported a CPA evaluation according to the advanced encryption standard (AES) embedding method by using a side-channel attack standard evaluation board-R (SASEBO-R).

A tamper-resistance simulation method using a tamper-resistance simulation method using a High-accuracy power consumption model at the algorithm level developed in the present study has not

been reported to our knowledge. This study proposed a new simulation method by which tamper resistance to an encryption device could be verified at the algorithm level. In the proposed simulation method, a sophisticated power consumption model was only introduced

into a circuit block with nonlinearity, and it is a target of side-channel attack.

The verification environment defined by programming language was used in the other parts. Using this simulation method with mixed-level, high-accuracy and high-speed simulation could be realized. Using the encryption algorithm of AES, three device configuration methods of truth table, PPRM3, and composite field methods were evaluated. In the future, we will compare the results obtained in this study with those obtained by power analysis attacks such as DPA and CPA by using ASIC. Moreover, we will investigate interconnection delays at algorithm level.

2.2.1.3.1.5. Messerges DPA Attacks

Messerges et al. [58] have mounted the following DPA-like attacks on smart cards running RSA exponentiation algorithms. Their attacks have been based on monitoring the power consumption leakage. They are mounted on left-to-right or right-to-left exponentiation algorithms that processed the key one bit at a time.

2.2.1.3.1.6. Zero-Exponent, Multiple-Data Attack (ZEMD)

This attack is the same DPA attack as explained in Section A.3. Hence, unlike the SEMD and the MESD attacks, it assumes that the attacker knows how the exponentiation algorithm is performed and can simulate it to compute intermediate points. The intermediate point computed by Messerges is the result of the multiplication in the iteration processing the guessed bit rather than the result of the squaring of the next iteration as was done by Coron. This is because his attack was on a square-and-multiply algorithm where the multiplication was performed only when the current bit of the key is 1. But Coron’s attack was on a double-and-add-always algorithm where the addition was performed regardless of the value of the current bit and its result collected if that bit was 1.

Hence, the result of the doubling of the next iteration can reveal the validity of the guess. Also Messerges used as a partitioning function the Hamming weight of some byte in the intermediate result being 8 or 0 Messerges' countermeasure to the MESD and the ZEMD attacks is to blind the input before every exponentiation and unbind it at the end

2.2.1.3.1.7. Second-Order DPA Attack

The definition of a high-order DPA attack was first presented by Kocher et al. [43] as a DPA attack that combines multiple samples from within a trace. The following is the definition according to Messerges [58]:

An nth-order DPA attack makes use of n different samples in the power Consumption signal that correspond to n different intermediate values calculated during the execution of an algorithm. The power leakage model which is proposed, and experimentally verified, by Messerges [58] assumes that the power consumption of an instruction, C (t), varies linearly with the Hamming weight H(w) of the data w processed by this instruction at time t. i.e. $C(t) = \in H(w) + l$.

Where $\in$ and l are some hardware-dependent constants.

Messerges presented a second-order DPA attack on a typical algorithm of a public key cryptosystem. He chose two instructions of the algorithm that are not necessarily consecutive to monitor their power consumption every time the algorithm is executed. The first instruction processes random data and the second one processes a part of the secret key XOR ed with input data XOR ed with the random data of the first instruction. To reveal bit j of the secret key, the attacker sets bit j of the input data to 0 and the other ones to random values and gathers the power consumption traces of both instructions with different input data. The attacker repeats the same procedure but with setting bit j of the input data to 1. By averaging the difference in power consumption between these two instructions for the 0 and 1 set of traces, the attacker could reveal the value of the key bits.

2.3. PREVENTING SIDE CHANNEL ATTACKS.

2.3.1. General Data-Independent Calculations

All operations are performed by the module shall be data-independent in their time consumption. In other words, the time that operations take totally must be independent of input data or key data. The different sub operations are performed according to input or key bits, and these sub operations should take same number of clock cycles. The general feature of making the time needed for operation execution fixed for every piece of data prevents all timing attacks. This is because these attacks are based on variations in the computation time according to input and key bits.

2.3.2. Blinding

Techniques used for blinding signatures can be adapted to prevent attacks from knowing the input to the modular exponentiation function [22].

2.3.3. Avoiding Conditional Branching and Secret Intermediates

According to [4], avoiding procedures that can use secret intermediates of keys for conditional branching operations will mask many SPA characteristics. Software implementations of critical code shall not contain branching statement computations should be performed using functions that utilize elementary operations such as AND, OR and XOR and not using any branching and conditional of portions of the code.

2.3.4. Licensing Modified Algorithms

The most effective general solution is to design and implement cryptosystems with the assumption that information leak. A few companies develop approaches for securing existing cryptographic algorithms to make systems remain secure for ever though the underlying circuits may leak information.

2.4. POWER CONSUMPTION OF CRYPTOGRAPHIC DEVICES

2.4.1. Power Models

A requirement of a deferential power analysis attack which utilizes correlation as it's primary method of determining the key is that there be a model which can approximate the power consumption of a circuit during the encryption process. There are several different methods for constructing this

power model. An accurate method for describing the power consumption of a device is to simulate the device in a software environment designed to show power consumption.

If the target cryptographic device has an architecture that is known, then such a simulation may provide a precise prediction of the power consumption of the circuit during encryption. In instances where the target device's architecture is not entirely known or it is infeasible to simulate it, a more general power model must be used. The power models currently used in these instances are the Hamming weight and the Hamming distance models.

2.4.2. Hamming Weight Power Model

The Hamming weight model is the most basic power consumption model. It is most applicable to approximate the power consumption of a circuit utilizing a data or address bus, and relies on the basic premise that a bus will consume an amount of power that is proportional to the numbers of bits that are switched on within that bus. If no bits are switched on, the bus will consume very little power compared to a data bus with all bits switched on.

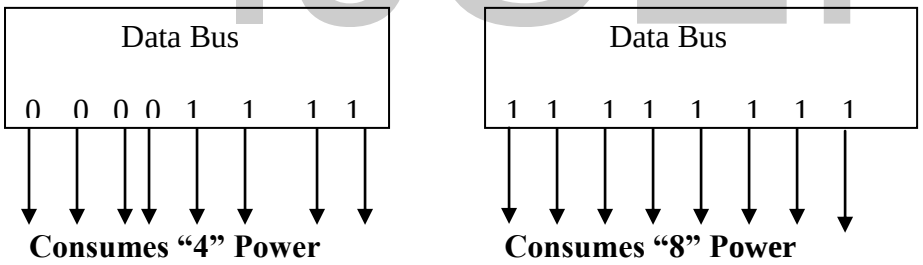


Figure 2.3.An example showing the Hamming weight model

. The Hamming weight of the value on the data bus is taken to be proportional to the power consumption of the bus. This model can be used when very little information is known about the circuit that the attack is being performed on. It also carries an advantage over the Hamming distance model in that it does not require any information about the data on the bus during the encryption routine. However, the model only weakly describes power consumption within a circuit. While deferential power analysis attacks are possible using the HW model, they are not as efficient compared

to the Hamming distance model. In general, whenever some information is known about the target circuit's implementation the HD model should be used [57].

2.4.3. Hamming Distance Power Model

The Hamming distance model is an extension of the Hamming weight model which uses changes in logic values in a certain time interval to determine power usage. The change can occur in many different circuit components such as a data or address bus, register, memory, or some other component. Using this model, you can determine the approximate power consumption of a circuit as being proportional to the number of 0,1 and 1,0 transitions made within the circuit. The number of bit transitions is simply the Hamming weight of the exclusive or of the two values. For example, the Hamming distance between two register values (R0 and R1) is given as:

HD (R0; R1) = HW (R0 ` R1)

Several basic assumptions are made when this power model is used. It is assumed that bits which do not change (0,0, 1,1) do not contribute to the power consumption of a circuit. It is also assumed that a 0, 1 and 1, 0 transition consume an equal amount of power. In most circuits this may be found to not be the case. If more information is known about the specific power consumption of the device, minor enhancements can be made to the model by weighting transitions differently. An example showing how a register updating its value on a clock edge is shown in Figure.2.4

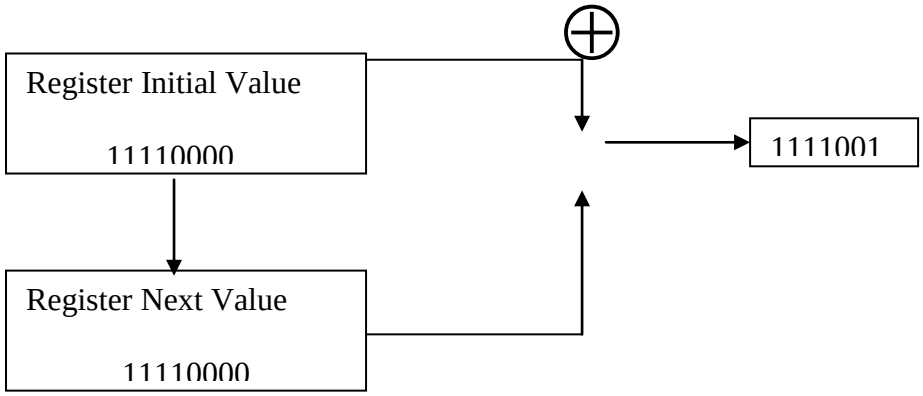


Figure 2.4. An example showing the Hamming Distance model.

As the register updates from one state to the next, it consumes a certain amount of power. In this example, the Hamming distance shows that the power consumed when the register updates is proportional to 6. Regardless of the weaknesses, the Hamming distance model provides a very convenient method for determining the expected power consumption of a circuit during a certain time frame. In any case where a change in data can be observed, the Hamming distance model should be used over the Hamming weight model.

2.5. Scalar Multiplication

The operation consisting in calculating the multiple of a point $k \cdot P = P + P + \cdots + P$ (k times) is called scalar multiplication and the integer k is thus referred to as the scalar.

Scalar multiplication is used in ECDSA signature [46] and ECDH key agreement [55] protocols. Implementing such protocols on embedded devices requires particular care from both the efficiency and the security points of view. Indeed scalar multiplication turns out to be the most time consuming part of the aforementioned protocols, and since it uses secret values as scalars, side-channel analysis endangers the security of those protocols

2.5.1. Atomicity Improvement for ECSM

We [37] address the problem of protecting elliptic curve scalar multiplication implementations against side-channel analysis by using the atomicity principle. First of all we reexamine classical assumptions made by scalar multiplication designers and we point out that some of them are not relevant in the context of embedded devices. They describe the state-of the-art of atomic scalar multiplication and propose an atomic pattern improvement method. Compared to the most efficient atomic scalar multiplication published so far, our technique shows an average improvement of up to 10.6%.

We [37] propose a new atomic pattern for scalar multiplication on elliptic curves over Fp and detail our method for atomic pattern improvement. Firstly we maximize the use of squarings to replace

multiplications since the latter are slower. Secondly we minimize the use of field additions and negations since they induce a non-negligible penalty.

In particular, they point out that the classical hypothesis taken by scalar multiplication designers to neglect the cost of additions/subtractions in Fp is not valid when focusing on embedded devices such as smart cards. In this context our method provides an average 18.3% improvement for the right-to-left mixed scalar multiplication from [44] protected with the atomic pattern from [17]. They recommend that algorithm designers, addressing the scope of embedded devices, take into account additions and subtractions cost when these operations are heavily used in an algorithm. Moreover the issue of designing efficient atomic patterns should be considered when proposing non regular sensitive algorithms.

2.5.2. Securing ECSM against Side-Channel Attacks

A common disadvantage is that each of them requires specially selected elliptic curves: All curves suitable for [55] group order divisible by 4; curves suitable for [46] group order divisible by 3; and curves suitable for [68] (curves with Montgomery form) again have group order divisible by 4. None of these methods is applicable to the NIST and SECG recommended curves given in [67] and [15], whose use is often encouraged in order to ease interoperability.

We [64] have presented a method for elliptic curve point multiplication that can be shown to provide security against side-channel attacks. The algorithm uses a special signed-digit encoding to ensure that point doublings and point additions occur in a uniform pattern. No dummy additions are required; implementing the method using randomized projective coordinates and storing precomputed points in extended point representation limits information leakage to a minimum.

2.6. Conclusion

We have presented a background on side-channel attacks along with the different methods used to countermeasures against side-channel attacks. We provide the basic nature of the power analysis

attacks and power consumption of cryptographic devices in side-channel analysis. SPA attacks (excluding Sommer's attack) can be prevented by making the ECSM execution uniform over all iterations, preferably with no dummy operations. The first order DPA attacks are based on the fact that intermediate points computed by the algorithm can be guessed by the attacker. Hence, to prevent them these intermediate points should be randomized. To resist those attacks, the key value should be randomized before the ECSM execution.

CHAPTER .3

GENERALIZATION OF N. M. EDIEB'S POWER ANALYSIS ATTACKS ON ELLIPTIC CURVE CRYPTOSYSTEMS

3.1. Introduction

We present a different method to compute the scalar multiplication (ECSM), which is the core operation of ECCs, and the various costs associated with them. We analyze and study Major and Minor Collisions and provide an analytic result for their probability of occurrence as well as effect of the fixed sequence window method.

3.1.1. Elliptic Curve Cryptosystems

Let K be a finite field and E be an elliptic curve (EC) over K defined by the following Weierstrass equation

$$E: y^2+a_1xy+a_3y= x^3+ a_2x^2+a_4x + a_6 \quad (3.1)$$

Where $a_i \in K$ and $\Delta \neq 0$, where Δ is the discriminant of E

Let L be an extension field of K. Then E (L) denotes the set of L-rational points (x, y) on E, where (x, y) $\in$ L × L and satisfy (3.1), together with the point at infinity ∂. The addition of two points on the curve is performed using a chord-and-tangent rule the set E (L) with the addition operation form an abelian group, where ∂ is the identity. We denote the field inversion by I, the multiplication by M, the squaring by S. The point addition is denoted by A. When the two operands of the addition are the same point, the operation is referred to as point doubling and is denoted by D.

3.1.2. Elliptic curves over prime fields

If $K = F_p$, where p > 3 is a prime, (3.1) can be simplified to

$$E: y^2 = x^3 + ax +b \quad (3.2)$$

Where a and b $\in F_p$, The discriminant of this curve is $\Delta = -16(4a^3+27b^2)$. The negative of a point P = (x, y) is −P = (x, −y) such that P + (−P) = ∂.This simplification is generally applicable when the characteristic of K is not 2 or 3.

The affine coordinate (A) representation of a point P = (x, y) can be replaced by projective coordinates representations in order to render the point addition and doubling operations less costly in terms of field operations. The following representations are the best known.

- Standard (homogeneous) projective coordinates (P); the projective point (X Y: Z), Z ≠ 0, corresponds to the affine point (X/Z, Y/Z), ∂ corresponds to (0 : 1 : 0) and the negative of (X : Y : Z) is (X : −Y : Z).
- Jacobian projective coordinates (J); the projective point (X : Y : Z), $Z_6 = 0$, corresponds to the affine point (X/Z_2, Y/Z_3), O corresponds to (0: 1 : 0) and the negative of (X: Y: Z) is (X: −Y : Z).
- Chudnovsky coordinates (C); the Jacobian point (X: Y: Z) is represented as(X: Y: Z: Z^2 : Z^3).

3.1.3 Elliptic curves over binary fields: If K = F_{2^m} , (3.1) can be simplified to

$$E: y^2 = x^3 + ax + b, \qquad (3.3)$$

Where a and b $\in F_{2^m}$. The discriminant of this curve is Δ = b and the negative of a point P = (x, y) is −P = (x, x + y). Such a curve is known as non-supersingular.

3.1.4. Elliptic Curve Scalar Multiplication (ECSM)

Scalar multiplication in the group of points of an elliptic curve is analogous to exponentiation in the multiplicative group of integers modulo a fixed integer. Thus, it is the fundamental operation in EC-based cryptographic systems. The scalar multiplication, denoted kP, is the result of adding the

point P to itself k times, where k is a positive integer, that is kP = P + P + · · · + P | {z} k copies and −kP = k (−P). u is said to be the order of P if u is the smallest integer.

Let $(k_{n-1}, k_{n-2}, \ldots, k_1, k_0)_2$ be the binary representation of k, i.e., $k_i \in \{0, 1\}$ for $0 \leq i < n -$ 1. Thus,

$$kP = \left(\sum_{i=0}^{n-1} k_i 2^i\right) P = 2(2(\cdots 2(2(k_{n-1} P) + k_{n-2} P) + \cdots) + k_1 P) + k_0 P$$

$$= (k_{n-1} 2^{n-1} P) + \cdots (k_1 2P) + (k_0 P) \qquad (3.4)$$

Hence, kP can be computed using the straightforward double-and-add approach in n iterations. These algorithms are analogous to the square-and-multiply algorithms employed in exponentiation-based cryptosystems.

Algorithm: 3.1.4.1. Left-to-Right Double-and- Add Algorithm

Input: $k = (k_{n-1} \ldots k_0)\ 2$ and $P \in E(F_q)$.

Output: kP.

1. Q ← ∂

2. for i from n − 1 down to 0 do

2.1 Q ← 2Q.

2.2 if $(k_i = 1)$ then

2.3 Q ← Q + P.

3. Return (Q).

Algorithm: 3.1.4.2. Right-to-Left Double-and- Add Algorithm

Input: $k = (k_{n-1}, \ldots, k_0)\ 2$ and $P \in E(F_q)$.

Output: KP.

1. Q← ∂; R← P.

2. for i from 0 to n − 1 do

2.1 if (k_i = 1) then Q ← Q + R.

2.2 R ← 2R.

3. Return (Q).

The expected number of point addition (A) and point doubling (D) operations performed in the binary algorithm (left-to-right or right-to-left) is (n − 1) D + $\frac{n}{2}$ A.

3.2. Window Methods

This method is sometimes referred to as m-ary method. What is common among them is that, if the window width is w, some multiples of the point P up to (2^w − 1) P are precomputed and stored and k is processed w bits at a time. k is recoded to the radix2^w. k can be recoded in a way so that the average density of the nonzero digits in the recoding is 1/(w + ξ), where $0 \le \xi \le 2$ depends on the algorithm. Let the number of precomputed points be t, in the precomputation stage, each point requires either a doubling or an addition to be computed also depending on the algorithm.

This ECSM method is suitable for unknown or fixed point P. The cost is Storage: t points, where $2^{w-2} \le t \le 2^{w-1}$ depending on the algorithm.

Precomputation: t point operations (A or D).

Expected running time: (n − 1) D + n$\frac{n}{w+\xi}$ A, where $0 \le \xi \le 2$ depending on the algorithm.

3.2.1. Simultaneous Multiple Point Multiplication

This method is used to compute kP +lS where P may be a known point. This algorithm was referred to as Shamir's trick in [20]. If k and l are n-bit integers, then their binary representations are written in a 2 × n matrix called the exponent array. Given width w, the values iP +jS are calculated for $0 \le i, j < 2^w$. Now the algorithm performs d = $\lceil n/w \rceil$ iterations. In every iteration, the accumulator

point is doubled w times and the current 2 × w window over the exponent array determines the precomputed point that is to be added to the accumulator.

Algorithm: 3.2.1.1. Simultaneous Multiple Point Multiplication (Shamir-Strauss method)

Input: Window width w, d = $\lceil n/w \rceil$, k = $(K_{d-1}, \ldots, K_1, K_0)_{2^w}$ l= $(L_{d-1}, \ldots, L_1, L_0)_{2^w}$, and P, S $\in$ E(F_q).

Output: kP + lS.

1. Precomputation. Compute iP + jS for all i, j $\in [0, 2^w - 1]$.

2. Q $\leftarrow K_{d-1}$P + L_{d-1}S.

3. for i from d − 2 down to 0 do

3.1 Q $\leftarrow$ 2^wQ.

3.2 Q $\leftarrow$ Q + (K_i P + L_i S).

4. Return (Q).

Storage: $2^{2w} - 1$ points. For w = 1, 3 points. For w = 2, 15 points.

Precomputation: $(2^{2(w-1)} - 2^{w-1})$ D + $(3 \cdot 2^{2(w-1)} - 2^{w-1} - 1)$ A.

For w = 1, 1 A.

For w = 2, 1 D + 11 A.

Expected running time: (d − 1)w D + $\frac{(2^{2w}-1}{2^{2w}}$d − 1) A.

For w = 1, (n − 1) D + ($\frac{3}{4}$n − 1) A.

For w = 2, (n − 1) D + ($\frac{15}{32}$n -1) A.

Using sliding windows can save about 1/4 of the precomputed points and decrease the number of additions to, which is about 9% saving for w $\in$ {2, 3}.

3.2.2. Interleaving Method

This method is also a multiple point multiplication method, that is we want to compute $\sum k^j P_j$ for points P_j and integer's k^j. In the comb and simultaneous multiplication methods, each of the precomputed values is a sum of the multiples of the input points. In the interleaving method, each precomputed value is simply a multiple of one of the input points.

Algorithm 3.2.2.1. Interleaving Method

Input: width w, $d = \lceil n/w \rceil$, $k = (K_{d-1}, \ldots, K_1, K_0)_{2^w}$ $l = (L_{d-1}, \ldots, L_1, L_0)_{2^w}$, and

$P, S \in E(F_q)$.

Output: kP + lS.

1. Precomputation. Compute iP and iS for all $i \in [0, 2^w - 1]$.

2. $Q \leftarrow K_{d-1} P$.

3. $Q \leftarrow Q L_{d-1} S$.

4. for i from d − 2 down to 0 do

   4.1 $Q \leftarrow 2^w Q$.

   4.2 $Q \leftarrow Q + K_i P$.

   4.3 $Q \leftarrow Q + L_i S$.

5. Return (Q).

Storage: $2^{w+1} - 2$ points.

Precomputation: 2(w-1) D + $2(2^w - w - 1)$ A.

Expected running time: $w(d-1)$ D + $(2d-1)$. $\frac{(2^{2w}-1)}{2^{2w}}$ A

In general, if different basis and/or representations are used for k and l, we have

Storage: 2t points, where $2^{w-2} \leq t \leq 2^{w-1}$.

Precomputation: 2t point operations (A or D).

Expected running time: (n − 1) D + 2 $\frac{n}{w+i}$ A, where $1 \leq i \leq 2$ depending on the algorithm

3.3. Power and Electromagnetic Analysis Attacks on ECCs

The attacks are those that monitor the power consumption and / or the electromagnetic emanations of a device, e.g., a smart card or a handheld device, and can infer important information about the instructions being executed or the operands being manipulated at a specific instant of interest.

These attacks are broadly divided into two categories; simple and differential analysis attacks. We will refer to the former category as SPA attacks and the latter as DPA attacks.SPA and DPA are the acronyms for simple power analysis and differential power analysis.

Power analysis attacks use the fact that the instantaneous power consumption of a hardware device is related to the instantaneous computed instructions and the manipulated data. The attacker could measure the power consumption during the execution of a cryptographic algorithm, store the waveform using a digital oscilloscope and process the information to learn the secret key. Kocher et al., in [43], first introduced this type of attack on smart cards performing the DES operation. Then Messerges et al. [47] augmented Kocher's work by providing further analysis and detailed examples of actual attacks they mounted on smart cards.

In general, SPA attacks are those based on retrieving valuable information about the secret key from a single leaked information power consumption or electromagnetic emanation trace. On the other hand, DPA attacks generally include all attacks that require more than one such trace along with some statistical analysis tools to extract the implicit information from those traces.

3.3.1 SPA Attack on ECCs and its Countermeasures

Coron [18] has transferred the power analysis attacks to ECCs and has shown that an unaware implementation of EC operations can easily be exploited to mount an SPA attack. Monitoring

of the power consumption enables us to visually identify large features of an ECC implementation such as the main loop in Algorithms 3.1.4.1 and 3.1.4.2.

Window methods process the key on a digit (window) level. The basic version of this method, that is where $\xi = 0$ in Section 3.1, is inherently uniform since in most iterations, w D operations are followed by 1 A, except for possibly when the digit is 0. Therefore, fixed-sequence window methods were proposed [50], [57] and [65] in order to recode the digits of the key such that the digit set does not include 0.

3.3.2 DPA Attack on ECCs and its Countermeasures

The relation between the instructions executed by a cryptographic algorithm and the key bits is not directly observable from the power signal; an attacker can apply differential power analysis (DPA). DPA attacks are in general more threatening and more powerful than SPA attacks because the attacker does not need to know as many details about how the algorithm was implemented. The technique also gains strength by using statistical analysis and digital signal processing techniques on a large number of power consumption signals to reduce noise and to amplify the differential signal. The latter is indicated by a peak, if any, in the plot of the processed data. This peak appears only if the attacker's guess of a bit or a digit of the secret key is correct. The attacker's goal is to retrieve partial or full information about a long-term key that is employed in several ECSM executions.

As for the SPA attack, Kocher et al. were the first to introduce the DPA attack on a smart card implementation of DES [43]. Techniques to strengthen the attack and a theoretical basis for it were presented by Messerges et al. in [47]. Coron applied the DPA attack to ECCs [18]. It is based on randomly splitting the key into two parts such that each part is different in every ECSM execution. An additive splitting using subtraction is attributed to [17]. It is based on computing

$$kP=(k-r)P + rP, \qquad (3.5)$$

The authors mention that the idea of splitting the data was abstracted in [12]. Where r is a n-bit random integer, that is, of the same bit length as k. alternatively, Ciet and Joy [16] suggest the following additive splitting using division, that is, k is written as

$$k = \lfloor k/r \rfloor + (k \bmod r). \quad (3.6)$$

Hence, if we let $k_1 = (k \bmod r)$, $k_2 2 = \lfloor k/r \rfloor$ and S = rP, we can compute

$$kP = k_1 p + k_2 P \quad (3.7)$$

Where the bit length of r is n/2. They also suggest that (3.6) should be evaluated with Shamir-Strauss method as in Algorithm 3.2.1. However, they did not mention whether the same algorithm should be used to evaluate (3.5).

The following multiplicative splitting was proposed by Trichina and Bellezza [15] where r is a random integer invertible modulo u, the order of P. The scalar multiplication kP is then evaluated

$$P = [kr^{-1} (modu)](rP) \quad (3.8)$$

To evaluate (3.8), two scalar multiplications are needed; first R = rP is computed, and then $kr^{-1}R$ is computed.

3.4. Key Splitting Methods

3.4.1 Additive Splitting Using Subtraction (scheme I)

This approach was suggested by [17] and revisited by [13] as follows. In order to compute the point kP, the n-bit key k is written as $k = k_1 + k_2$, such that $k_1 = k - r$ and $k_2 = r$, where r is a random integer of length n bits. Then kP is computed as

$$kP = k_1 P + k_2 P. \quad (3.9)$$

It is important to note that each of the terms of (3.1) should be evaluated separately and their results combined at the end using point addition. This observation is based on the following lemma. Let $k_{b \to a}$ denote $\lfloor k(\bmod 2^{b+1}) 2^a \rfloor$ or, simply, the bits of k from bit position b down to bit position a, with $b \geq a$.

Lemma 3.4.1 Let splitting scheme I can be evaluated using Algorithm 2.3 with w = 1 (d = n). Then, at the end of some iteration j, $0 < j \leq n - 1$, there are only two possible values for Q, those are $[k_{n-I-j}]P$ or $[k_{n-I-j} - 1]P$.

Proof. Algorithm 3.2.1 and similarly Algorithm 3.2.2 computes the required point by scanning $k_1 = (k_{1\,n-1}, \ldots, k_{1\,0})_2$ and $k_2 = (k_{2\,n-1}, \ldots, k_{2\,0})_2$ from the most significant end down to the least significant end. Hence, at the end of iteration j, the accumulator Q contains the value

$$Q = k_{1n-I-j}P + {}_{2n-I-j}P \quad (3.10)$$

$= [k_{1n-I-j} + k_{2n-I-j}]P$. We can write k, k_1 and k_2 as

$$k = k_{n-I-j^j}2^j + {}_{j-1-0} \quad (3.11)$$

$$k_i = k_{i\ n-I-j^j}2^j + k_{i\ j-1-0} \quad (3.12)$$

Since $k = k_1 + k_2$ we have

$$k_{1\ j-I-0} + k_{2\ j-I-0}j = k_{j-I-0} + b\,2^j \text{ where } b \in \{0, 1\} \quad (3.13)$$

and $k_{1\ n-I\to j} + k_{2\ n-I\to j} = k_{n-I\to j}$-b

The attack would proceed in the same way, whether the algorithm processes a single bit or a digit per iteration, though it would be more involved in the latter case depending on the digit size. The attacker can double the number of traces gathered and compute the necessary intermediate points as if there was no countermeasure in place.

Hence each term should be computed separately using a SPA-resistant algorithm fixed-sequence window method. The key splitting process, in the subtraction processing key every time, then the attacker can obtain less noisy information about the key words such as their hamming weights by averaging the side-channel trace.

Moreover, if it is difficult for the attacker to locate the instances where the key is manipulated, then by correlating different traces, he can detect where the same data is processed. Therefore, it is

desirable to use a previously split version of the key to generate the new one. Hence k_1 and k_2 can be refreshed as $k_1 \pm rt$, $k_2 \mp rt$, before the t-th execution of the ECSM, where the addition /subtraction is modulo the group order of the points on the elliptic curve and r_t is an n-bit random integer.

3.4.2 Additive Splitting Using Division (scheme II)

As an alternative to the previous splitting in [16] suggest that a random divisor r be chosen and the key k written as k = g * r + h, where g = $\lfloor k/r \rfloor$ and h = k mod r. Let S = rP, then kP can be computed as

$$kP = gS + hP. \quad (3.14)$$

We choose the bit length of r to be l = $\lceil n/2 \rceil$. That is, r is chosen uniformly at random from the range [2^{l-1}, $2^l - 1$]. Hence, the bit length of g is at most $\lfloor n/2 \rfloor + 1 \leq l + 1$ and at least l and that of h is at most l [33].

An ECSM is first performed to compute the point S, where the scalar is of size half that of k. Then, unlike splitting scheme I, a multiple point multiplication method can be safely used. In the following, we will justify this assertion.

Let the representations of k, g and h to the base 2^w, for some w ≥ 1, be

$(K_{2z-1}, \ldots, K_1 1, K_0)2^w$, $(G_z, \ldots, G1, G0)2^w$ and $(H_{z-1}, \ldots, H_1, H_0)$, respectively, where z = $\lceil l/w \rceil$ that is l ≤ zw _ l +w ≤ 1. If l < zw, then $G_z = 0$, otherwise, if l = zw, then Gz ≤1. As before, let $K_{b \to a}$ denote $\lfloor k \bmod 2^{bw+1}) / 2^{awc} \rfloor$ or, simply, the w-bit digits of k from digit position b down to digit position a, with b ≥ a. Let (3.7) be evaluated using Algorithm 3.2.1 or Algorithm 3.2.2, replacing d by z + 1 in these algorithms and setting Hz = 0. Then at the end of some iteration j, 1 < j _ z, the accumulator Q contains the value

$$Q = G_{z \to j} S + H_{z \to j} P,$$

$$= (G_{z \to j} * r + H_{z \to j})P. \quad (3.15)$$

Let $k^j = G_{z\to j} * r + H_{z\to j}$, which is of length 2z − j w-bit digits1. In general— exceptions follow—, $k^j \neq K_{2z-1\to j}$. This is true since $K_{2z-1\to j} = G_{z\to j} * r + h_j$, where h_j is the l-bit remainder of the division of $K_{2z-1\to j}$ by r. Since $K_{2z-1\to j} \neq k$, then from the division theorem, the pair $(G_{z\to j}, h_j)$ is not equal to (g, h), hence, in general $h_j \neq H_{z\to j}$.

3.4.2.1. Major Collisions

A major collision is defined as the occurrence of $k^j = K_{2z-1\to j}$ at some iteration $j \in [1, z-1]$. The intermediate point computed at this value of k^j is the same value that would be computed when no countermeasure is in place. The condition of this collision is provided by the following lemma.

Lemma 3.4.2.1 For some $j \in [1, z-1]$, $k^j = K_{2z-1\to j}$ 2z−1 iff $G_{j-1\to 0} = 0$.

Proof: We have

$$k = g * r + h, = (G_{z\to j} * 2^{jw} + G_{j-1\to 0}) * r + (Hz_{z\to j} * 2^{jw} + H_{j-1\to 0})$$

$$= k^j * 2^{jw} + G_{j-1\to 0} * r + H_{j-1\to 0} \quad (3.16)$$

But $k = K_{2z-1\to j} * 2^{jw} + K_{z-1\to 0}$. Hence, if $Gj_{j-1\to 0} = 0$,

We have $k^j = K_{2z-1\to 0}$ and

$K_{j-1\to 0} = H_{j-1\to 0}$. On the other hand, if $k^j = K_{2z-1\to j}$, then b $(!G_{j-1\to 0} * r)/2^{jw} = 0$.

However, $r \geq 2^{l-1}$ that is, $r \geq 2(^{z-1)w}$. Hence, $G_{j-1\to 0} = 0^2$.

The probability of the occurrence of this collision is around 2^{-jw}. It increases with the iterations of a multiple-point multiplication in ECSM algorithm. It is negligible in the first iterations that are critical for the attacker in a DPA attack .Moreover; these collisions can be avoided when evaluating (3.7) for all j as follows. After performing the division of k by r, the quotient g is inspected. If the least significant w bits are found to be 0, another r is chosen. Note that this incurs a negligible reduction in the choice space of r from 2^{l-1} to approximately 2^{l-w-1}.

Another way to avoid these collisions is to make the quotient g always odd. That is if g is even, it is decremented by one and h is updated by adding r to it. This may increase the bit length of h to l + 1.

3.4.2.2. Minor Collisions

A minor collision occurs when at some iteration j $\in$ [1, z], for two values of r: r_1 and r_2, such that $r_1 \neq r_2$, we have $k_1^j = k_2^j j \neq K_{2z-1 \rightarrow j}$. The conditions favoring these collisions are not straightforward to analyze. Some of them occur when h1 = h2, but also many of them occur with $g_1 \neq g_2$ and $h_1 \neq h_2$. Also in some cases, collisions occur when gcd(r_1, r_2) $\neq$ 1, where gcd is the greatest common divisor.

In the following we refer to ϕ as a t-time collision value, if at iteration j, $k^j = \phi$ for t different values of r. We have conducted some experiments to study the probability of happening of these collisions for n = 40 and 50 when divided by all divisors of length 20 and 25 bits, respectively, with window width w = 4. We found that for different values of j, after excluding the values of g with w least significant bits, about 63% of the values of k^j on average were collision-free. About 25.6% were two-times collision values. The maximum number of collisions t for some value varied with the iteration; it was higher towards the middle iterations than the first and last iterations. For example, in the middle iterations, some 40-bit integers exhibited k_j values with up to 132-times collision and up to 1735-times for 50-bit integers. The density of values that have the higher number of collisions is usually 1 or 2. On the other hand, after the first iteration, the maximum number of collisions we obtained was 12 for 40-bit integers and 23 for 50-bit integers.

3.5. Proposed System

3.5.1. Major Collisions

A major collision is defined as the occurrence of $k^j = k_{2z-1} \rightarrow j$ at some iteration $j \in [1, z-1]$. The intermediate point computed of this value of k^j is the same value that would be computed when no counter measure is in place.

Lemma 3.5.1.: For some $j \in [1, z-1]$, $k^j = k_{2z-1} \rightarrow j$ iff $G_{j-1\rightarrow 0} = 0$

Proof: We know that $K = g * r + h$

$$g = G_{z\rightarrow j} * 2^{jw} + G_{j-1\rightarrow 0}$$

$$h = H_{z\rightarrow j} * 2^{jw} + H_{j-1\rightarrow 0}$$

Now $K = (G_{z\rightarrow j} * 2^{jw} + G_{j-1\rightarrow 0}) * r + (H_{z\rightarrow j} * 2^{jw} + H_{j-1\rightarrow 0})$

$$= (G_{z\rightarrow j} * 2^{jw}) * r + G_{j-1\rightarrow 0} * r + H_{z\rightarrow j} * 2^{jw} + H_{j-1\rightarrow 0}$$

$$K = (G_{z\rightarrow j} * r + H_{z\rightarrow j}) * 2^{jw} + G_{j-1\rightarrow 0} * r + H_{j-1\rightarrow 0}$$

Let $k^j = G_{z\rightarrow j} * r + H_{z\rightarrow j}$

$$K = k^j * 2^{jw} + G_{j-1\rightarrow 0} * r + H_{j-1\rightarrow 0} \quad (3.17)$$

Case (i):

Let us assume that $G_{j-1\rightarrow 0} = 0$

$$(1) \Rightarrow K = k^j * 2^{jw} + H_{j-1\rightarrow 0} \quad (3.18)$$

But we know that $k^j = k_{2z-1} \rightarrow j * 2^{jw} + k_{j-1\rightarrow 0}$ (3.19)

From (3.18) & (3.19) $k^j = k_{2z-1} \rightarrow j$ and

$$H_{j-1\to 0} = k_{j-1\to 0}$$

Case (ii):

Let us assume that $k^{j} = k_{2z-1} \to j$

$$1) \Rightarrow K = k_{2z-1} \to j * 2^{jw} + G_{j-1\to 0} * r + H_{j-1\to 0} \quad (3.20)$$

But we know that $K = k_{2z-1} \to j * 2^{jw} + k_{j-1\to 0}$ (3.21)

From (3.20) & (3.21) $G_{j-1\to 0} * r + H_{j-1\to 0} = k_{j-1\to 0}$

$$\frac{G_{j-1\to 0} * r}{2^{jw}} + \frac{H_{j-1\to 0}}{2^{jw}} = \frac{k_{j-1\to 0}}{2^{jw}}$$

$$\left\lfloor \frac{G_{j-1\to 0} * r}{2^{jw}} \right\rfloor + \left\lfloor \frac{H_{j-1\to 0}}{2^{jw}} \right\rfloor = \left\lfloor \frac{k_{j-1\to 0}}{2^{jw}} \right\rfloor$$

$$\left\lfloor \frac{G_{j-1\to 0} * r}{2^{jw}} \right\rfloor + 0 = 0$$

$$\therefore \quad \left\lfloor \frac{G_{j-1\to 0} * r}{2^{jw}} \right\rfloor = 0$$

$r \geq 2^{l-1} \quad \Rightarrow \quad r \geq 2^{(z-1)w}$

$z\text{-}1 = \left\lceil \frac{l}{w} \right\rceil \qquad \frac{r}{2^{jw}} \geq \frac{2^{(z-1)w}}{2^{jw}}$

$z\text{-}1 \leq \frac{l-1}{w} \qquad \frac{r}{2^{jw}} \not\longrightarrow 0$

$w(z\text{-}1) \leq l\text{-}1 \quad \therefore G_{j-1\to 0} = 0$

The Probability of major collisions $= \frac{2^{(z-1)w} / 2^{jw}}{2^{(z-1)w}} = \frac{1}{2^{jw}} = 2^{-jw}$

3.5.1. 1.Mathematical Proof of the Major Collisions

1. Let n = 10, z=11.
2. [1, z-1] = [1, 10].

3. j € [1, z-1] $\Longrightarrow$ j = 6.

4. z-1 = 10 $\Longrightarrow$ 10_2 = 1010. $\Longrightarrow$ Window length w = 4.

5. we know that $z = \left\lceil \frac{l}{w} \right\rceil$

6. $11 = \left\lceil \frac{l}{4} \right\rceil$

7. $40 < l \leq 44$

8. l = 41,42,43,44.

9. Now $z = \left\lceil \frac{l}{w} \right\rceil$

10. $z\text{-}1 \leq l\text{-}1/w$

11. $w(z-1) \leq l\text{-}1$

12. $w(z-1) + 1 \leq l$ since w = 4, z = 11, l = 42

13. $4(11\text{-}1) + 1 \leq l$

14. $41 \leq 42$

15. $r \geq 2^{l-1}$ & $w(z\text{-}1) \leq l\text{-}1$

16. $r \geq 2^{l-1}$ & $l\text{-}1 \geq w(z\text{-}1)$

17. $r \geq 2^{l-1}$ & $2^{l-1} \geq 2^{w(z-1)}$

$\therefore r \geq 2^{w(z-1)} \Longrightarrow r \geq 2^{4(11-1)}$

$r \geq 2^{40} \therefore r \in [40, \infty]$

18. The probability of Major collisions = $\frac{1}{2^{jw}} = 1/2^{(6)(4)} = 1/2^{24}$

$= 2^{-24} = 0.000000059605$

$= 0.59605 \times 10^{-7}$

3.5.1.2. Implementation of Major Collisions

```
import java.util.Scanner;
import java.util.Vector;
public class major {
   public static void main (String[] args) {
      Scanner s=new Scanner(System.in);
            System.out.print("Enter a Prime Integer:");
            int z=s.nextInt();
      int n=z-1;
      System.out.print("Enter a Value in btw., (1-"+n+"):");
      int j=s.nextInt();
      int w=Integer.toBinaryString(n).length();
      int l1=z*w;
      int l2=n*w;
      int l=0;
   Vector<Integer> vL=new Vector<Integer>();
      for(int i=l2;i<=l1;i++)
      {
         if(i>=z && i<=n)
      { vL.add(i);}
      }
  //  Vector vR=new Vector();
  //  for(int i=0;i<vL.size();i++)
  //  {
    double r=Math.pow(2,n*w);
```

```
//   }
     double prob=1/(Math.pow(2,j*w));
     System.out.println("W="+w);
     System.out.println("L1="+l1);
     System.out.println("L2="+l2);
     System.out.println("R="+r);
    // System.out.println("R="+Math.pow(2,40));
     System.out.println("Probability of Major Colisions ="+prob);
      System.out.println("Probability of Major Colisions ="+Math.pow(2,-24));
     }
  }
```



```
11
Enter a Value in btw., (1-10):
7
W=4
L1=44
L2=40
R=1.099511627776E12
Probability of Major Collisions =3.725290298461914E-9
Probability of Major Collisions =5.9604644775390625E-8
BUILD SUCCESSFUL (total time: 15 seconds)
```

3.5.2. Minor Collisions

A Minor collision occurs when at some iteration j $\in$ [1, z] for two values of r: r_1 and r_2, such that $r_1 \neq r_2$ we have $k_1^j = k_2^j \neq k_{2z-1} \rightarrow j$

Lemma 3 5.2.: Probability of the occurrence of the minor collision is around $\frac{2^{-jw}}{2}$

Proof: We know that $k = g * r + h$ and

$k^j = G_{z \rightarrow j} * r + H_{z \rightarrow j}$

Now $k_1^j = G_{1z \rightarrow j} * r_1 + H_{1z \rightarrow j}$

$$k_2^j = G_{2z\to j} * r_2 + H_{2z\to j}$$

Case(i) : Let $h_1 = h_2 \Rightarrow H_{1z\to j} = H_{2z\to j}$

(3.22)

For $k_1^j = k_2^j$ we have

$$G_{1z} * r_1 + H_{1z\to j} = G_{2z\to j} * r_2 + H_{2z\to j}$$

$G_{1z\to j} * r_1 = G_{2z\to j} * r_2$ From (1)

$$\left\lfloor \frac{G_{1z\to j} * r_1}{2^{1+jw}} \right\rfloor = \left\lfloor \frac{G_{2z\to j} * r}{2^{1+jw}} \right\rfloor \quad (3.23)$$

Since $z = \left\lceil \frac{l}{w} \right\rceil$ it follows that

$$z \le \frac{l-1}{w}$$

$$l + zw \le l$$

For $r_1 \ge 2^l$ and $r_2 \ge 2^l$, $r_1 \ge 2^{l+jw}$ and $r_2 \ge 2^{l+jw}$

$$\frac{r_1}{2^{1+jw}} \ge \frac{2^{1+zw}}{2^{1+jw}} \text{ and } \frac{r_2}{2^{1+jw}} \ge \frac{2^{1+zw}}{2^{1+jw}}$$

Therefore $\frac{r_1}{2^{1+jw}} \neq 0$ and $\frac{r_2}{2^{1+jw}} \neq 0$

Eq(2) possible only when $G_{1z\to j} = 0$ and

$$G_{2z\to j} = 0$$

Therefore the probability of occurrence of minor collision: $\frac{2^{1+zw} \diagup 2^{1+jw}}{2^{1+zw}} = \frac{1}{2^{1+jw}}$

$$= \frac{2^{-jw}}{2}$$

The above probability is the half of that of major collisions

Case(ii) : Let $g_1 \neq g_2$ and $h_1 \neq h_2$

For $k_1^j = k_2^j$

$$G_{1z} * r_1 + H_{1z\to j} = G_{2z\to j} * r_2 + H_{2z\to j} + \left\lfloor \frac{H_{1z\to j}}{2^{1+jw}} \right\rfloor = \left\lfloor \frac{G_{2z\to j} * r}{2^{1+jw}} \right\rfloor \quad (3.24)$$

Since z = $\left\lceil \frac{l}{w} \right\rceil$

$$z \leq \frac{l-1}{w}$$

$1 + zw \leq l$

Now $r_1 \geq 2^l$, $r_2 \geq 2^l$

$r_1 \geq 2^{l+jw}$, $r_2 \geq 2^{l+jw}$

$\frac{r_1}{2^{1+jw}} \geq \frac{2^{1+zw}}{2^{1+jw}}$ and $\frac{r_2}{2^{1+jw}} \geq \frac{2^{1+zw}}{2^{1+jw}}$

$\frac{r_1}{2^{1+jw}} \not\longrightarrow 0$ and $\frac{r_2}{2^{1+jw}} \not\longrightarrow 0$

Eq(3) possible only when $G_{1z\to j} = 0$ and $G_{2z\to j} = 0$

Therefore the probability of occurrence of minor collision: $\frac{2^{1+zw} / 2^{1+jw}}{2^{1+zw}} = \frac{1}{2^{1+jw}}$

$$= \frac{2^{-jw}}{2}$$

3.5.2.1.. Mathematical Proof of the Minor Collisions

1. Let n = 11, z=11.
2. [1, z] = [1, 11].
3. j € [1, z-1] $\Longrightarrow$ j = 6.
4. z-1 = 10 $\Longrightarrow$ 11_2 = 1011. $\Longrightarrow$ Window length w = 4
5. we know that $z = \left\lceil \frac{l}{w} \right\rceil$
6. $z = \left\lceil \frac{l}{w} \right\rceil$ ⅛
7. $z \leq l\text{-}1 / w$
8. $11 \leq l\text{-}1 / 4 \Longrightarrow 44 \leq l\text{-}1$
9. $l = 45$
10. l = 45, 46, 47, 48
11. Now $z = \left\lceil \frac{l}{w} \right\rceil$
12. $z\text{-}1 \leq l\text{-}1 /w$
13. $w(z - 1) \leq l\text{-}1$
14. $w(z - 1) + 1 \leq l$ since w = 4, z = 11, l = 46
15. $4(11\text{-}1) + 1 \leq 46$
16. $41 \leq 46$
17. $r_1, r_2 \geq 2^l \Longrightarrow r_1, r_2 \geq 2^{46}$ $\therefore r_1, r_2$ € [46 , ∞]
18. The probability of Major collisions = $\frac{2^{-jw}}{2} = 2^{-(6)(4)} / 2 = 2^{-24} / 2$

$= 0.0000000298025$

$= 0.298025 \times 10^{-7}$

68

3.5.2.2. Implementation of Minor Collisions

```
package javaapplication3;

import java.util.Scanner;
import java.util.Vector;

public class Main {

    public static void main(String[] args) {

        Scanner s=new Scanner(System.in);

        System.out.print("Enter a Prime Integer:");

        int z=s.nextInt();
        int n=z;
        System.out.print("Enter a Value in btw., (1-"+n+"):");
        int j=s.nextInt();
        int w=Integer.toBinaryString(n).length();

        int l1=z*w;
        int l2=n*w;
        int l=(z*w)+1;

    Vector<Integer> vL=new Vector<Integer>();
        for(int i=l2;i<=l1;i++)
        {

       if(i>=z && i<=n)
       { vL.add(i);}
       }


  //  Vector vR=new Vector();
  //  for(int i=0;i<vL.size();i++)
  //  {
    double r=Math.pow(2,l);
//    }
float temp=-(j*w);
    double prob=Math.pow(2,temp);
    prob=prob/2;

    System.out.println("W="+w);
    System.out.println("L="+l);
    System.out.println("R="+r);
    System.out.println("R="+Math.pow(2,45));
    System.out.println("Probability of Minor Collisions ="+prob);
   System.out.println("Probability of Minor Collisions ="+Math.pow(2,-25));
```

}

}

Enter a Prime Integer:
11
Enter a Value in btw., (1-11):
7
W=4
L=45
R=3.5184372088832E13

R=3.5184372088832E13

Probability of Minor Collisions =1.862645149230957E-9

Probability of Minor Collisions =2.9802322387695312E-8

BUILD SUCCESSFUL (total time: 28 seconds)

Algorithm 3. 5.2.3. Fixed-Sequence Window Method

Input: Window width w, d = $\lceil n/w \rceil$ an n-bit odd integer e and P $\in$ E(Fp).

Output: eP

1. Precomputation.

   1.1 T[2^{w-1}w−1] ← P.

   1.2 T[2^{w-1}-1] ← 2P.

   1.3 for i from 2^{w-1} to 2^{w} − 2 do

   T[i + 1] ← T[i] + T[2^{w} − 1].

   1.4 for i from 2^{w-1} − 1 down to 0 do

   T[i] ← −T[2^{w} − 1 − i].

2. e’ = SHR(e) = (E'_{d-1} . . .E'_{0})2^{w}.

3. Q ← T[E'_{d-1} + 2^{w-1}].

4. for i from d − 2 down to 0 do

4.1 Q ← 2^{w} Q.

4.2 Q ← Q + T[$E_1^{'}$].

5. Return (Q).

3.5.2.3. The Effect of Probability of Minor Collisions on Fixed Sequence Window

The Probability of major collisions = 2^{-jw+1} , j $\in [1, z-2]$

Mathematically we find the probability of occurrence of minor collision:

$$= \frac{2^{-jw}}{2}, \text{j} \in [1, z]$$

In the Existing system according to major collisions the condition of major collision depends on the least significant jw+1 bits of g the probability of occurrence of this collision is around 2^{-jw+1} for both values of h_{jw} which are expected to be equally likely. The condition of minor collision depends on the value of the least significant jw bits of length. The probability of the occurrence of these collisions is around $\frac{2^{-jw}}{2}$ for the both values of h_{jw} which are expected to be equally likely.

Chapter 4

SECURITY THROUGH ELLIPTIC CURVES FOR WIRELESS NETWORKS IN MOBILE DEVICES

4.1. INTRODUCTION

Elliptic curve cryptosystems (ECCs) are suitable for implementation on devices with limited memory and with less computational capability such as smart cards and also with limited power such as wireless hand held devices. This is due to the fact that elliptic curves over large finite fields provide the same security level as other cryptosystems such as RSA for much smaller key sizes.

Considering power analysis attacks, there are two main types that were presented by Kocher et al. These are simple and differential power analysis attacks (referred to as SPA and DPA respectively). Both of them are based on monitoring the power consumption of a cryptographic token while executing an algorithm that manipulates the secret key.

We cannot perform heavy cryptographic computations such as public key cryptography with a 2048-bit level security in mobile devices. But in elliptic curves are suitable for smaller key size with bit level of 170-180 achieved with 1024 RSA bit level of security. Simple power analysis attacks are the attacks to retrieving the valuable information processing the key from a single leaked information, power consumption or electromagnetic emance trace.

We implement a secret key to avoid retrieving the valuable information by the attacker through SPA attacks. The secret key is proposed as a prime integer in proposed EKR Modified Montgomery Inversion algorithm.

4.2. Key Splitting Methods

We discuss about the SPA-resistant algorithms and compare the resultant performance when combined with each form of key splitting.

This approach was suggested by Clavier and Joye in [15] and revisited by Ciet [10] as follows. In order to compute the point kP, the n-bit key k is written as

$$k = k_1 + k_2,$$

such that $k_1 = k - r$ and $k_2 = r$, where r is a random integer of length n bits. Then kP is computed as

$$kP = k_1 P + k_2 P. \quad (4.1)$$

It is important to note that each of the terms of (4.1) should be evaluated separately and their results combined at the end using point addition. That is the multiple-point multiplication methods that use a common accumulator to save doubling operations.

Lemma 4.2.2. Let splitting scheme I at the end of some iteration j, $0 < j \leq n - 1$, there are only two possible values for Q, those are $[k_{n-I-j}] P$ or $[k_{n-I-j} - 1] P$.

Proof. : $k_1 = (k_{1\,n-1} \ldots k_{1\,0})_2$ and $k_2 = (k_{2\,n-1}, \ldots, k_{2\,0})_2$ from the most significant end down to the least significant end. Hence, at the end of iteration j, the accumulator Q contains the value

$$Q = k_{1n-I-j} P + k_{2n-I-j} P \quad (4.2)$$

$$= [k_{1n-I-j} + k_{2n-I-j}] P.$$

We can write k, k_1 and k_2 as

$$k = k_{n-I-j^j} 2^j + k_{j-1-0}$$

$$k_i = k_{i\ n-I-j^j} 2^j + k_{i\ j-1-0} \quad (4.3)$$

Since $k = k_1 + k_2$ we have

$$k_{1\ j-I-0} + k_{2\ j-I-0} j = k_{j-I-0} + b\, 2^j \text{ where } b \in \{0, 1\} \quad (4.4)$$

and $$k_{1\ n-I\to j} + k_{2\ n-I\to j} = k_{n-I\to j} - b$$

The DPA attack would proceed in the same way, whether the algorithm processes a single bit or a digit per iteration, though it would be more involved in the latter case depending on the digit size.

4.3. Modular division:

In the following algorithm, a and b are integers internally represented each by an array of w-bit digits. The length of each array is d = $\lceil n/w \rceil$ digits. Savas and Koc mention that b in Modular Inversion is need not be less than the modulus u, but will be in $[1, 2^m - 1]$ where m = dw. Also note that the values $R^2 \bmod u$, where $R = 2^m$, and u^1 are computed once per modulus, i.e., per curve.

Algorithm 4.3. 1. Modular Division

Input: u: a n-bit prime, d = $\lceil n/w \rceil$, m = dw, $R^2 \pmod u = (2m)^2 \pmod u$,

$u^1 = u^{-1} \bmod 2^w$, $a \in [1, p-1]$ and $b \in [1, 2^m - 1]$. Output: $ab^{-1} \pmod u$.

1. Compute $b^{-1} R \pmod u$.
2. Compute $x = a(b^{-1}R)R^{-1} \pmod u$
3. Return(x).

Algorithm. 4.3.2. Montgomery Multiplication

Input: u: a n-bit prime, d = $\lceil n/w \rceil$, m = dw, $u^1 = u^{-1} \bmod 2^w$, $x = (x_{d-1} \ldots, x_0)2^w$

and $y = (y_{d-1} \ldots y_0)2^w$. Output: $xy2^{-m} \pmod u$.

1. A ← 0. // $A = (ad, a_{d-1}, \ldots, a_0)2^w$
2. for i from 0 to d − 1 do
   2.1 $u_i \leftarrow (a_0 + x_i y_0) \bmod 2^w$
   2.2 $A \leftarrow (A + x_i y_0 + u_i m)/ 2^w$
3. if (A > u) then A ← A- u.
4. Return (A).

The following algorithm was presented by Savas and Koc [] as the modified Kaliski-Montgomery Inverse.

Algorithm 4.3.3. Montgomery Inversion

Input: u: a n-bit prime, d = $\lceil n/w \rceil$, m = dw, $R^2 \pmod u = (2^m)^2 \pmod u$,

$u^1 = u^{-1} \bmod 2^w$ and $b \in [1, 2^m - 1]$. Output: $b^{-1} R \pmod u$.

1. Compute f and x = $b^{-1} 2f$ Where $n \le f \le m + n$.
2. if ($f \le m$) then
   - 2.1 x ← $xR^2 R^{-1} \pmod u$ //x = $b^{-1} 2^{m+n} \pmod u$
   - 2.2. f ← f + m. // f > m, x = $b^{-1} 2^f \pmod u$
3. x ← $x2^{2m-f} R^{-1} \pmod u$ // x = $b^{-1} 2^f 2^{2m-f} 2^{-m}$

   x = $b^{-1} 2^m \pmod u$
4. Return(x).

4.4. EXISTING SYSTEM

Nevine Maurice Ebied's modified Almost Montgomery inverse algorithm to be resistant to SPA attacks. In the following algorithm. Swap Address(c, d) denotes interchanging the memory addresses of the integer's c and d. This is an inexpensive operation, hence its usage as a dummy operation to balance the branches of the main loop. Nevine Maurice Ebied's modified Almost Montgomery inverse algorithm "if" statement was implemented such that the number of conditions checked three times per loop iteration. The algorithm was implemented in Java, step 3.4 as

if((xLSb = = 0) && (xLSb = = 0) && (xLSb = = 0)).

If the condition is false, due to short-circuit evaluation, then the flow control will move to the following "if" after the first check, otherwise, it will perform the check three times. The following "if step is similar but with the condition checked only two times

if((yLSb = = 0) && (yLSb = = 0)).

Algorithm 4.4.1. Almost Montgomery Inverse

Input: u: a n-bit prime, d = $\lceil n/w \rceil$, m = dw and $b \in [1, 2^m - 1]$.

Output: f and $b^{-1} 2^f \pmod u$, where $n \le f \le m + n$.

1. x ← u; y ← b; r ← 0; s ← 1.

2. f ← 0.

3. While (v > 0) do

3.1. U ← x − y; V ← −U.

3.2. T ← r + s.

3.3. f ← f + 1.

3.4 if (((lsb(x) = 0))) then // This "if" is special

Swap Address(x, U); SwapAddress (x, U) // dummy

SHR(x); SHL(s).

3.5. else if ((lsb(y) = 0)) then // This "if" is special

SwapAddress(y, V);Swap Address (y, V) // dummy

SHR(y); SHL(r).

3.6. else if (V >= 0) then

SwapAddress(y, V); SwapAddress(s, T)

SHR(y); SHL(r).

3.7. else

Swap Address(x, U); Swap Address(r, T)

SHR(x); SHL(s).

4. T ← u − r; V ← u + T .

5. if (T > 0) then

Return (f, T)

Else

Return (f, V).

The drawback of this algorithm is that an SPA of the number of iterations of the main loop directly leaks the value of f. If f is uniformly distributed, the search space of b s reduced from 2^w to $2^{m-\log 2^m}$, which is not a significant reduction. It is interesting to study how f is actually distributed.

4.5. E.K.R Modified Montgomery Inversion

4.5.1. Linear Congruence's

A congruence of the form ax ≡ b (mod m) where m is a positive integer, a and b are integers, and x is a variable, is called Linear congruence. Such congruence's arise throughout number theory and its applications.

4.5.1.1. Definition: If a and b are integers, then a is said to be congruent to b modulo n, write a ≡ b (mod n), if n divides (a – b). The integer n is called the modulus of the congruence.

4.5.1.2. Definition: The **equivalence class** modulo n of an integer b is the set of all integers congruent to b modulo n.

4.5.1.3. Definition: The ring of **integers modulo** n, denoted by Zn, is the set of (equivalence classes of) is the integers {0, 1, 2, n–1}. Addition, subtraction, and multiplication in Zn are performed modulo n.

We modified the Nevine Maurice Ebied's Almost Montgomery inverse and A **SECRET KEY** of Savas and Koc [] to be resistant to SPA attacks as in the following algorithm.

Algorithm 4.5.1 EKR Modified Montgomery Inversion

Input: u: a n-bit prime, d = $\lceil n/w \rceil$,

m = dw, $R^2 \pmod u = (2^m)^2 \pmod u$, $u^1 = u^{-1} \bmod 2^w$ and b $\in$ $[1, 2^m - 1]$,

t is Secret key. t: No of precomputed points $1 \leq t \leq n$

W: Window width least significant of bit

$2^{w-z} \leq t \leq 2^{w-1}$

Output: $b^{-1} R \pmod u$.

1. Select a number b such that $(b, 2^m) = 1$
2. Compute b such that $bb^{-1} \equiv 1 (\text{mod } 2^m)$
3. If $f > m$ then $x = b^{-1} 2^f \text{ (mod u)}$ $\because x = b^{-1} 2^f \text{ (mod u)}$
4. If $(f \leq m)$ then
5. $x \leftarrow R^2 R^{-1} \text{(mod u)}$ $\because R = 2^m$
6. $x = b^{-1} 2^{m+f} \text{ (mod u)}$ $f \leftarrow m+f$
7. Return(x)

In EKR modified Montgomery Inverse Algorithm of Savas and Koc, we select f such that gcd $(b, 2^m) = 1$, $m \leq f \leq m+n$. So b is not reduced from 2^m to $2^{m-\log 2m}$. Therefore this is a significant reduction and because f is not uniformly distributed and it can't leak the value

4.5.2. Mathematical Proof of EKR Modified Montgomery Inverse Algorithm

Input: u: a n-bit prime, $d = \lceil n/w \rceil$, $m = dw$, $R^2 \text{ (mod u)} = (2^m)^2 \text{ (mod u)}$, $u' = u^{-1} \bmod 2^w$ and $b \in [1, 2^m - 1]$.

t: No of precomputed point's $1 \leq t \leq n$

w: Window width Least significant of bit $2^{w-z} \leq t \leq 2^w - 1$

Output: $b^{-1} R \text{ (mod u)}$.

Step (1). U = 17; $5_{10} = 101_2 \Rightarrow w = 3$

$$d = \left\lceil \frac{n}{w} \right\rceil = \left\lceil \frac{5}{3} \right\rceil = \lceil 1.266 \rceil = 2$$

$m = dw = (2)(3) = 6$

$m = 6$

Since $1 \leq t \leq n \Rightarrow 1 \leq t \leq 5$

Let us choose $t = 5$ (secret key)

$2^{w-z} \leq t \leq 2^w - 1$

$2^{w-z} \leq 5 \leq 2^{w-1}$

W = 1 $\Rightarrow$ ½ ≤ 5 ≤ 1 wrong

W = 2 $\Rightarrow$ 1 ≤ 5 ≤ 3 wrong

** W = 3 $\Rightarrow$ 2 ≤ 5 ≤ 7 *** Least Integer

W = 4 $\Rightarrow$ 4 ≤ 5 ≤ 15

W = 8 $\Rightarrow$ 4 ≤ 5 ≤ 31

Least Integer w = 3

2. Select a number b such that $(b,2^m)=1$

$2^m = 2^6 = 64$

$\gcd(b,2^m)=1 \Rightarrow \gcd(6,64) = 1$

Let us we choose b = 7

3. Compute b^{-1} such that $bb^{-1} \equiv 1 (\text{mod } 2^m)$

$(7\, b^7) \equiv 1 (\text{mod } 64)$

$7(55) \equiv 1 (\text{mod } 64)$

$b^{-1} \equiv 55 (\text{mod } 64)$

$n \leq f \leq m + n$

$5 \leq f \leq 6 + 5$

$5 \leq f \leq 11$

f = 5, 6, 7, 8, 9, 10, 11,

4. If f > m then $x \equiv b^{-1} 2^f (\text{mod } u)$ $\because x \equiv b^{-1} 2^f (\text{mod } u)$

Since m = 6 f > m

$f = 7 \Rightarrow x = (55)\, 2^{7} \pmod{17} \Rightarrow x \equiv 2 \pmod{17}$

$f = 8 \Rightarrow x = (55)\, 2^{8} \pmod{17} \Rightarrow x \equiv 4 \pmod{17}$

$f = 9 \Rightarrow x = (55)\, 2^{9} \pmod{17} \Rightarrow x \equiv 8 \pmod{17}$

$f = 10 \Rightarrow x = (55)\, 2^{10} \pmod{17} \Rightarrow x \equiv 16 \pmod{17}$

$f = 11 \Rightarrow x = (55)\, 2^{11} \pmod{17} \Rightarrow x \equiv 15 \pmod{17}$

5. If $f \leq m$ then $x \leftarrow R^{2} R^{-1} (\mathrm{mod} u)$ $\because R = 2^{m}$

$x = b^{-1} 2^{m+f} \pmod{u}$ $f \leftarrow m+f$

$f = 5 \Rightarrow x \equiv (55)\, 2^{6+5} \pmod{17}$

$x \equiv 15 \pmod{17}$

$f = 6 \Rightarrow x \equiv (55)\, 2^{6+6} \pmod{17}$

$x \equiv 13 \pmod{17}$.

4.5.3. Performance and Complexity

The Montgomery Inversion and Almost Montgomery Inversion algorithms are to protect against SPA attacks. Both are focused on prime fields. So we analyze the Almost Montgomery inversion algorithm and introduce a secret key implement a new algorithm EKR Modified Montgomery inversion algorithm to implement a secret key as a input

According to the time units depends upon the value of function f , the loop will be check number of iterations. The function f will be greater than m, loop can be check m + n times. If the value of function f less than to m and loop can be check m times.

Complexity: The complexity of the algorithm is depends on f. The loop will be check number of iterations.

The function f will be greater than m complexity = m + n.

If the value of function f less than to m complexity = m

4.5.2. Implementation of EKR Modified Montgomery Inverse Algorithm:

```
package javaapplication1;
import java.util.ArrayList;
import java.util.Scanner;
public class Main {
public static void main(String[] args)
   {  Scanner s=new Scanner(System.in);
      System.out.println("Enter a Prime Integer(U):");
      int u=s.nextInt();
      String binU=Integer.toBinaryString(u);
      int n=binU.length();
      System.out.println("Enter a secrete Key(t):");
       int t=s.nextInt();
      double w=0;
for(int wi=1;wi<=t;wi++)
{
   int i1=(int)Math.pow(2,wi-2);
   int i2=(int)Math.pow(2,wi)-1;
   if(i1<=t && t<=i2)
   {
      w=wi;
      break;
   }
}
```

```
System.out.println("Window width (W): "+w);
int d=(int)Math.ceil(n/w);
System.out.println("Length of Each Array (d) :"+d);
int m=(int)(d*w);
System.out.println("....(m) :"+m);
int on1=(int)Math.floor(Math.pow(2,m-1));
int n2=(int)Math.pow(2,m);
for(int i=m;i<on1;i++)
{int n1=i;
while(n1!=n2)
 {
        if(n1>n2)
                n1=n1-n2;
        else
                n2=n2-n1;
 }
 //System.out.println("GCD of two number is "+n1+"and i is "+i);
if(n1==1){on1=i;break;}
}
System.out.println("B="+on1);
int b=on1;
int b_inverse=0;
for(int i=1;i<b;i++)
{
      int t2m=(int)Math.pow(2,m);
```

```
        int temp=t2m*i;
        temp=temp+1;
       b_inverse=temp/b;
if((temp%b)==0)
{
   break;
}
}
System.out.println("B inverse: "+b_inverse);
ArrayList<Integer> af=new ArrayList<Integer>();
for(int i=n;i<=(m+n);i++)
{    af.add(i);
}
int x=0;
for(int f=n;f<=(n+m);f++)
{
if(f>m)
{
 x=(int)(b_inverse*Math.pow(2, f))/u;
System.out.println("f="+f+"\nx="+x);
}
else if (f<m)
{
   x=(int)(b_inverse*(Math.pow(2,m+f)))/u;
```

```
System.out.println("f="+af.get(f)+"\nx="+x);
}

}

   }

}
```

C. Input and output

```
Enter a Prime Integer(U):
111
Enter a secrete Key(t):
117
Window width (W): 7.0
Length of Each Array (d) :1
....(m) :7
B=7
B inverse: 55
f=8
x=126
f=9
x=253
f=10
x=507
```

f=11

x=1014

f=12

x=2029

f=13

x=4059

f=14

x=8118

BUILD SUCCESSFUL (total time: 29 seconds)

4.6. CONCLUSION

The proposed EKR Modified Montgomery Algorithm is to be resistant to SPA attacks .The EKR Modified Montgomery Inverse Algorithm eliminate the number of Iterations of the main loop directly leaks the value of f and also it is mathematically proved that f is uniformly distributed with a significant reduction

A function that is easy to evaluate but its inverse is infeasible unless the secret key t is known. So the attacker can not guess the key (t) to retrieve the valuable information in smart cards and mobile devices.

CHAPTER 5

THE ADVANTAGES OF ELLIPTIC CURVE CRYPTOGRAPHY FOR WIRELESS SECURITY

5.1. Introduction

Applications of healthcare, financial services and government depend on the underlying security already available in the wired computing environment. Both for secure (authenticated, private) Web transactions and for secure (signed, encrypted) messaging, a full and efficient public key infrastructure is needed. Three basic choices for public key systems are available for these applications:

- RSA
- Diffe-Hellman (DH) or Digital Signature
- Algorithm (DSA) modulo a prime *p*
- Elliptic Curve Diffie-Hellman (ECDH) or Elliptic Curve Digital Signature Algorithm (ECDSA)

The RSA is a system that was published in 1978 by Rivest, Shamir, and Adleman, based on the difficulty of factoring large integers. Whitfield Diffie and Martin Hellman proposed the public key system now called Diffe-Hellman Key Exchange in 1976. DH is the key agreement and DSA is the signature, and they are not directly interchangeable, although they can be combined to do authenticate key agreements. Both the key exchange and digital signature algorithm are based on the difficulty of solving the discrete logarithm problem in the multiplicative group of integers modulo a prime *p*.

5.2. Equivalent Security Levels

The same level of resistance against the best known attacks, the system parameters for an elliptic-curve-based system can be chosen to be much smaller than the parameters for RSA or mod *p* systems. For example, an elliptic curve over a 163-bit field currently gives the same level of security as a 1024-bit RSA modulus or Diffie-Hellman prime. The difference becomes even more dramatic as the

desired security level increases. Public key protocols are used in combination with symmetric key algorithms.

The overall strength of the system is weak. Recently the new federal Advanced Encryption Standard (AES) was introduced, providing greater security than its symmetric key predecessor. At key lengths of 128, 192, and 256, AES has made ECC systems even more attractive as a key agreement alternative. This growing difference in key bit length for equivalent security levels accounts for the performance advantages to be obtained from substituting ECC for RSA/DH/DSA in public key Cryptographic protocols. The number of standard documents is shown in the table below. [10].

Symmetric	ECC	DH/DSA/RSA
80	163	1024
128	283	3072
192	409	7680
256	571	15,360

Table5-1 Key sizes for equivalent security levels (in bits)

5.3. Role of Groups in Key Exchange and Signatures

The Diffe-Hellman Key Exchange and Digital Signature Algorithm can be described abstractly in the mathematical language of groups. A group is a set of elements with an operation specifying how to combine two elements to get another element of the set such that the operation satisfies certain technical properties. When we refer to mod *p* in groups where *p* is some large prime number, we mean the set of natural numbers less than *p* where the operation is multiplication taking the remainder modulo *p*.

A secret key exchange between two parties, A and B, can be achieved publicly if a group *G* and a fixed group element *g* are agreed upon, and if each generates a random number which they keep to themselves. If A generates the random number *a* and broadcasts the group element *ga* and B generates

the random number *b* and broadcasts the group element *gb*, the common secret will be *gab*, where the notation *ga* means to compose the element *g* with itself *a* times in the group. The security of this protocol depends on the discrete log problem being hard to solve in the given group. The discrete log problem is: given *ga* and *g*, find *a*.

5.4. Standard Techniques for Fast Exponentiation

Many different fast exponentiation techniques are used to perform group exponentiations. For example, to perform binary exponentiation, express the exponent as a binary string; then for each bit in the expansion, either performs a squaring or a squaring and a multiplication with the base, depending on whether bit 0 or 1 occurs in the expansion. More sophisticated and efficient versions of the binary method have been developed [7]. Other methods available involve:

1. Windowing: using some precomputed values of the base and different "window" sizes to break up the binary expansion of the exponent into chunks to be processed iteratively
2. NAF: nonadjacent form of the exponent so that no two adjacent bits are both non -zero. Compressible exponents [9] for methods 1 and 2, and [69]
3. For elliptic curves, the group operation is written as addition instead of multiplication, and in that case exponentiation is more appropriately referred to as scalar multiplication, but the same techniques apply. For elliptic curves, the "square-and- multiply" technique described above is referred to as double-and-add. When using affine coordinates, a field multiplication can be saved each time a double-and-add operation is performed in a scalar multiplication [24], leading to a more efficient implementation of elliptic curve cryptographic protocols for general elliptic curves.

When field inversions are more costly than 6 field multiplications, another technique given in [13] is beneficial. It allows one to trade an inversion for 6 multiplications, and leads to an efficient algorithm for tripling a point on general elliptic curves. Finding further ways to improve elliptic curve scalar multiplication is an active area of research [6], [18] and [17].

5. 5. S/MIME

MIME stands for Multipurpose Internet Mail Extensions, a specification for formatting messages so that they can be sent over the Internet. MIME was initiated in 1992 by the IETF. S/MIME stands for Secure MIME, and provides the following security services for electronic messages [14]: authentication, message integrity and non-repudiation of origin (using digital signatures) and privacy and data security (using encryption).

5.5.1. PROTOCOL

Suppose two parties A and B wish to exchange signed encrypted messages. Assume that A and B already have their own public key/private key pairs, and have certificates on their public keys from some common trusted certificate authority (CA). If A wants to send a message to B, A can obtain the certificate on B's public key and check its validity. Then using B's public key, A encrypts a message to B (usually just a symmetric key that subsequently acts as the content encryption key). A may include data such as a message encrypted with a symmetric key algorithm using the content encryption key. A then signs the whole message using its own private key. When B receives the data from A, B first checks the certificate on A's public key for validity. B then uses A's public key to verify A's signature on the message. B uses its own public key to decrypt the content encryption key and uses that key to decrypt the received data [68].

To analyze the work load on each party, note that the sender and receiver must each perform three public key operations.

The sender, A, must:

1) Verify 1 signature (on B's certificate)

2) Perform 1 encryption (using B's public key)

3) Sign message

The receiver, B, must:

4) Verify 1 signature (on A's certificate)

5) Verify 1 signature (on A's message)

6) Perform 1 decryption (using its own private key)

Currently RSA certificates are often issued even on elliptic curve public keys. Certificates are signed only once but verified many times, and since only the CA must perform the expensive RSA signing operation using its private key, the individual parties do not incur much computational burden in performing the RSA signature verifications of the certificates. However, using an elliptic curve cryptosystem to perform steps 2, 3, 5, and 6 can significantly decrease the computational burden on the individual parties.

5.5.2. Performance Advantages Comparison of ECC with RSA

We will start by giving some sample timings for RSA and ECC on different platforms. In Table 5-2 rows 1 and 2 are taken from [1], and do not claim to be optimized, but show two different platforms and are directly comparable to RSA numbers for the same platforms. Rows 3 and 4 in Table 5-2 are taken from [6] and take advantage of the special form of the generalized Mersenne primes for the NIST curves given in [15] by using specialized routines for fast modular reduction for these primes [3]. Row 3 uses affine coordinates and a binary NAF for the exponent. Row 4 uses mixed Jacobian affine coordinates and a windowed NAF for the exponent. In Table 5-3 RSAd is the private key operation, whereas RSAe is the public key operation. Rows 1 and 2 are from [27], as above in the elliptic curve timings. Row 3 is from [18].

S.NO	Processor	MHZ	163-bit	193-bit	256-bit	384-bit	521-bit
1	Ultra SparcII	450	6.1	8.7	-	-	-
2	Strong ARM	200	22.9	37.7	-	-	-
3	Pentium II	400	-	18.3	42.4	136.4	310.4
4	Pentium II	400	-	2.1	5.1	16.4	27.8

Table. 5-2. Sample elliptic curve exponentiation timings over prime fields (in ms)

S.NO	Processor	MHZ	1024-RSAd	1024 RSAe	2048RSAd	2048RSAe
1	Ultra SPARC II	450	32.1	1.7	205.5	6.1
2	Strong-Arm	200	188.7	10.8	1273.8	39.1
3	ARM7TDMI	1	12,070	1180	-	-

Table 5-3 Sample RSA encrypt/decrypt timings (in milliseconds).

5.5.3. Comparisons

At the 163-bit ECC/1024-bit RSA security level, an elliptic curve exponentiation for general curves over arbitrary prime fields is roughly 5 to 15 times as fast as an RSA private key operation, depending on the platform and optimizations. At the 256-bit ECC/3072-bit RSA security level the ratio has already increased to between 20 and 60, depending on optimizations. To secure a 256-bit AES key, ECC-521 can be expected to be on an average of 400 times faster than 15,360-bit RSA.

5.5.4. Impact on S/Mime

For example, if elliptic curve cryptosystems are used in the S/MIME protocol for signing and encryption, the sender will perform an ECDSA signature and an ECDH key agreement operation instead of an RSA signature and an RSA encryption. For the purpose of mobile communication, a Strong-Arm 200 MHz processor can be thought of as a typical device. Without assuming an optimized version of ECC, use row 2 of the ECC table instead of row 4. On such a device, at the current minimum 1024-bit security level, the difference in the computation time for the sender per message is roughly 47 ms instead of 200 ms. For the recipient of a message, it would mean performing an ECDSA signature verification and an ECDH operation instead of an RSA signature verification and an RSA decryption, so the difference would be about 79 ms instead of 200 ms/message. Using the optimized versions of ECC quoted in row 4, the costs for the sender and receiver per message can be cut to roughly 8 and 12 ms. At higher security levels (e.g., for keying AES-256), RSA operations will be far too costly computationally for such a small device (roughly 45 s for sender and receiver per

message),whereas ECC will cost instead roughly 56 ms for the sender per message and 84 ms for the receiver per message.

5.5.5. Identity-Based Encryption

Another application of elliptic curves in cryptography has recently emerged in the form of a new system for doing ID Encryption. ID Encryption is a public key encryption scheme where *any* string can be a user's public key, including, for example, the user's email address or name. The advantage of ID-Based Encryption is that no certificate is needed to bind names to public keys. This feature may help to launch a public key infrastructure, since a receiver does not have to obtain a public key and a certificate before receiving encrypted communications. The sender can use the receiver's ID as its public key, and does not need to obtain and verify a certificate on the recipient's public key before hand. Once an encrypted communication has been received, a user can contact a central CA to obtain the secret key corresponding to its public key. ID signatures were first proposed by Shamir in 1984 [16], and Maurer- Yacobi proposed ID key agreement in 1991 [11], but these schemes were not deployed due to heavy computational costs. In 2001 Boneh and Franklin proposed a scheme that can be implemented efficiently using elliptic curves and an associated pairing map [19]. The security of the new scheme depends on the assumption that a new problem, the Bilinear Diffe-Hellman (BDH) problem, is hard.

5.6. Performance Comparisons of Encryption and Decryption Time in RSA With Different Platforms

5.6.1. Key Generation Time Performance

S.NO	PROCESSOR	MHZ	BITS	KEYGENERATION TIME
1	WINDOWS Xp2	2.81	256	78
	112MB OF RAM		512	391
			1024	3698
			2048	7012
2	WINDOWS Xp2	2.18	256	219
S.NO	1.96 GB OF RAM		512	500
			1024	4297
			2048	35000

Table.5-4. Key generation time performance

5.6.2. Encryption Time Performance

S.NO	PROCESSOR	MHZ	BITS	ENCRYPTION TIME	EXISTING SYSTEM
1	Windows Xp2	2.81GHZ	256	5	
	112MBOF RAM		512	16	
			1024	94	
			2048	156	
2	Windows Xp2	2.18GHZ	256	0	
	1.96GB OF		512	16	
1	ULTRA SPARC	450	1024	78	1.7
2	STRONG ARM	200			10.8
3	ARM 7TDMI	1			1180
2	Windows Xp2	2.18GHZ	2048	406	
1	ULTRA SPARC	450			6.1
2	STRONG ARM	200			39.1
3	ARM 7TDMI	1			---

Table.5-5. Encryption Time Performance

5.6.3.Decryption Time Performance

S.NO	PROCESSOR	MHZ	BITS	DECRYPTION TIME	EXISTING SYSTEM
1	Windows Xp2	2.81GHZ	256	16	
	112 MB OF RAM		512	47	
			1024	203	
			2048	406	
2	Windows Xp2	2.18GHZ	256	0	
	1.96 GB OF RAM		512	94	
1	**ULTRA SPARC**	**450**	**1024**	**351**	**32.1**
2	**STRONG ARM**	**200**			**188.7**
3	**ARM 7TDMI**	**1**			**12070**
2	Windows Xp2	2.18GHZ	2048	2625	
1	**ULTRA SPARC**	**450**	**2048**		**205.5**
2	**STRONG ARM**	**200**			**1273.8**
3	**ARM 7TDMI**	**1**			**---**

Table.5-6. Decryption time performance

5.6.4. Performance Comparisons

The performance of this system 163 bit equivalent 1024 RSA security level is on elliptic curve exponentiation for general curves over 5 to 10 times faster than RSA. The performance of encryption and decryption performance time increases from bits 256 to 2048 depending upon different platforms. The encryption and decryption performance time increases or decreases depend upon the type of messages.

A mobile phone has very limited bandwidth, memory, computational power and battery power, but the mobile phones cannot perform heavy (cryptographic) computations such as public key cryptography with a 2048-bit key. The advantages of elliptic curve cryptography were with smaller key sizes i.e. bit level of 170-180 achieved with 1024 RSA level of security.

5.7. Performance Analysis between EKR Modified Montgomery Inversion Algorithm and Fast Exponentiation

S.NO	PROCESSOR	ITS	Multiple keys in ms	Single keys in ms	Existing System P-II Row 3	Row 4
1	Windows Xp2	163	0.0065	0.018	---	---
	1.96 GB RAM	192	0.015	0.011	18.3	2.1
	2.18 GHZ	256	0.013	0.009	42.4	5.1
		384	0.031	0.013	136.4	16.4
		512	0.01	0.009	---	---
		521	0.014	0.008	310.4	27.8
		1024	0.012	0.008		
		2048	0.012	0.01		

Table5-7.Performance Analysis between EKR MM Inversion Algorithm and F.E

5.7.1. Performance Comparisons

In the existing system the values of Row3 uses affine coordinates and a binary NAF for exponent. The values of Row 4 uses mixed Jacobian Affine coordinates and a windowed NAF for the exponent.

In the proposed EKR Modified Montgomery Inversion Algorithm uses some precomputed values of the base and different window sizes. If the window width is w, some multiple of the point P up to (2^m-1) P are computed and stored. Here k is processed w bit at a time. Let the number of precomputed point be t. In the precomputed stage each point requires either a double or an addition to be computed depending on the algorithm. Compare to performance considerations in EKR Modified Montgomery Inversion Algorithm is faster than when using Fast exponentiation of the above system.

The public key cryptographic operations at 2048 bit level security were suitable for EKR Modified Montgomery Inversion Algorithm in mobile devices but RSA is not suitable.

5.8. Conclusions

The performance of encryption and decryption performance time increases from bits 256 to 2048 depending upon different platforms in RSA. The advantages of elliptic curve cryptography were with smaller key sizes i.e. bit level of 170-180 achieved with 1024 RSA level of security. Compare to performance considerations in EKR Modified Montgomery Inversion Algorithm is faster than when using Fast exponentiation of the above system**.** We provide more security through EKR Modified Montgomery Algorithm for mobile devices features such as messages, Communications and allows Virtual Private Networks to corporate networks, and ECC allows more efficient implementation of all of these features rapidly becoming more dependent on security features such as the ability to do secure email, secure Web browsing, and virtual private networking to corporate networks, and ECC allows more efficient implementation of all of these features.

CHAPTER 6

ELLIPTIC CURVE CRYPTOGRAPHY AND ITS APPLICATIONS

6.1. Introduction

In 1976, Diffe and Martin Hellman introduced the concept of public key cryptography (PKC). Since then, many implementations of it have been proposed, and many of these cryptographic applications are based on their security. The mathematical problems, of integer factorization problem (IFP) and the finite field discrete logarithm problem (DLP) over the years, sub-exponential time algorithms were developed to solve these problems.

Cryptographic applications are based on their security on the intractability of Discrete Logarithm Problem include the Diffe- Hellman key agreement scheme, the Elgamal encryption scheme and the Digital signature Algorithm (DSA). The two problems QS (Quadratic Sieve) and NFS (Number of field sieve) are very similar and all of the modern factoring algorithms can be used to calculate discrete logarithms in the multiplication group of a finite field. Former this point of view the amount of work done in computing the algorithm in Z* P, where P has K-bit composite number n. Therefore if any improvements in algorithms are discovered for either one of the problems, the improvements can also be expected for the problem [22]. In the following section, different type of a problem called the Elliptic curve discrete logarithm problem which is the best algorithm that computes elliptic curve algorithms runs in full exponential time

6.2. ECDLP Algorithms

The best-known general-purpose algorithm for solving the ECDLP is the Pollard ρ-method. It can be sped up with special techniques while running on parallel processors to O ($\sqrt{(\pi n)}$ /(2r)), where n is the number of elliptic curve additions and r is the number of processors running [22]. When the orders of the curves become sufficiently large, however, methods like Pollard ρ and BSGS become infeasible [5]. In 1997, V. Shoup showed that the running time of any algorithm that solves the DLP

for arbitrary groups takes Ω (√P log p) steps, where p is the largest prime factor of N Hence, to significantly improve the efficiency of general purpose algorithms might prove to be a futile task.

6.2.1. Curve Selection and Setup

The preferred method for generating good curves is to select the curves at random. Randomly generated curves are curves with coefficients that are taken from the output of pseudo-random number generators. Below is an example of an algorithm that selects an appropriate curve over Fp.

Input: A large finite field Fp, and a small positive integer s'.

Output: An elliptic curve E over Fp, such that E (Fp) = S .r, S ≤ S' and r prime

1. Draw E at random, with coefficients in Fp.

2. Compute the order of E, #E(Fp).

3. Check the MOV and anomalous conditions. If any one of these fails, go to Step 1

4. Attempt to factor #E(Fp) in 'reasonable' time. If attempt fails, go to Step 1.

5. If #E(Fn) = S .r , S ≤ S' and r prime, then return E. Otherwise, go to Step 1.

If some times E is not chosen random the problem is hard in step(2). The purpose of Step 5 is to make sure that #E(Fp) has a large prime factor. This is to guard against a Pohlig-Hellman attack on the ECDLP.

System parameters	A finite field F, Coefficients that defines the curve E, A point on E called the generator G, And ord (G)
Public key	A point on curve P=kG , for some secret k.
Secret key	The integer k ,where 0<k<q ,q=ord(P)

Table. 6-1 setting up an elliptic curve cryptosystem.

The above table is listing the basic computations necessary to set up elliptic curve cryptosystems. Key pairs are easy to generate; the private key is a randomly chosen integer k, such that the public key P = kG. The basic assumption of ECC is that it is hard to compute the secret key k from the public key P. The elliptic curve parameters can be used for groups of users, where each user in the group has a public and private key pair.

6.3. NIST Recommended Fields and Curves

The Federal Information Processing Standards (FIPS) is a compilation of standards and guidelines issued by NIST for government use. The revised FIPS 186-2 includes the elliptic curve digital signature algorithm(ECDSA),with recommendations on the selection of finite fields and elliptic curves. These recommended curves have special properties that allow for optimized performance. They are also checked to ensure that none of them belong to the class of super singular and anomalous curves, which are susceptible to MOV and other known attacks. The following considerations were made when choosing the finite fields and elliptic curves.

6.3.1. Choice of Key Lengths

All curves are chosen to have cofactors 1, 2 or 4. This is done to ensure efficiency in computation. As a result, private and public keys have approximately the same length in bits [11].

6.3.2. Choice of Fields

Each field is chosen such that the length of the order in bits is at least twice the key length of common private-key (symmetric-key) block ciphers. This is done because an exhaustive key search of a k-bit block cipher is expected to take about the same time as the solution of an ECDLP, when using the Pollard's rho algorithm for a suitable elliptic curve over a finite field with an order of length 2k [58].

Symmetric cipher key length	Example Algorithm	Bit-length of p in prime field Fp	Dimension m of binary field F2m
80	SKIPJJACK	192	163
112	Triple-DES	224	233
128	AES small	256	283
192	AES medium	384	409
256	AES Large	521	571

Table 6-2 compares private-key lengths with sizes of various fields.

6.3.3. Choice of p in GF (p) and m in GF (2^m)

For binary fields GF (2^m), m was chosen so that there exists a Kolbitz curve of almost prime order over GF (2^m). For` prime fields GF(p), p was chosen to be either a Mersenne prime, or a Mersenne-like prime with bit size being a multiple of 32 [54]. A Mersenne prime is a prime of the form 2^n-1, where n is a prime.

6.3.4. Performance Analysis

In this section, software implementations of the DSA and RSA digital signature schemes will be compared to ECDSA and the ElGamal encryption scheme will be compared to its elliptic curve counterpart.

Curves: Random curves over GF (2^{191}) and F_p =1,Platform : Pentium Pro 200 MHZ using C, C ++ and Assembly language **PCs**

System	Key Generation	Signature	Verification	Total Time
ECDSA-F^{192}	11.7	11.3	60	83
ECDSA-$F_{p=192}$	5.5	6.3	26	37.8
RSA-1024	1(Sec)	43.3	0.65	1,043.05
DSA-1024	22.7	23.6	28.3	74.6

Table 6-3. Curves: Random curves over GF (2^{191}) and F_p =1

Curves: Kolbitz and random curves over NIST curves GF(2163), GF(2233),GF(2183) Kolbitz Curves

Platform: Pentium II 400 MHZ, using C

System	**Key Generation**	**Signature**	**Verification**	Total Time
ECDSA-F_2^{192}	1.47	2.11	4.09	7.67
System	**Key Generation**	**Signature**	**Verification**	Total Time
ECDSA-F_2^{233}	3.11	4.03	7.87	15.01
ECDSA-F_2^{283}	4.50	5.64	11.46	21.6

Table 6-4. Kolbitz and Random curves

Random Curves

System	Key Generation	Signature	Verification	**Total Time**
ECDSA-F_2^{192}	2.12	2.64	6.46	**11.2**
ECDSA-F_2^{233}	4.58	5.52	14.08	**24.18**
ECDSA-F_2^{283}	6.88	8.08	21.15	**36.11**

Table. 6-5. Random Curves

RSA

System	Key Generation	Signature	Verification	Total Time
RSA-1024	2740.87	66.56	3.86	2811.29
RSA-2048	26,442.04	440.69	13.45	26896.18

Table 6-6. RSA Key Generation and Verification

DSA

System	Key Generation	Signature	Verification	Total Time
DSA-768	14,735	15.55	26.13	1,4776.48
DSA-1024	54,674	24.28	47.23	5.9421.28

Table 6-7.DSA Key Generation and Verification

The results from Tables 6-3 to 6-7 clearly show that Kolbitz curves produce more efficient computation speeds compared to RSA and DSA, and together with the NIST recommended parameters provided in FIPS 186-2, it proves to be a far superior digital signature scheme in terms of efficiency.

Curves: Kolbitz curves over GF(2^{163}) Table 6-8. Kolbitz curves over GF(2^{163})

Key Generation and verification

Kolbitz Curves

System	Key Generation	Signature	Verification	Total Time
ECDSA-F_2^{163}	590	800	2340	3730
RSA-512	360(Sec)	5100	310	3,65410

Table .6-9. Kolbitz Curves Key

System	Key Generation	Signature	Verification	Total Time
ECDSA-F_2^{192}	751	1,011	1,826	3,588
ECDSA-F_2^{233}	1,552	1,910	3,701	7,163
ECDSA-F_2^{283}	2,3691	2,760	2,760	10,614

Generation and Verification

Random Curves

System	Key Generation	Signature	Verification	Total Time
ECDSA-F_2^{192}	1,085	1,335	3,243	5,663
ECDSA-F_2^{233}	2,478	3,066	7,321	12,865
ECDSA-F_2^{283}	3,857	4,264	11,587	19,708

Table.6-10. Random curves Key Generation and Verification

RSA ($e = 2^{16} + 1$)

System	Key Generation	Signature	Verification	Total Time
RSA-1024	580,405	15,889	1,008	597,302
RSA-2048	-	111,956	3,608	-

Table. 6-11. RSA ($e = 2^{16} + 1$) Key Generation and Verification

DSA

System	Key Generation	Signature	Verification	Total Time
DSA-768	-	6,031	11,594	-
DSA-1024	-	9,529	18,566	-

Table. 6-12. DSA Key generation and Verification.

From Tables 6-3 to 6-7 and 6-8 to 6-12, it is clear that the verification process for ECDSA is slower than that of RSA, but the total timing for all three procedures (key generation, signature and verification) of the ECDSA is much faster than RSA and DSA. For an ECDSA key size of 163 over Kolbitz curves, it takes a total of 3,588 ms to complete all three procedures, while it takes RSA 597,302 ms, for a key size of 1024. As for DSA, the verification process alone is longer than the total time required for ECDSA.

6.3.5 .Comparisons.

It is clear from the results of Tables 6-i , 6 -iv that NIST recommended finite fields and Kolbitz curves offer superior performance compared to random curves and fields not in FIPS186-2. On PCs, ECDSA outperforms RSA and DSA significantly in overall timing. Although its verification process is slightly slower, a 2-3 ms difference is negligible on the PC. On the Palm Pilot, using a 1024-bit RSA key was unpractical. Thus a 512-bit key was chosen instead. Compared to the 163-bit ECDSA key, ECDSA showed better overall performance.

The verification process, however, was much slower than the 512-bit RSA. To offset this imbalance, a combination of RSA and ECDSA protocols could be used. For details, refer to [17]. On pagers, the overall timing for ECDSA is much better than RSA and DSA. In conclusion, ECDSA is clearly the most suitable PKC scheme in constrained environments.

6.4. Encryption Schemes

In this section, the ElGamal encryption and decryption algorithms will be compared to its elliptic curve version. Thus, a 768-bit conventional ElGamal should be compared to a 151-bit ECC ElGamal, while a 1024-bit conventional ElGamal should be compared to a 173-bit ECC-ElGamal. However, for key sizes of 151 and 173 bits on the ECC ElGamal, there does not exist trinomial in polynomial bases (PB) nor optimized normal basis in normal bases (NB) [14], hence 155 and 183-bit key sizes will be used instead. That there is s light improvement in security levels in the ECC versions of ElGamal.

System	Encryption	Decryption
ELGamal-768	13,100	6,640
ECC ElGamal-	248	123
ECC ElGamal-	300	139
System	Encryption	Decryption
ELGamal-1024	29,780	15,230
ECC ElGamal-	357	179
ECC ElGamal-	460	212

Table.6-13. Security levels in ECC versions of Elgamal

6.4.1. Comparisons

Elliptic curve operations in NB run approximately 22% faster than PB for encryption, while decryption in NB is approximately 15% faster than PB. Overall, the performance of EC ElGamal is much better than conventional ElGamal; the improvement in overall speed is over 50% [14].

6.4.2. A Survey of Current ECC Applications

ECC can be broadly divided into four categories: the Internet, smart cards, PDAs and PCs.

Smart cards are one of the most popular devices for the use of ECC. Many manufacturing companies are producing smart cards that make use of elliptic curve digital signature algorithms. These manufacturing companies include Phillips, Fujitsu, MIPS Technologies and Data Key, while vendors that sell these smart cards include Funge Wireless and Entrust Technologies. Smart cards are very flexible tools and can be used in many situations. For example, smart cards are being used as bank (credit/debit) cards, electronic tickets and personal identification (or registration) cards.

6.5. EKR Cryptosystem Resistance Against To Simple Power Analysis Attacks.

6.5.1. Introduction

Elliptic curve cryptosystems are more efficient and secure than the conventional cryptosystems like RSA cryptosystems. We developed a Secret Key in the EKR Modified Montgomery Inversion Algorithm to eliminate the number of Iterations of the main loop directly leaks the value of f and also the attacker can not guess the Secret key (t) to retrieve the valuable information in smart cards and

mobile devices. We want to develop the new cryptosystem based on EKR Modified Montgomery Inversion Algorithm to resistance against Simple Power Analysis Attacks in Side-channel Attacks in Elliptic Curve Cryptosystems like RSA Cryptosystems

6.5.2. E K R Cryptosystems with One Public Key and One Private Key to Resistance against To Simple Power Analysis Attacks

We develop a new cryptosystem i.e EKR Cryptosystem to resistance against Simple Power Analysis Attacks with one Public key and one Private Key to resistance against to Simple power Analysis Attacks in Side-Channel Attacks in elliptic curve cryptosystems based on EKR Modified Montgomery inversion algorithm

Algorithm: 6.5.2. EKR Cryptosystem to Resistance against to SPA Attacks

Step (1). Select u as an n-bit prime.

Step (2). Choose t as Secret key as a prime integer and $t \in [1, n]$.

Step (3). Compute w if $2^{w-2} \leq t \leq 2^{w} - 1$ and $1 \leq w \leq n$.

Step (4). Compute m = dw

Step (5). Select a number b such that $(b, 2^{m}) = 1$

Step (6). Compute b^{-1} such that $bb^{-1} \equiv 1 (\text{mod } 2^{m})$.

Step (7). Choose Public key = (u, b). Choose Private Key = (u, b^{-1})

Step (8). Now we want to encrypt the message M = 8 we have

$$C = M^{E} (\text{mod } N)$$

$$C = M^{b} \; b^{-1} \; 2^{m} (\text{mod } u)$$

C = Cipher text

Step (9). To decrypt the cipher text we have

$$M = C^{D} (\text{mod } N)$$

$$M = C^{b-1} \; b^{-1} \; 2^{m} (\text{mod } u), \quad M = \text{Plain Text}$$

6.5.2.1. Case Study 1. Performance Analysis of EKR Cryptosystem with one Public Key and one Private Key to Resistance against to Simple Power Analysis Attacks.

Step (1). Select u as a prime integer = 17.

Step (2). Select t is secret key as a prime integer = 5.

Step (3). Compute w if $2^{w-2} \leq t \leq 2^{w} - 1$ and $1 \leq w \leq n$.

Step (4). Compute m = dw; m=6.

Step (5). Select a number b such that $(b, 2^{m}) = 1 \Longrightarrow b = 7$.

Step (6). Compute b^{-1} such that $bb^{-1} \equiv 1 (\text{mod } 2^{m}) \Longrightarrow b^{-1} = 55$.

Step (7). Public Key = (u, b)

= (17, 7).

Private Key = (u, b^{-1}).

= (17, 55).

Step (8). Encrypt the message M = 8 then

$$C \equiv M^{E} (\text{mod } N)$$

$$C \equiv M^{b} b^{-1} 2^{m} (\text{mod } u)$$

$$C \equiv 8^{7} * 55 * 64 (\text{mod } 17).$$

C = 16. Cipher Text = 15.

Step (9). To decrypt the cipher text as follows

$$M = C^{D} (\text{mod } N)$$

$$M = C^{b^{-1}} b^{-1} 2^{m} (\text{mod } u),$$

$$M = 15^{55} * 55 * 64 (\text{mod } 17)$$

M = 8 , Plain Text = 8.

6.2.5.1. Case Study I. Performance Analysis of RSA Cryptosystems with one Public Key and one Private Key

Step (1). Select two prime integers p = 17. q = 5

Step (2). Compute n = pq = 85, Φ (n) = (p-1)(q-1) = 64

Step (3). Choose e=5, Check gcd (5, p-1) = gcd (5, 16) = 1

Check gcd (5, q-1) = gcd (5, 4) =1

Check gcd (e, (p-1) (q-1)) = gcd(5, 64) = 1

e = 5.

Step (4). Compute d such that ed =1(mod Φ (n))

Step (5). Compute $d = e^{-1} \pmod{\Phi(n)}$

$d = 5^{-1} \pmod{64}$

Find a unique value d such that 64 divides 5d-1 value ---- pi

d = 13

Step (6). Public Key = (n, e) = (85, 5).

Private Key = (n, d). = (85, 9).

Step (7). Encrypt the message M = 8 then

$C \equiv M^{E} \pmod{N}$

$C \equiv 8^{5} \pmod{85} = 43$, Cipher Text = 43

Step (8). To decrypt the cipher text we have $M = C^{d} \pmod{n}$

$M = 43^{13} \pmod{85}$

M = 8, Plain Text = 8

6.5.2.2. Case Study I1. Performance Analysis of EKR Cryptosystem with one Public Key and one Private Key to Resistance against to Simple Power Analysis Attacks.

Step.(1). Select u as a prime integer = 13.

Step.(2). Select t is secret key as a prime integer = 5.

Step (3). Compute w if $2^{w-2} \leq t \leq 2^{w} - 1$ and $1 \leq w \leq n$.

Step (4). Compute m = dw; m=6.

Step (5). Select a number b such that $(b, 2^{m}) = 1 \Rightarrow b = 7$.

Step (6). Compute b^{-1} such that $bb^{-1} \equiv 1 (\text{mod } 2^{m}) \Rightarrow b^{-1} = 55$.

Step (7). Public Key = (u, b)

= (13, 7).

Private Key = (u, b^{-1}).

= (13, 55).

Step (8). Encrypt the message M = 8 then

$$C \equiv M^{E} (\text{mod } N)$$

$$C \equiv M^{b} b^{-1} 2^{m} \ (\text{mod } u)$$

$$C \equiv 8^{7} * 55 * 64 \ (\text{mod } 13).$$

C = 2097152 X 55 X 64(mod 13) = 11, Cipher Text = 11.

Step (9). To decrypt the cipher text as follows

$$M = C^{D} (\text{mod } N)$$

$$M = C^{b-1} \ b^{-1} 2^{m} \ (\text{mod } u),$$

$$M = 11^{55} * 55 * 64 (\text{mod } 13)$$

$$= 6.654881814988989267387731337 7 \text{ X } 10^{30}$$

M = 8 , Plain Text = 8.

6.5.2.2. Case Study II. Performance Analysis of RSA Cryptosystems with one Public Key and one Private Key.

Step (1). Select two prime integers p = 13. q = 5

Step (2). Compute n = pq = 65, $\Phi (n) = (p-1)(q-1) = 48$

Step (3). Choose e=5, Check gcd (5, p-1) = gcd (5, 12) = 1

Check gcd (5, q-1) = gcd (5, 4) =1

Check gcd (e, (p-1) (q-1)) = gcd(5, 48) = 1

e = 5.

Step (4). Compute d such that ed =1(mod Φ (n))

Step (5). Compute $d = e^{-1} \pmod{\Phi(n)}$

$d = 5^{-1} \pmod{48}$

Find a unique value d such that 64 divides 5d-1 value ---- pi

d = 10

Step (6). Public Key = (n, e) = (65, 5).

Private Key = (n, d). = (65, 10).

Step (7). Encrypt the message M = 8 then

$C \equiv M^{E} \pmod{N}$

$C \equiv 8^{5} \pmod{65} = 43$, Cipher Text = 43

Step (8). To decrypt the cipher text we have $M = C^{d} \pmod{n}$

$M = 43^{10} \pmod{65}$

M = 64, Plain Text = 64. **** Wrong ****

6.5.2.3. Performance Comparisons

In the existing system experiments were performed on both PCs and mobile devices. Data was collected from various studies conducted by research institutes and individual experiments. Elliptic curve cryptosystems provides more security and efficiency for mobile devices and PC's but also it is alternative to conventional cryptosystems like RSA and DSA.

In the proposed EKR Cryptosystem to resistant against to Simple Power Analysis Attacks was most suitable for side-channel attacks in elliptic curve cryptosystems. Because secret key "t" was infeasible in cryptosystem and also the generation of public Key and Private Key are infeasible, so the attacker unable to guess the secret key in the cryptographic operation. The attacker could not measure

the power consumption using the execution of a cryptographic algorithm store the wave form using oscilloscope and process the information to learn the secret key.

The generation of secret key and Private Key was not easy to guess to retrieve the valuable information from single leaked information in a power consumption electromagnetic emanation trace in Simple Power Analysis Attacks. Finally the proposed application i.e. EKR cryptosystem to resistance against to Simple power Analysis attacks most suitable for mobile devices than the conventional cryptosystems like RSA.

6.6. Conclusions

ECC is the most for suitable PKC schemes use in a constrained environment. Its efficiency and security makes it an attractive alternative to conventional cryptosystems, like RSA and DSA, not just in constrained devices, but also on powerful computers.

It is no doubt about ECC was being recognized as a fast powerful cryptographic scheme. We provided a new cryptosystem i.e EKR Cryptosystem to resistance against to Simple Power Analysis Attacks and more efficient to conventional cryptosystems like RSA. Wireless devices are rapidly becoming more dependent on more security features such as the ability to do secure email, web browsing and virtual private networking to corporate networks and ECC allows more efficient implementation of all of these features.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

7.1. Conclusions

In this thesis we dealt with key randomization countermeasures to simple power analysis attacks. Mathematically **we proved that the proof for probability of occurrence of minor collisions in the second splitting scheme**. And also we proposed the system for the effect of the minor collisions on the fixed sequence window method.

According to the existing system it is **not proved the probability of minor collisions** increased or reduced of that major collision depends on the value of the least significant bits of jw length. It has been **proved that mathematically the probabilities of minor collisions are reduced up to 50% of major collisions.**

We discovered a new algorithm i.e **"EKR Modified Montgomery Inversion Algorithm"** and introduces to a secret key t to eliminate the number of Iterations of the main loop directly leaks the value of f and also it is mathematically proved that f is uniformly distributed with a significant reduction. So the attacker can not guess the key (t) to retrieve the valuable information in smart cards and mobile devices.*Performance considerations in EKR Modified Montgomery Inversion Algorithm are faster than when using Fast exponentiation of the above system has been presented.*

A new cryptosystem i.e EKR Cryptosystem to resist against to Simple Power Analysis Attacks and more efficient to conventional cryptosystems like RSA systems has been discovered.

7.2. Future Work

The computational task is to compute a product [K]P where P is a point on an elliptic curve E(Fq) over a Finite Field and K is a secret positive integer. Point P is usually not secret, and indeed may often be chosen by the attacker. Multiplier K may either be ephemeral or serve as a long-time key.

It is interesting to pursue to develop the system is to **send the information with multiple users through Secret Key (t) such as messages, Communications and allows Virtual Private Networks in Mobile devices.**

Summary of Major Contributions

1. Introduction To Elliptic Curves And Elliptic Curve Cryptography was published in the Journal of Computer Science May-June 2009
2. Elliptic Curve Cryptography and Its Applications were published in the proceedings of International conference on Photonics, Nanotechnology and Computer Applications -Prist University Thanjavur Vol No. 2 Feb 2009 page 215.
3. Elliptic Curve Cryptosystems And Side-Channel Attacks was accepted to be publish in the International Journal of Network Security , Vol.12.No.2, P.136-146, Mar.2011
4. A New Variant Nevine Maurice Ebied's Key Randomization Counter Measures to Power Analysis Attacks on Elliptic Curve Cryptosystems was published in International Journal of Computer Science and Network Security Volume 9 No-2 February 2009 Pages 446-455.
5. A Paper entitled "Security through Elliptic Curves for Wireless Networks in Mobile Devices "published in Lecture Notes in Computer Science in vol.1 pp 208-213, World Congress On Engineering and Computer Science 2010, San Francisco, USA.
6. A paper entitled "A Study of Intrusion Detection in Data mining" published in Directory of Open Access Journals Lecture Notes in On Engineering and Computer Science, Scopus Index, ISSN: 2078-0966 in Volume 2192, Issue :1, pp 1889-1894, July 2011
7. A paper entitled "E. Kesavulu Reddy cryptosystem to Resistance against Simple Power analysis Attacks in mobile Devices" published in Directory of Open Access Journals , Lecture notes on Engineering and Computer Science, Scopus Index, ISSN :2078-0958 in Vol.2197, Issue:1, pp 540-544, July 2012.

REFERENCES

1. Agrawal D., B. Archambeault, J. R. Rao & P. Rohatgi. The EM Side- Channel(s) Attacks and Assessment Methodologies. Internet Security Group,IBM Watson search Center.
2. Anderson R Eli Biham, and Lars Knudsen. Serpent: A proposal for the Advanced Encryption Standard. In First Advanced Encryption Standard (AES) Candidate Conference, National Institute of Standards and Technology (NIST), 1998.
3. Awasthi A. K. and S. Lal, \ID-based Ring Signature and Proxy Ring Signature Schemes from Bilinear Pairings," International Journal of Network Security, vol. 4, no. 2, pp. 187-192, Sept. 2007.
4. Alexandridis. Georgios C. Artemios G. Voyiatzis, and Dimitrios N. Sopranos Crypto Palm: A Cryptographic Library for Palm OS "Pp 1-10 University of Patras, GR-26504, Patras Greece.
5. BAYAM. Keklik ALPTEKIN, Berna ORS " Differential Power Analysis Resistant Hardware Implementation of the RSA Cryptosystem ", Turk J Elec Eng & Computer Science, Vol.18, No.1, 2010, TUBITAK.
6. Billet. Ollivier and Marc Joye The Jacobi Model of an Elliptic Curve and Side Channel Analysis, Applied Algebra, Algebraic Algorithms and Error Correcting Codes, Vol 2643 of Lecture Notes in Computer Science, pp 43-42, Springer Verlag, 2003.
7. Blake. I. F. G. Seroussi, and N. P. Smart, "Elliptic Curves in Cryptography," London Mathematical Society Lecture Note Series, 265, Cambridge Univ. Press, 2000.
8. Brumley and. Billy Bob, Nicola Tuveri Remote Timing Attacks are Still Practical? Aalto University School of Science, Finland
9. Boneh, D. and Daswani, N. "Experimenting with electronic commerce on the PalmPilot". In proceedings of Financial Cryptography '99. Lecture Notes in Computer Science, vol. 1648. p 1 – 16 Springer-Verlag Heidelberg, 1999.

10. Boneh, D. ,R. A. DeMillo & R. J. Lipton. "On the importance of eliminating errors in cryptographic computations". Journal of Cryptology, pp 14:101–119, 2001 EUROCRYPT '97.
11. Boneh, D. and M. Franklin, "Identity-Based Encryption from the Weil Pairing," J. Kilian, Ed., Advances in Cryptology — CRYPTO, 2001, LNCS, vol. 2139, , pp. 213–29, Springer-Verlag, 2001.
12. Brown *et al* .M., "Software Implementation of the NIST Elliptic Curves over Prime Fields," D. Naccache, Ed., Topics in Cryptology — CT-RSA 2001, LNCS, vol.2020, pp 250–65 .Springer-Verlag, 2001, pp. 250–65.
13. Brown, M., Hankerson, D., Lopez, J., and Menezes, A. "Software Implementation of the NIST Elliptic Curves over Prime Fields" In proceedings of Cryptographer's Track at RSA Conference 2001,SanFrancisco,Lecture Notes in Computer Science, vol. 2020, pp 250 – 265, Springer-Verlag Heidelberg, January 2001:
14. Certicom Press Release. "Certicom Announces Elliptic Curve Cryptosystem Challenge Winner". November 6, 2002.
15. Certicom Research. Standards for efficient cryptography {SEC 1: Elliptic curve cryptography. Version 1.0, 2000.
16. Chari. S, C. S. Jutla, J. R. Rao & P. Rohatgi. "Towards sound approaches to Counteract power-analysis attacks." In Advances in Cryptology – CRYPTO 99, LNCS, vol. 1666, pp. 398–412. Springer-Verlag, 1999. 24.
17. Chevallier-Mames, M. Ciet, and M. Joye. Low-cost Solutions for Preventing Side-Channel Analysis, Side-Channel Atomicity. IEEE Transactions on Computers, 53(6):760–768, 2004.
18. Chudnovsky D. V, and G. V. Chudnovsky, Sequences of numbers generated by addition in formal groups and new primality and factorization tests, Advances in *Applied Maths. 7 (1986/7), 385{434.*

19. Ciet. M. Aspects of Fast and Secure Arithmetic are for Elliptic Curve Cryp tography. Ph.D. thesis, pp 120, 170, 171, 180, Universities Catholiquede Louvain, 2003.
20. Ciet. M. & M. Joye. "(Virtually) free randomization techniques for elliptic curve Cryptography". In Information and Communications Security – ICICS '03, LNCS, vol. 2836, pp. 348–359. Springer-Verlag, 2003.
21. Ciet. M., J.-J. Quisquater & F. Sica. "Preventing differential analysis in GLV Elliptic curve scalar multiplication". In Cryptographic Hardware and Embedded Systems – CHES '02, LNCS, vol. 2523, pp. 540–550, 4, 25, 126, 173, 179. Springer- Verlag, 2003.
22. Ciet *et al.* M, "Trading Inversions for Multiplications in Elliptic Curve Cryptography," preprint, 2003.
23. Clavier. C & M. Joye. "Universal exponentiation algorithm a first step towards provable SPA-resistance". In Cryptographic Hardware and Embedded Systems CHES '01, LNCS, vol. 2162, pp. 300–308, 4, 24, 120, Springer-Verlag, 2001.
24. Compton. Kevin J, Brian Timmy. Joel VanLavenz, A Simple Power Analysis Attack on the Serpent Key Schedule, Computer Science and Engineering ***Division***, MI 48109-2212, USA, September 24, 2009.
25. Coron. J.-S,"Resistance against differential power analysis for elliptic curve Cryptosystems". In Cryptographic Hardware and Embedded Systems – CHES 99, LNCS, vol. 1717, pp. 292–302., 2, 22, 24, 158, 170, 180, 181, 186, Springer -Verlag, 1999.
26. Daniel M. Gordon, "A Survey of Fast Exponentiation Methods,"Algorithms,Vol 27, No. 1, pp . 129–46 , 1998.
27. ElGamal.T, "A public key cryptosystems and a signature scheme based on Discrete logarithms" IEEE Transactions on Information Theory, 31(4), pp: 469– 472, 1985.
28. Enge, A. Elliptic curves and their applications to cryptography. Kluwer Academic Publishers, 1999.

29. Elliptic Curve Cryptosystem: Remarks on the Security of the Elliptic Curve Cryptosystem, Certicom 1997.

30. ECC Additional Groups for IKE, Mar. 2001.

31. Eisentraeger *et al.* K."Fast Elliptic Curve Arithmetic and Improved Weil Pairing Evaluation," M. Joye, Ed., Topics in Cryptology — CT-RSA 2003, LNCS, vol. 2612, pp. 343–54 Springer-Verlag, 2003.

32. Giacomo de Meulenaer and Francos' Xavier Standeart "Stealthy Compromise Wireless sensor Nodes with Power analysis Attacks" Associate Researcher of the Belgian Fund for Research.

33. Goos. G. and Hartmanis J., editors., "World Wide Number Field Sieve Factoring Record: on to 512 Bits," "Advances in Cryptology Asia crypt 96", Kyongju, Korea, LNCS, vol. 1163, pp. 382–94, Springer-Verlag, 1996.

34. Gupta. V. S. Gupta, and S. Chang, "Performance Analysis of Elliptic Curve Cryptography for SSL,"ACM Workshop Wireless Security, Mobicom 2002, Atlanta, A, Sept. 2002.

35. Guajardo. J, and C.Paar, Efficient Algorithms for Elliptic Curve Cryptosystems," B. S. Kaliski Jr., Ed., Advances in Cryptology, CRYPTO, 1997, LNCS, vol. 1294, pp. 342–56 , Springer-Verlag, 1997.

36. Hiroaki MORIMOTO, Hedeyo MAMIYA, Atsuko MIYAJI, ," Secure Elliptic Curve Exponentiation against RPA, ZRA, DPA and SPA", IEICE TRANS, FUNDAMENTALS, Vol. E-89, pp 2207 – 2215, August 2006.

37. Giraud1. Christopher and Vincent Verneuil2,3⋆ Atomicity Improvement for Elliptic Curve Scalar Multiplication, pp 1-24, inria-00459461, Feb 2010.

38. Hyun Kim. Taee Tsuyoshi Takagi, Dong-Guk Han, Howon Kim and Jongin Lim " Power Analysis Attacks and Countermeasures on η_T Pairing Over Binary Fields", ETRI Journal Volume 30, pp 68 -80, Number 1, February 2008.

39. Hisill. Huseyin Kenneth koon-Ho Wrong, Gary carter and Ed Dawson " Faster Operations on Elliptic Curves" Australasian Information Security Conference(AISC 2009), pp 1099-1016,Wellington, New Zealand, January 2009.
40. Hankerson, D, A. Menezes & S. Vanstone. Guide to Elliptic Curve Cryptography, Springer-Verlag, 2004. 8, 9, 10, 11, 18, 130, 131, 138, 145, 168, 197.
41. Heuberger & H. Prodinger. Personal communication. August, 2003.
42. . Hamacher. V. C Z. G. Vranesic & S. G. Zaky. Computer Organization. Boston: McGraw-Hill, fifth Edition, 136-138, 2002.
43. Higgins. Li, Z, J., and Clement, M. "Performance of finite field arithmetic in an Elliptic curve cryptosystem". Ninth Symposium in Modeling, Analysis and Simulation of Computer and Telecommunication Systems, pp 249 – 258 IEEE Computer Society, 2001: p 249 – 258.
44. Joye. M, Fast Point Multiplication on Elliptic Curves without Precomputation. In J. vonzur Gathen, J. L. Inane, and C K. Koc, editors, Arithmetic of Finite Fields – WAIFI 2008, volume 5130 of Lecture Notes in Computer Science, pages 36–46. Springer, 2008.
45. Joye. M, & K. Villegas. "A Protected Division Algorithm". In Smart Card Research and Advanced Applications – CARDIS '02, pp. 59–68,122,136, Usenix Association, 2002.
46. Joye. M, and Jean-Jacques Quisquater "Hessian Elliptic Curves and Side-Channel Attacks", Lecturer notes in Computer Science, vo l 2162, pp 402-410, Springer Verlag, 2001.
47. Joye.M Elliptic curves and Side channel Analysis , ST Journal of System Research 4(1) pp 283-306, 2003.
48. Johnn Grobschadl " Performance and Security Aspects of Client-side SSL/TLS processing on Mobile Devices " First International Workgroup on Side-Channel Analysis and secure design-COSADE 2010.

49. Koschuch. Manuel , Johnn Grobschadl, Udo Payer and Michael Krigger " Workload Characterization of a Lightweight SSL Implementation Resistant to Side-Channel attacks" LNCS 5339 pp. 549-365, 2008.

50. Kirtane. Varad, C.Pandu Rangan "Side Channel Attack Resistant Implementation of Multi-Power RSA using Hensel Lifting" Dept.of Computer Science IIT-Madras.

51. Kocher. P, "Timing attacks on implementations of Diffie-Hellman, RSA DSS and other systems." In Advances in Cryptology – CRYPTO '96, LNCS, vol. 1109, pp. 104–113. Springer-Verlag, August 1996.

52. Kocher. P, J. Jaffe & B. Jun. "Differential power analysis". In Advances in Cryptology – CRYPTO '99, LNCS, vol. 1666, pp 2, 22, 24, 172, 194. Springer-Verlag, 1999. .

53. Lopez, J. and Dahab, R. "Performance of Elliptic Curve Cryptosystems" Technical report IC- May 2000.

54. Lim. C. H. and P. J. Lee, "More Flexible Exponentiation with Precomputation," Y. G. Desmedt, Ed., Advances in Cryptology — CRYPTO, LNCS, vol. 839, pp. 95–107, Springer-Verlag, 1994.

55. Liardet, P.-Y., and Smart, N. P. Preventing SPA/DPA in ECC systems using the Jacobi form. In Cryptographic Hardware and Embedded Systems C. K. Ko_c, D. Naccache, and C. Paar, Eds., pp. 401-411{ CHES 2001[Pre-]Proceedings (2001).

56. Medwed. Marcel, Franqois- Xavier Standaert, Johann Grobschadi and Francesco Regazzomi "Fresh Re-Keying: Security against Side-Channel and Fault Attacks for Low-Cost devices". LNCS 6055 (2010), pp. 279-296, AFRICACRYPT 2010,

57. Mangard. Stefan, Elisabeth Oswald, and Thomas Popp. Power Analysis Attacks: Revealing the Secrets of Smart Cards (Advances in Information Security). Springer, 2007.

58. Messerges. T. S, E. A. Dabbish & R. H. Sloan. "Investigations of power analysis Attacks on smart cards". In USENIX Workshop on Smart- card Technology, pp. 151– 161, 2, 24, 172, May 1999.

59. Miller. V. S. "Use of Elliptic Curves in Cryptography," H. C. Williams, Ed Advances in Cryptology, CRYPTO, LNCS, vol. 218, pp.417–26, 1985, Springer-Verlag, 1986.

60. Maurer's, and Y. Yacobi, "Non-interactive Public Key Cryptography," Advances in Cryptography Euro crypt '91,LNCS, vol. 547, pp. 498–507, Springer-Verlag, 1991.

61. Menezes, A., Okamoto, T., and Vanstone, S. "Reducing elliptic curve logarithms in a finite field". Proceedings of the twenty-third annual ACM Symposium on Theory of computing. Annual ACM Symposium on Theory of Computing. ACM Press, 1991: p 80 – 89.

62. Menezes, A. J. Elliptic curve public key cryptosystems. Kluwer Academic Publishers, 1993.

63. Menezes, A. J, P. C. van Oorschot & S. A. Vanstone. Hand book of Applied Cryptography. CRC Press, pp 139-46, 1996.

64. Moller. Bodo,"Securing Elliptic Curve Point Multiplication against Side- Channel Attacks Springer Verlag LNCS pp 324-334, Information Security-ISC 2001.

65. Morain F, & J. Olivos. "Speeding up the computations on an elliptic curve Using addition-subtraction chains".Informatique theoriqueet Applications Theoretical Informatics and Applications, 24(6), pp 531–544, 15, 28, 36, 37, 42, 44, 46, 50, 70, 209, 1990.

66. Maurice Ebied 's. Nevine thesis "Key Randomization Counter Measures to Power Analysis Attacks on Elliptic Curve Cryptosystems" pp 124-126, 139-142, University of Waterloo, Canada, 2007.

67. National Institute of Standards and Technology (NIST). Digital Signature Standard. Federal Information Processing Standards Publication (FIPS) 186-2,
January 27 2000.

68. Okeya, K., and Sakurai, K. Power analysis breaks elliptic curve cryptosystems even secure against the timing attack. In Progress in Cryptology{ INDOCRYPT2000 (2000), B. K. Roy and E. Okamoto, Eds., vol. 1977 pp. 178-90 of Lecture Notes in Computer Science.

69. Okeya, K,& T. Takagi. "The width-w NAF method provides small memory and fast elliptic scalar multiplications secure against side channel attacks". In Topics in Cryptology – CT-RSA '03, LNCS, vol. 2612, pp. 328– 343. Springer-Verlag, 2003. 23, 128, 164, 197.

70. Omura, J. and Massey, J. "Computational method and apparatus for finite field Arithmetic", U.S. Patent number 4,587,627, May 1986.

71. Page. D, and F. Vercauteren, \A fault attack on pairing-based cryptography," IEEE Transactions on Computers, vol. 55, no. 9, pp. 1075-1080, 2006.

72. Quan. J& G.Bai, "A DPA-Resistant Digit-Parallel Modular Multiplier over GF (2m)", Proc. of Sixth International Conference on Information Technology: New Generations (ITNG), pp.53-57, 2009.

73. Reitwiesner. G.W, " Binary Arithmetic", Advances in Computers, Vol.1,pp 231- 960, 14,15,31, 1,58,70.

74. Rohatgi. P., D. Agrawal, B. Archambeault, S. Chari & J. R. Rao. "Power, EM and all that: Is your crypto device really secure?" A talk given as part of the 7th Workshop on Elliptic Curve Cryptography – ECC 2003.

75. .Sasaki. A and K.Abe, "Algorithm Level Evaluation of DPA Resistivity against Cryptosystems", IEEJ Trans. on Electronics, Information and Systems, pp.1221-1228, 2006.

76. SAKAI. Yasuyuki and Kouichi SAKURAI On the Vulnerability of Exponent Recodings for the Exponentiation against Side-Channel Attacks, IEICE Trans, Fundamentals, vol-e-88-a, no.1 January 2005.

77. Savas E& C¸ cetin Kaya Koc¸. "The Montgomery modular inverse revisited." IEEE Transactions on Computers, 49(7), pp 138 139, 140.,763–766, 2000.

78. Santosh Ghosh, Debdeep Mukhopadhyay, and Dipanwita Roy Chowdhury Fault Attack and Countermeasures on Pairing Based Cryptography" International Journal of Network Security, Vol.12, No.1. pp.21-28 jan.2011.
79. Satoh, T. and Araki, K. "Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves". Commentarial Mathematic Universitatis Sancti Pauli 47, pp 81 – 92.1998 .
80. Schneier, B. Applied cryptography. John Wiley & Sons, Inc., 1994.
81. Shamir. A. "Identity-Based Cryptosystems and Signature Schemes," Advances in Cryptology: Proc. CRYPTO '84, LNCS, vol. 7, pp. 47–53.,1984,
82. Solinas. J. A, "Efficient arithmetic on Kolbitz curves". Designs, Codes and Cryptography, 19,pp 195–249, , 14, 15, 20, 37, 46, 50, 58, 70, 89, 90, 91. 2000.
83. .Suzuki. D, M. Saeki, and K.Shimizu, "Evaluation of Side-Channel resistance for Block Cipher Architecture", Proc. of Symposium on Cryptography and Information Security, SCIS2010, 1A1-1, 1A1-2, 2010.
84. Th´eriault. N. "SPA resistant left-to-right integer recodings". In Selected Areas In Cryptography – SAC '05, LNCS, vol. 3897, pp. 345–358, 23, 128, 133, 16 Springer- Verlag, 2006.
85. VanLaven.Joel, Mark Brehob, and Kevin J. Compton. A computationally feasible SPA attack on AES via optimized search. In Security and Privacy in the Age of Ubiquitous Computing: 20th International Information Security Conference (SEC 2005), pages 577{588, 2005.
86. Weghenkel. Bjorn, Thesis entitled "Statistical Methods for Side-Channel based Reverse Engineering, July 6, 2009.
87. Wang, Shuo, Fan Zhang, Jianwei Dai, Lei Wang, and Zhijie Jerry Shi, Making Register File Resistant to Power Analysis Attacks, pp 572-582, IEEE,2008.
88. Wang. Y, J.Leiwo, T.Srikanthan, L.Jianwen, "An Efficient Algorithm for DPA-resistant RSA", Proc. of IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), pp.1659-1662, 2006.

89. Yacobi.Y. "Discrete-log with Compressible Exponents," Advances in Cryptology, CRYPTO '90, LNCS, vol. 537, Springer-Verlag, pp. 639–43. K.Yamakoshi and A.Yamagishi, "Estimation of CPA attack for AES using Simulation method", IEICE Technical Report, ISEC2009-3, *pp.13-20, 2009.*

90. Yamamoto. D, T.Ochiai, K.Itoh, M.Takenaka, N.Torii, D.Uchida, T.Nagai, and S.Wakana, "Hybrid Correlation Power Analysis", Proc. of Symposium on Cryptography and Information Security, SCIS2010, 3B1-2, 2010.

91. Yoshikawa. Masaya and Toshiya Asai, High-Level Simulation for Side Channel Attacks, Vol II Proceedings of the International Conference of Engineers and Computer Scientists 2011 IMCES2011, Hong Kong

SECURITY THROUGH ELLIPTIC CURVES FOR WIRELESSNETWORKS IN MOBILE DEVICES

Dr.E.KESAVULU REDDY

KEY FEATURES

- **Probability of Major Collisions in Side Channel Attacks.**
- **Mathematical Proof of the Major Collisions**
- **Programing in Java and Test results of major Collisions in Side Channel Attacks.**
- **Probability of Minor Collisions in Side channel Attacks.**
- **Mathematical Proof of the Minot Collisions**
- **Programming in Java and Test results of Minor Collisions in Side channel Attacks.**
- **The Effect of Probability of Minor Collisions on Fixed Sequence Window**
- **E. KESAVULU REDDY Modified Montgomery Inverse Algorithm resistant to Simple Power Analysis Attacks.**
- **Mathematical Proof of the E. KESAVULU REDDY Modified Montgomery Inverse Algorithm resistant to Simple Power Analysis Attacks.**
- **Programming in Java and Test result of E. KESAVULU REDDY Modified Montgomery Inverse Algorithm resistant to Simple Power Analysis Attacks.**
- **Performance analysis between EKR Modified Montgomery Inversion Algorithm and Fast Exponentiation**
- **Applications of elliptic curves in identity based encryption that may help to launch deployment of a public key infrastructure.**
- **E.KESAVULU REDDY Cryptosystem to resistance against Simple Power Analysis Attacks**

Dr.E.Kesavulu Reddy working as an Assistant Professor from department of Computer Science, Sri Venkateswara University College of Commerce Management and Computer Science, Tirupati (AP)-India. He received Master of Computer Applications from S.V.University, Tirupati, Master of philosophy in Computer Science from Madurai Kamraj, Madurai, Tamilnadu and Doctorate in Computer Science from S.V.University, Tirupati, Andhra Pradesh- India. He was over 10 years in teaching and developed a Probability and analytical result of Minor and major Collisions in Side channel Attacks, E. Kesavulu Reddy Modified Montgomery Inverse Algorithm and a Cryptosystem i.e. E.KESAVULU REDDY Cryptosystem resistance against to Simple Power Analysis Attacks.

SECURITY THROUGH ELLIPTIC CURVES FOR WIRELESSNETWORKS IN MOBILE DEVICES

Dr.E.KESAVULU REDDY

KEY FEATURES

- **Probability of Major Collisions in Side Channel Attacks.**
- **Mathematical Proof of the Major Collisions**
- **Programing in Java and Test results of major Collisions in Side Channel Attacks.**
- **Probability of Minor Collisions in Side channel Attacks.**
- **Mathematical Proof of the Minot Collisions**
- **Programming in Java and Test results of Minor Collisions in Side channel Attacks.**
- **The Effect of Probability of Minor Collisions on Fixed Sequence Window**
- **E. KESAVULU REDDY Modified Montgomery Inverse Algorithm resistant to Simple Power Analysis Attacks.**
- **Mathematical Proof of the E. KESAVULU REDDY Modified Montgomery Inverse Algorithm resistant to Simple Power Analysis Attacks.**
- **Programming in Java and Test result of E. KESAVULU REDDY Modified Montgomery Inverse Algorithm resistant to Simple Power Analysis Attacks.**
- **Performance analysis between EKR Modified Montgomery Inversion Algorithm and Fast Exponentiation**
- **Applications of elliptic curves in identity based encryption that may help to launch deployment of a public key infrastructure.**
- **E.KESAVULU REDDY Cryptosystem to resistance against Simple Power Analysis Attacks**

Dr.E.Kesavulu Reddy working as an Assistant Professor from department of Computer Science, Sri Venkateswara University College of Commerce Management and Computer Science, Tirupati (AP)-India. He received Master of Computer Applications from S.V.University, Tirupati, Master of philosophy in Computer Science from Madurai Kamraj, Madurai, Tamilnadu and Doctorate in Computer Science from S.V.University, Tirupati, Andhra Pradesh- India. He was over 10 years in teaching and developed a Probability and analytical result of Minor and major Collisions in Side channel Attacks, E. Kesavulu Reddy Modified Montgomery Inverse Algorithm and a Cryptosystem i.e. E.KESAVULU REDDY Cryptosystem resistance against to Simple Power Analysis Attacks.