On Evaluation and Computation of Novel Short Weierstrass Elliptic Curves for Random Number Generation in Kernel Applications

A thesis submitted to Bharathidasan University in fulfillment of the thesis requirement for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

by

Kunal Abhishek

[Ref. No. 5990/Ph.D.K3/Computer Science/Part Time/ July 2017]

Under the supervision of

Prof. (Dr.) E. George Dharma Prakash Raj

பாரதிதாசன் பல்கலைக்கழகம்
Bharathidasan University
Tiruchirappalli – 620024, Tamilnadu, India

27 January, 2022

School of Computer Science, Engineering & Applications
Bharathidasan University
Tiruchirappalli – 620024
Tamilnadu, India

Dr. E. George Dharma Prakash Raj
Associate Professor

CERTIFICATE

Certified that the work reported in this thesis entitled "**On Evaluation and Computation of Novel Short Weierstrass Elliptic Curves for Random Number Generation in Kernel Applications**" is based on the bonafide work done by **Mr. Kunal Abhishek** under my guidance in the School of Computer Science, Engineering & Applications, Bharathidasan University, Tiruchirappalli - 620 024, during the period 2017-2022 and has not been included in any other thesis submitted previously for the award of any degree.

Tiruchirappalli - 620 024
27 January, 2022

[E. GEORGE DHARMA PRAKASH RAJ]
SUPERVISOR

Kunal Abhishek
Research Scholar
School of Computer Science, Engineering & Applications
Bharathidasan University
Tiruchirappalli – 620023

DECLARATION

Declared that the work presented in this thesis is based on the original work done by me under the kind guidance of **Dr. E. George Dharma Prakash Raj**, Associate Professor, School of Computer Science, Engineering & Applications, Bharathidasan University, Tiruchirappalli - 620 024, during the period 2017-2022 and has not been included in any other thesis submitted previously for the award of any degree.

Tiruchirappalli - 620 024 **[KUNAL ABHISHEK]**

27 January, 2022 RESEARCH SCHOLAR

Curiginal

Document Information

Analyzed document	01,Kunal Abhishek_PhD Thesis_INTRODUCTION AND SURVEY.pdf (D126350887)
Submitted	2022-01-28T03:59:00.0000000
Submitted by	Dr.E.George Dharma Prakash Raj
Submitter email	georgeprakashraj@yahoo.com
Similarity	9%
Analysis address	georgeprakashraj.bdu@analysis.urkund.com

Sources included in the report

SA	**Bharathidasan University, Tiruchirappally / Kunal Rest Part of Thesis.pdf** Document Kunal Rest Part of Thesis.pdf (D125629229) Submitted by: georgeprakashraj@yahoo.com Receiver: georgeprakashraj.bdu@analysis.urkund.com	12
SA	**Bharathidasan University, Tiruchirappally / Kunal Rest Part of Thesis.pdf** Document Kunal Rest Part of Thesis.pdf (D125630202) Submitted by: georgeprakashraj@yahoo.com Receiver: georgeprakashraj.bdu@analysis.urkund.com	24
W	URL: https://gabicast.com/zkz48rz/use-of-elliptic-curves-in-cryptography.html Fetched: 2022-01-28T03:59:27.2370000	1
SA	**Final Thesis Manoj Ranjan Mishra Feb 2017.pdf** Document Final Thesis Manoj Ranjan Mishra Feb 2017.pdf (D30147404)	1
SA	**Elliptic Curves with Applications in Cryptography.pdf** Document Elliptic Curves with Applications in Cryptography.pdf (D53985275)	1
SA	**2039373c.pdf** Document 2039373c.pdf (D18067019)	1
W	URL: https://www.cse.iitk.ac.in/users/nitin/courses/WS2010-ref2.pdf Fetched: 2019-10-19T20:21:43.3700000	1

Curiginal

Document Information

Analyzed document	Kunal Rest Part of Thesis.pdf (D126352926)
Submitted	2022-01-28T04:38:00.0000000
Submitted by	Dr.E.George Dharma Prakash Raj
Submitter email	georgeprakashraj@yahoo.com
Similarity	7%
Analysis address	georgeprakashraj.bdu@analysis.urkund.com

Sources included in the report

SA **Bharathidasan University, Tiruchirappally / Introduction.pdf** 16
Document Introduction.pdf (D125627662)
Submitted by: georgeprakashraj@yahoo.com
Receiver: georgeprakashraj.bdu@analysis.urkund.com

SA **Bharathidasan University, Tiruchirappally / Kunal Review of Literature.pdf** 3
Document Kunal Review of Literature.pdf (D125628589)
Submitted by: georgeprakashraj@yahoo.com
Receiver: georgeprakashraj.bdu@analysis.urkund.com

SA **Bharathidasan University, Tiruchirappally / 01,Kunal Abhishek_PhD Thesis_INTRODUCTION AND SURVEY.pdf** 22
Document 01,Kunal Abhishek_PhD Thesis_INTRODUCTION AND SURVEY.pdf (D126350887)
Submitted by: georgeprakashraj@yahoo.com
Receiver: georgeprakashraj.bdu@analysis.urkund.com

W URL: https://sciendo.com/pdf/10.2478/cait-2021-0020 3
Fetched: 2022-01-28T04:40:03.0030000

SA **Bharathidasan University, Tiruchirappally / KunalAbhishek_PhD Thesis_Final_27Jan2022.pdf** 4
Document KunalAbhishek_PhD Thesis_Final_27Jan2022.pdf (D126270078)
Submitted by: bdulib@gmail.com
Receiver: bdulib.bdu@analysis.urkund.com

SA **CPE-21-1177_Proof_hi.pdf** 2
Document CPE-21-1177_Proof_hi.pdf (D112830945)

W URL: https://yogyui.tistory.com/entry/R-%ED%9A%8C%EA%B7%80%EB%B6%84%EC%84%9D-%EB%AA%A8%EB%8D%B8-%EC%84%B1%EB%8A%A5%ED%8C%90%EB%8B%A8-RMSE-MAE-R-square 3
Fetched: 2022-01-20T16:52:46.9470000

W URL: https://amp.doubtnut.com/question-answer/a-sequence-b0b1b2-is-defined-by-letting-b05-and-bk4-bk-1-for-all-natural-number-k-show-that-bn5-4n-f-642543360/hindi 1
Fetched: 2022-01-15T16:24:43.4000000

SA **2 Malik Asifa mythesis38.pdf** 2
Document 2 Malik Asifa mythesis38.pdf (D40602364)

Dedicated to my Master and Teachers..

Contents

Preface

Elliptic curve mathematics has been used in construction of cryptographic systems for more than three decades. The discrete logarithm problem induced by an elliptic curve is supposed to offer maximum security per bit key as compared to other legacy primitives such as ElGamal, RSA etc. Moreover, elliptic curve poses fully exponential complexity in solving its discrete logarithm which is popularly known as elliptic curve disrete logarithm problem (ECDLP) and hence, elliptic curve cryptography (ECC) is of interest to modern cryptographic system designers. For strategic applications such as kernel application in particular, the elliptic curves need to be randomly computed to avoid any (intentionally non-disclosed) properties of its coefficients and prime which may get exploited by the attackers using non-disclosed or even futuristic vulnerabilities. Therefore, random approach of computation of elliptic curves is only considered as the the trusted computational method in this thesis and subsequently, the estimates of computational resources to compute elliptic curves over large prime fields randomly are proposed to use them in cryptographic applications.

In addition, as a proof of concept, a novel method of designing cryptographically secure pseudo random number generator using the proposed elliptic curves and clock-controlled linear feedback shift registers (LFSRs) is presented to achieve non-reproducibility of its generated bitstreams for the operating system kernels in addition to other randomness properties. None of the existent kernel Cryptographically Secure Pseudo Random Number Generators

(CSPRNGs) or other CSPRNGs provide non-reproducibility of its generated bitstreams to the date as per literature.

Hence, this thesis covers two broad areas i.e., in the first part, it covers the computational aspects of cryptographically secure Short Weierstrass elliptic curves which are comprehensively discussed in Chapter 4 and Chapter 5 with the proposals of two new elliptic curves whereas in the second part, a novel CSPRNG called KCS-PRNG (CSPRNG for Kernel Applications) using the proposed elliptic curves is presented in Chapter 6 of this thesis. However, Chapter 1 deals with the preliminaries and motivation of the work presented in this thesis whereas Chapter 2 covers the discussion on the survey and related works that were carried out in the Cryptography and Computer Science domains. Chapter 2 also lays the foundation for formulation of seven important research problems with respect to the evaluation and computation of novel Short Weierstrass elliptic curves for their implementation in the proposed KCS-PRNG for kernel applications. These research problems are covered in Chapter 3 of the thesis. Finally, the thesis concludes with future directions in Chapter 7.

Tiruchirappalli

27 January, 2022 **KUNAL ABHISHEK**

Abstract

Elliptic curves were first introduced by H. W. Lenstra in elliptic curve factoring algorithm in 1984. Latter in 1985, Victor S. Miller and Neal Koblitz independently proposed the discrete points of elliptic curve group over a finite field in construction of discrete log cryptosystems. Elliptic curves enable fast and secure public key cryptosystems and exhibit algebraic structures to offer benefits like smaller key sizes and higher cryptographic strength per bit as compared to RSA. The key advantage of elliptic curve cryptosystems is that the discrete logarithm problem induced by elliptic curve (ECDLP) does not have any known sub-exponential algorithm which can break the ECDLP provided that the elliptic curve parameters are chosen carefully. The elliptic curve cryptosystems are much difficult to break albeit easy to implement and hence, they are the popular choices to design modern cryptosystems.

This thesis covers seven research problems related to Short Weierstrass elliptic curves in the first part and their applications in the random number generation used in operating system kernel in the second part respectively. In the first part of the thesis, six research problems with respect to evaluation, computation and trusted security aspects of elliptic curves which are aimed for cryptography are addressed. It is imperative to note that elliptic curves over large prime fields only offer sufficient ECDLP hardness and appropriate symmetric security levels for implementation of cryptosystems. However, the computation of elliptic curves randomly over the desired large prime fields demands reasonably high

computational resources and time. These computational resources are considered in terms of (i) the number of CPU clock cycles and, (ii) the number of attempts or searches made in the security parameter space of the elliptic curve. The estimates of the number of CPU clock cycles helps in determining processor requirements whereas the number of attempts or searches helps to decide the number of CPU cores for speeding up the curve generation process. Hence, for the first time in the literature, two novel statistical estimates of computational resources of elliptic curves are proposed for computation of cryptographically safe elliptic curve randomly over a given prime field size using a standard procedure. The proposed computational resource estimates of elliptic curves help to provide the feasibility of deriving new elliptic curves over very large prime field sizes which additionally solves the problem of reasonably long co-existence of the existing elliptic curve based cryptosystems in presence of the quantum adversaries possessing certain number of the qubits. Apart from this, it is asserted in the thesis that strategic and military grade cryptosystems require only those elliptic curves for cryptographic implemention which are not only secure but also trusted. Hence, two popular *deterministic* and *random* computational approaches of elliptic curves are evaluated from computation, security and trust perspectives. The proposed study asserted that the random approach is preferable over the deterministic approach for computation of elliptic curves aimed for implementation in the cryptosystems for strategic or/and military usage. Thus, asserting the essential trust requirements in the computation of elliptic curve, a new security notion called *trusted security acceptance criteria*

is proposed in the thesis to ensure that the computed elliptic curves are trusted for implementation in cryptosystems. Subsequently, the problem of recommendation of trusted elliptic curves over 256 bit and 384 bit prime field sizes is also solved with the proposal of two new elliptic curves in this thesis which are named as Kunal-George 256 bit first random elliptic curve, in short, KG256r1 and, Kunal-George 384 bit first random elliptic curve, in short, KG384r1 respectively.

The second part of the thesis covers the seventh problem which deals with a critical issue of non-reproducibility of the pseudorandom bitstreams generated by a pseudo random number generator (PRNG) of an operating system kernel which nullifies the scope of predicting any internal state of the PRNG. For the first time in the literature, a concrete mechanism using cryptographically secure and trusted elliptic curves is proposed to address non-reproducibility issue of pseudo random bitstream generation. Subsequently, a novel CSPRNG called as **C**ryptographically **S**ecure **P**seudo **R**andom **N**umber **G**enerator for **K**ernel Applications (**KCS-PRNG**) which generates non-reproducible bitstreams is proposed in the thesis.

Hence, the thesis evaluates the computational approaches and estimates the computational resources of Short Weierstrass elliptic curves aimed for cryptography respectively. The thesis contributes new criteria to derive trusted elliptic curves over large prime field which are used in the novel design of the proposed KCS-PRNG which is proven to be a viable CSPRNG candidate for adoption in the operating system kernels.

Acknowledgements

"The Blind can't be taught of the sun. First, give them vision."

- Satyaprabu

First and foremost, I would like to offer my ever deepest gratitude to my Spiritual Master *His Holiness Baba JaiGuruDev* for all His blessings and Love that He bestowed on this lowly soul. I would like to show my ever deepest gratitude to my Bhaiya *Dr. K S Ganesh* and *Dr. Jayamala Indaje Madam* for all the incomparable Love and care they showered on me throughout and, during my stay with them to write the research articles and complete the thesis on time. Their support and affection are out of words to express here.

I extend my heartfelt gratitude to my Supervisor *Prof. (Dr.) George Dharma Prakash Raj* for mentoring and guiding me in completing the Ph.D. work with its current outcomes. His guidance and iterative revisions in drafting this thesis was of great help to me. My special gratitude to him for his extraordinary care and guidance, which made me complete this thesis in time. I always felt blessed being with him.

It was a delight working with my Doctoral Committee Members *Prof. (Dr.) G Ravi* and *Prof. (Dr.) K Mani* for their continuous monitoring, interactions and their feedbacks during Doctoral Committee meetings to improve the quality of the thesis work. They made me enthusiastic to deliver the best in my Ph.D. course. My heartfelt gratitude to them.

I would like to sincerely thank *Prof. (Dr.) G Gopinath*, Former Head, Computer Science Department and *Prof. (Dr.) M Balamurugan*, Head, Computer Science Department for their general interaction and motivations given to me during my Ph.D. course. I would also like to thank the Administrative Staffs of the University,

especially *Mrs. Sharda* for her kind support whenever I visited her seeking some administrative assistance.

I am indebted to *Dr. P V Anandamohan*, Former Technical Advisor, C-DAC for his extraordinary discussions and guidance that helped to improve my inferior drafts and shaping this thesis in the present form. His subject knowledge and expertise have really helped me to improve the quality of the thesis.

I would like to extend my sincere thanks to *Dr. P K Saxena*, Chairman, Advisory Board, Society for Electronic Transactions and Security (SETS), Chennai and Former Director, SAG, DRDO for his technical comments to formulate the problem statements covered specifically in Chapter 5 of the thesis.

The Ph.D. course was not possible for me without having encouragement from *Dr. S A V Satya Murty*, Former Executive Director, SETS and Former Director, IGCAR, Kalpakkam, Department of Atomic Energy, who motivated me to get enrolled in the Ph.D. course with its current theme of work. My heartfelt gratitude to him.

I would like to extend my special gratitude to *Late Shri Ramasubbu Sir*, *Shri SK Iyer Sir* and his family for their parental like support during my stay in Chennai and in Ph.D. course. Their consistent motivations, encouragements and parental care were magical for me.

I would like to thank *Dr. N Sarat Chandra Babu*, Executive Director and *SETS* for the opportunities given to me to pursue the Ph.D. course, conduct the research and complete the thesis work.

I would like to extend my heartfelt gratitude to *Dr. T R Reshmi*, Scientist, SETS who reviewed my research articles and thesis iteratively and offered great suggestions to improve them. Her discussions, suggestions and encouragements are admirable with a sense of gratitude.

I would like to extend my heartfelt gratitude to *Mr. T Santhosh Kumar* for his

special help in experimentation part of the thesis. His kind support will always be remembered. Thanks dude!

I hearty thank to the esteemed *anonymous reviewers* of my research articles and thesis for their invaluable comments that improved technical presentation and editorial quality of both the research papers and the thesis.

At this point of time, I remember and thank my previous mentors and guides from Weapons and Electronic Systems Engineering Establishment (WESEE), Indian Navy, New Delhi *Commodore S Vombatkere*, *Commodore A Anand*, *Commander (Retd.) Sashwat Raizada*, *Commander D K Singh* and *Commander (Retd.) Surendra Sharma* for introducing and nurturing me into Cryptography and Cyber Security domains and made me to understand the strategic requirements closely.

I would also like to thank *Prof. (Dr.) Saurabh Sen Gupta* (for his occasional technical discussions on elliptic curve cryptography) and *Dr. Arbindan* for their outstanding Ph.D. theses published on Internet, which inspired me to select proper format and style to compile this thesis using Latex.

Last but not the least, I humbly express my deepest gratitude to my *family members* including *Parents*, *Father-in-law*, *Wife* and *Kids* for their kind blessings, well wishes and ever support. My heartfelt thanks to my sister *Dr. Smita* and brother-in-law *Shri. Dilip Kumar* for their ever encouragements and love. My special heartfelt gratitude to my wife *Mrs. Barkha* who took care of my kids *Aditya* and *Samaira* well and tried her best to support me in completing my thesis work on time. *Aditya* and *Samaira* are always a boon to me whom I remember with love and smile ◡̈.

Tiruchirappalli

27 January, 2022 **KUNAL ABHISHEK**

xxx

List of Publications

International Journals (SCI/SCIE/ESCI indexed)

1. **Kunal Abhishek** and E. George Dharma Prakash Raj, *Evaluation of Computational Approaches of Short Weierstrass Elliptic Curves for Cryptography*, Cybernetics and Information Technologies (2021). (DOI: 10.2478/cait-2021-0045)

2. **Kunal Abhishek** and E. George Dharma Prakash Raj, *Computation of Trusted Short Weierstrass Elliptic Curves For Cryptography*, Cybernetics and Information Technologies (2021). (DOI: 10.2478/cait-2021-0020)

3. **Kunal Abhishek** and E. George Dharma Prakash Raj, *Computational Investment in Generation of Elliptic Curves Randomly over Large Prime Fields*, Concurrency and Computation Practice and Experience (2022). (Status: Under Revision)

4. **Kunal Abhishek** and E. George Dharma Prakash Raj, *On Random Number Generation for Kernel Applications*, Fundamenta Informaticae, IOS Press (2022). (Status: Accepted - In press)

IEEE Magazine

5. **Kunal Abhishek** and E. George Dharma Prakash Raj, *Operating System Security: A Short Note*, IEEE India Info. Vol. 14 No. 2 Apr - Jun 2019.

The publications based on the research and contributions of the thesis are depicted as below:

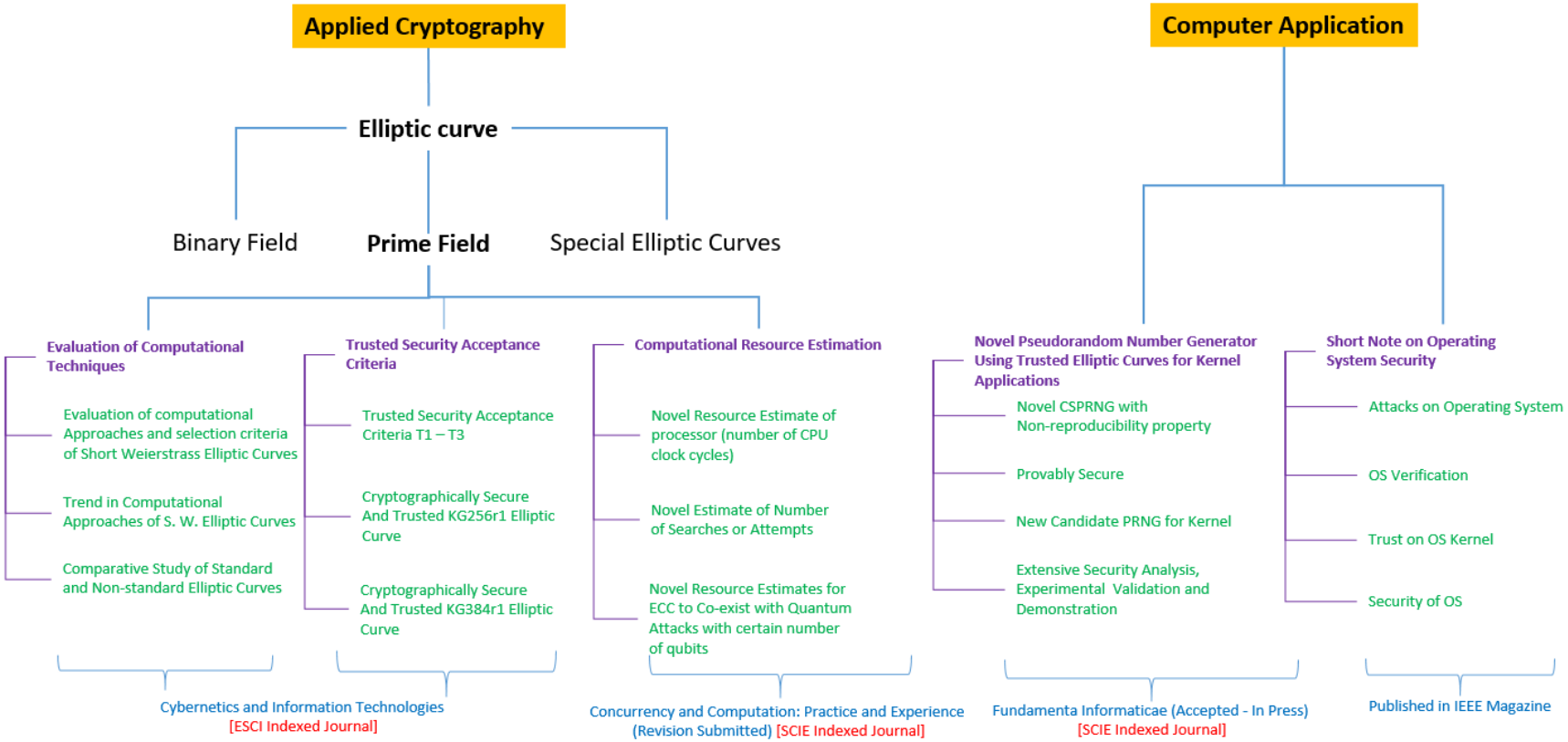


Figure 1: Contribution wise Publication of the Thesis

List of Figures

List of Tables

List of Algorithms

List of Symbols

Symbol	Description
$\mathbb{E}$	Elliptic curve
$\mathbb{E}'$	Twist of elliptic curve
$\mathbb{Z}$	Set of Integers
$\mathbb{E}_c$	Twist of elliptic curve by c, a non-square randomly selected $c \in \mathbb{F}_p$
$\mathbb{F}_q$	Field of characteristic q
$\mathbb{F}_p$	Field of prime characteristic
$\mathbb{F}_{2^n}$	Field of binary characteristic
a	Field element or coefficient of $\mathbb{E}$
b	Field element or coefficient of $\mathbb{E}$
t	Trace of elliptic curve
$\mathcal{O}$	Point at Infinity
h_c	Cross over class number
$h(N)$	Class number of order $h(N)$
$G_{x,y}$	Base point on $\mathbb{E}$
N	Cardinality or Order of $\mathbb{E}$
P	Point P on $\mathbb{E}$
Q	Point Q on $\mathbb{E}$

B	Constant $> 4\sqrt{q}$
μ	Constant
Ψ	A standard procedure to generate random $\mathbb{E}$ over $\mathbb{F}_p$
$\mathcal{K}$	Key Space
$\mathcal{T}$	Look-up Table of elliptic curves
$\mathcal{P}$	Period
$\mathcal{B}$	Bit complexity
$C1$	First cryptographic requirements conditions for elliptic curve
$C2$	Second cryptographic requirements conditions for elliptic curve
$C3$	Third cryptographic requirements conditions for elliptic curve
$R1$	First RNG Requirement
$R2$	Second RNG Requirement
$R3$	Third RNG Requirement
$R4$	Fourth RNG Requirement
k	Embedding degree of elliptic curve
K	Finite field over a prime
r	Correlation Coefficient
R^2	Coefficient of Determination
$R^2{}_{adjusted}$	Adjusted Coefficient of Determination
p-value	Probability value of a test statistic
Θ	Training data set
$\ddot{\Theta}$	Test data set
X_i	i-th variable in X
Y_i	i-th variable in Y
Z_i	i-th variable in Z
β_0	Regression Coefficient called Intercept
β_1	Regression Coefficient called Slope or Linear Effect Parameter

β_2	Regression Coefficient called Quadratic Effect Parameter
ϵ_i	Normally distributed i-th Error
σ^2	Variance
RSD	Residual Sum of Deviations
RSE	Residual Sum of Errors
RSS	Residual Sum of Squares
TSS	Total Sum of Squares
η	Estimate of the number of CPU clock cycles recorded to generate $\mathbb{E}$
ω	Estimate of the number of searches made to generate $\mathbb{E}$
$\triangleright$	Comment

Acronyms

AES	Advanced Encryption Standard
AGM	Arithmetic Geometric Mean
ANSSI	Agence Nationale de la Securite des Systemes d'Information
ASLR	Address Space Layout Randomization
BSGS	Baby Step Giant Step
CC	Clock Cycle
CF	CPU Clock Frequency
CLT	Central Limit Theorem
CM	Complex Multiplication
CPU	Central Processing Unit
CRT	Chinese Remainder Theorem
CSPRNG	Cryptographically Secure Pseudo Random Number Generator
CT	CPU Time
DDR4	Double Data Rate Fourth Generation (RAM)
DES	Data Encryption Standard
DLP	Discrete Logarithm Problem
DRNG	Deterministic Random Number Generator
ECC	Elliptic Curve Cryptography

ECDLP	Elliptic Curve Discrete Logarithm Problem
ECDSA	Elliptic Curve Digital Signature Algorithm
FFT	Fast Fourier Transformation
GCD	Greatest Common Divisor
HID	Human Interface Device
HRNG	Hardware based Random Number Generator
IETF	Internet Engineering Task Force
IV	Initialization Vector
JSON	Java Script Object Notation
JWE	JSON Web Encryption
Kbps	Kilobits per second
KCS-PRNG	Cryptographically Secure Pseudo Random Number Generator for Kernel
KG256r1	Kunal George first 256 bit randomly generated elliptic curve
KG384r1	Kunal George first 384 bit randomly generated elliptic curve
LC	Linear Complexity
LFSR	Linear Feedback Shift Register
LSB	Least Significant Bit
Mbps	Megabits per second
MPQS	Multi Polynomial Quadratic Seive
MOV	Menezes-Okomoto-Vanstone
MSB	Most Signficant Bit
NIST	National Institue of Standards and Technology
NSA	National Security Agency
NUMS	Nothing Upon My Sleeves
OLS	Ordinary Least Squares
OS	Operating System
PC	Personal Comuter

PKI	Public Key Infrastructure
PQC	Post Quantum Cryptography
PRNG	Pseudo Random Number Generator
RFC	Request for Comment
RNG	Random Number Generator
RQP	Random Quantum Polynomial
RSD	Residual Sum of Deviations
RSE	Residual Sum of Errors
RSS	Residual Sum of Squares
SEA	Schoof-Elkies-Atkins
SECG	Standards for Efficient Cryptography Group
SG	Sequence Generator
SSH	Secure Shell
TLS	Transport Layer Security
TRNG	True Random Number Generator
TSS	Total Sum of Squares
XOR	Exclusive OR

Chapter 1

Introduction

> "I have grown to love secrecy. It seems to be the one thing that can make modern life mysterious or marvelous to us. The commonest thing is delightful if only one hides it."
>
> - Oscar Wilde

Cryptographic primitives take advantage of the computationally intractable hard problems such as integer factorization problem, discrete logarithm problem etc. Since 1985, soon after the introduction of elliptic curves in cryptography by Neal Koblitz and Victor Miller, the discrete logarithm problem (DLP) offered by the elliptic curves is considered to be one of the most popular and widely accepted computationally intractable hard problem which has fully exponential complexity in cryptanalysis provided the elliptic curve parameters are drawn carefully.

The chapter presents scope, objectives, preliminaries including building blocks of the thesis and the main motivatations that inspired the work of the thesis.

1.1 Publications from this chapter

The Introduction in this Chapter has contributed the "Introduction" components of the following journal papers published in the thesis.

1. **Kunal Abhishek** and E. George Dharma Prakash Raj, *Evaluation of Computational Approaches of Short Weierstrass Elliptic Curves for Cryptography*, Cybernetics and Information Technologies (2021). (DOI: 10.2478/cait-2021-0045)

2. **Kunal Abhishek** and E. George Dharma Prakash Raj, *Computation of Trusted Short Weierstrass Elliptic Curves For Cryptography*, Cybernetics and Information Technologies (2021). (DOI: 10.2478/cait-2021-0020)

3. **Kunal Abhishek** and E. George Dharma Prakash Raj, *Computational Investment in Generation of Elliptic Curves Randomly over Large Prime Fields*, Concurrency and Computation Practice and Experience (2022). (Status: Under Revision)

4. **Kunal Abhishek** and E. George Dharma Prakash Raj, *On Random Number Generation for Kernel Applications*, Fundamenta Informaticae, IOS Press (2022). (Status: Accepted - In press)

1.2 Scope of the Thesis

The scope of the thesis broadly includes evaluation of computational techniques and computational resource estimation of **Short Weierstrass form of elliptic curves** and derivation of trusted elliptic curves over large prime fields. The trusted elliptic curves are aimed for implementation in construction of a novel **C**ryptographically **S**ecure **P**seudo **R**andom **N**umber **G**enerator for the operating system **K**ernels such as Linux, Windows, Android, Mac/iOS/BSD.

The scope of the research outcomes of the thesis is pictorically shown in Figure 1.1.

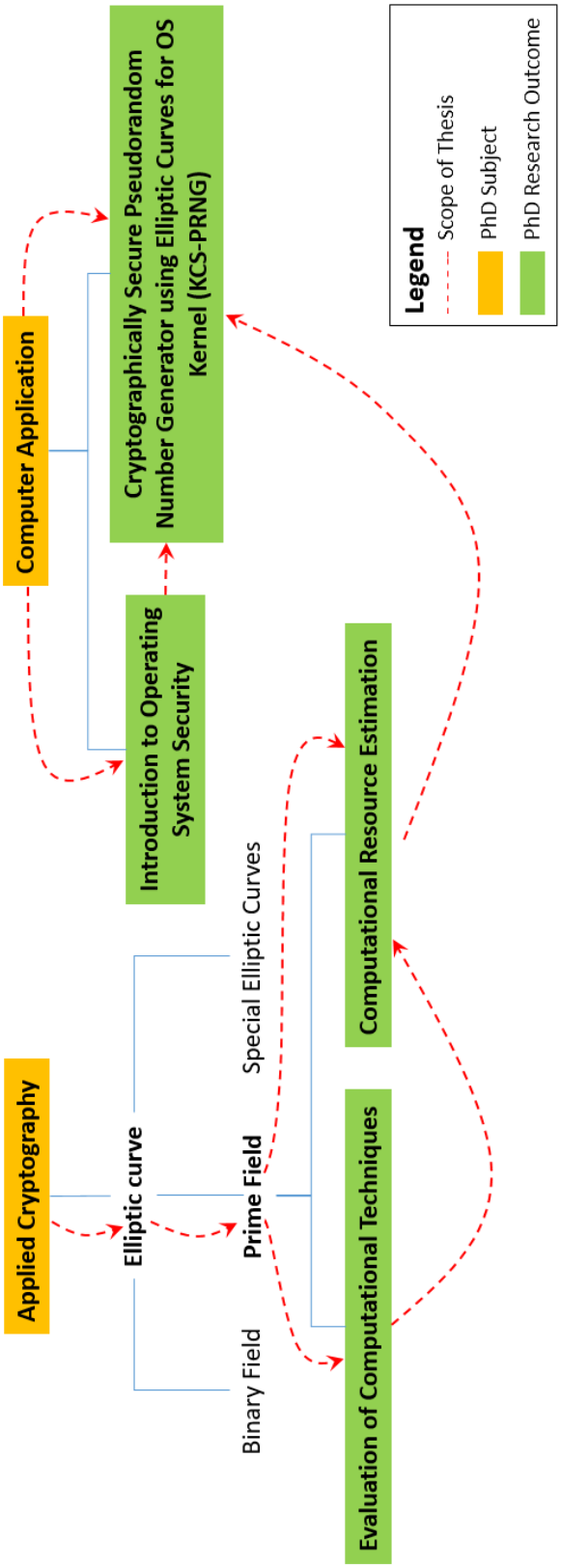


Figure 1.1: Scope of the thesis

1.3 Objectives of the Thesis

The objectives of the thesis are two-folded as given below:

1. To estimate the computational investment in terms of computing processor i.e., number of CPU clock cycles and number of searches or attempts made in the security parameter space of the elliptic curves, required for computation of elliptic curves randomly over a large prime field. Additionally, recommendation of two new elliptic curves over 256 bit and 384 bit prime fields which are cryptographically secure and trusted for use in security applications.

2. The recommended elliptic curves will be used in construction of a novel Cryptographically Secure Pseudo Random Number Generator (CSPRNG) with non-reproducibility property of its generated bitstreams which will be a viable candidate CSPRNG for the kernel applications.

The first objective is covered in the first part of the thesis whereas the second objective is covered in the second part of the thesis.

The pictorial view of thesis objectives is shown in Figure 1.2.

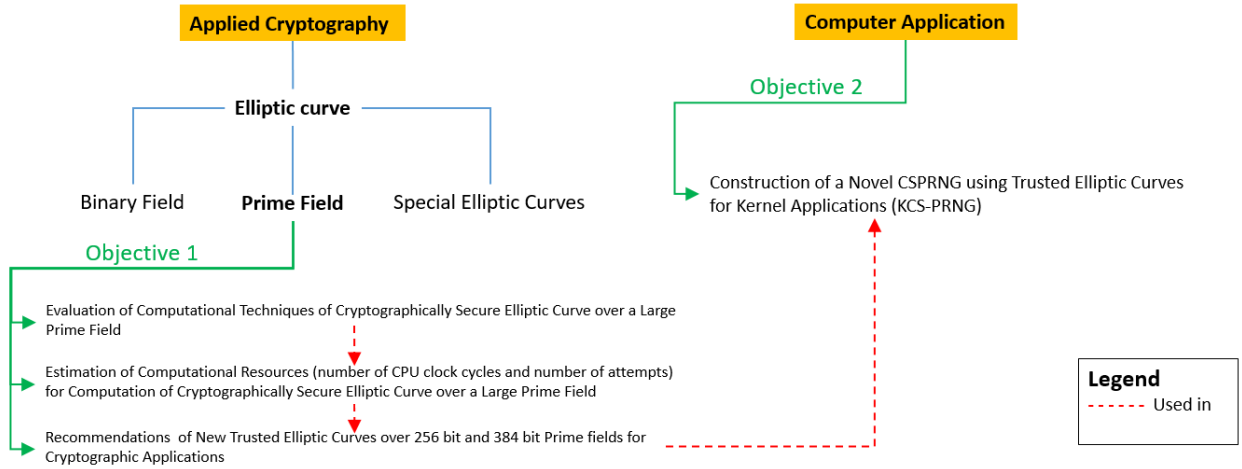


Figure 1.2: Objectives of the thesis

1.4 Preliminaries

1.4.1 Short Weierstrass Equation of Elliptic Curve

A Short Weierstrass elliptic curve $\mathbb{E}$ over a finite field $\mathbb{F}_p$ (where $q = p^m$, where p, a prime, is the characteristic of $\mathbb{F}_q$) denoted by $\mathbb{E}(\mathbb{F}_p)$ is the set of all solutions (x, y) to an equation

$$\mathbb{E} : y^2 = x^3 + ax + b \tag{1.1}$$

where the coefficients $a, b \in \mathbb{F}_p$ and $4a^3 + 27b^2 \neq 0$, together with a special point ∞ called the point at infinity which serves as the identity element of $\mathbb{E}$ [**1**]. The points (x, y) on $\mathbb{E}(\mathbb{F}_p)$ form an abelian group.

1.4.2 Elliptic Curve Discrete Logarithm Problem (ECDLP)

Definition 1 *(ECDLP): Given an elliptic curve $\mathbb{E}$ defined over a finite field $\mathbb{F}_q$, a point $P \in \mathbb{E}(\mathbb{F}_q)$ of order n, and a point $Q \in \langle P \rangle$, find the integer $l \in [0, n-1]$ such that*

$$Q = lP \tag{1.2}$$

The integer l is called the discrete logarithm of Q to the base P, denoted $l = log_P Q$ [**2**].

The security of any elliptic curve cryptosystem lies in selection of those elliptic curves whose discrete logarithm problem (ECDLP) is thought to be mathematically infeasible to solve. Moreover, the order of an elliptic curve is expected to be a prime to exhibit maximum ECDLP Security [**3**]. However, there are some special curves whose orders have special properties on which fast algorithms like Menezes, Okaoto and Vanstone (MOV) [**4**] and Frey-Ruck can be applied to solve ECDLP with sub-exponential complexity [**2**, **5**]. Examples of such special curves are supersingular elliptic curves [**4**] (Definition 2) whose ECDLP can be reduced

to some extension field $\mathbb{F}_{q^k}^*$ where k is some integer called embedding degree [**6**] (Definition 3) of the elliptic curve and the prime field anomalous curves [**7**] (Definition 4) respectively. One can use Theorem 1 [**4**] to determine if an elliptic curve of a certain order exists whereas Definition 2 and Definition 4 defines supersingular curve and prime field anomalous curve respectively. One needs to carefully consider elliptic curve with non-supersingularity, sufficient embedding degree, non-anomalous and suitable class number for intractable ECDLP required for cryptography.

Theorem 1 *There exists an elliptic curve of order $N = q + 1 - t$ over $\mathbb{F}_q$ where $q = p^m$, where p, a prime, is the characteristic of $\mathbb{F}_q$ and t is the trace of elliptic curve $\mathbb{E}(\mathbb{F}_q)$, if and only if one of the following condition holds:*

1. *$t \not\equiv 0 \pmod{p}$ and $t^2 \leq 4q$*
2. *m is odd and one of the following holds:*
   - *i. $t = 0$.*
   - *ii. $t^2 = 2q$ and $p = 2$.*
   - *iii. $t^2 = 3q$ and $p = 3$.*
3. *m is even and one of the following holds:*
   - *i. $t^2 = 4q$.*
   - *ii. $t^2 = q$ and $p \not\equiv 1 \pmod{3}$.*
   - *iii. $t = 0$ and $p \not\equiv 1 \pmod{4}$.*

Definition 2 *(Supersingular Elliptic Curves): If $\#\mathbb{E}(\mathbb{F}_q) = q + 1 - t$ be the order of elliptic curve $\mathbb{E}(\mathbb{F}_q)$ then $\mathbb{E}$ is said to be supersingular if $p \mid t$ where t be the trace of $\mathbb{E}$.*

It is deduced from Theorem 1 that $\mathbb{E}(\mathbb{F}_q)$ is supersingular iff $t^2 = 0, q, 2q, 3q$ or $4q$ [**1**]. However, a randomly computed elliptic curve has the probability $O(\frac{1}{\sqrt{p}})$ of being supersingular [**6**]. Supersingular elliptic curves are vulnerable to attack due to Menezes, Okamoto and Vanstone (MOV) which solves discrete logarithm problem (DLP) of supersingular curves to the DLP in a finite field with sub-exponential complexity [**7**].

Definition 3 *(Embedding Degree of Elliptic Curve): If* $\mathbb{E}(\mathbb{F}_p)$ *be the elliptic curve over* $\mathbb{F}_p$ *then* $\mathbb{E}$ *is said to have embedding degree* k*, a smallest positive integer, such that* $n \mid (q^k - 1)$ *where* n *be the base point order.*

It is observed that if the embedding degree k of $\mathbb{E}(\mathbb{F}_q)$ is low, say, $k < 6$ then $\mathbb{E}$ becomes a supersingular elliptic curve, if $k = 6$ then supersingular curve will be in characteristic 3 only. It is also observed that ECC standards do not allow elliptic curves with low embedding degrees. The ordinary elliptic curves certainly require $k > 6$. Generally, $k \geq 20$ is sufficient to guarantee intractibility of the discrete logarithm problem in $\mathbb{F}^*_{q^k}$. However, Boneh et. al. insisted to use $k \geq \frac{(q-1)}{100}$ for intractibility of discrete logarithm problem [**8**].

Definition 4 *(Prime Field Anomalous Curves): If* $\mathbb{E}(\mathbb{F}_p)$ *be the elliptic curve over* $\mathbb{F}_p$ *then* $\mathbb{E}$ *is said to be prime field anomalous if* $\#\mathbb{E}(\mathbb{F}_p) = p$ *where* $\#\mathbb{E}(\mathbb{F}_p)$ *be the order of* $\mathbb{E}$*.*

Prime field anomalous curves are trace one curves for which the ECDLP can be solved in linear time [**7**, **9**]. The prime field anomalous attack does not extend to any other classes of elliptic curves but the one having trace one [**7**].

Definition 5 *(Class Number): Let* $h(N)$ *denotes the class number of the order* N *of elliptic curve* $\mathbb{E}$*. Then* $h(N)$ *is the minimum degree of a number field over which the elliptic curve* $\mathbb{E}$ *admits a faithful lift.*

Large class number in Complex Multiplication (CM) theory is used to prevent possible lifting of elliptic curve to the number field using complex multiplication where ECDLP can be solved comfortably [**10**]. The CM-method is discussed in detail in Section 2.3.4.

1.4.3 Elliptic Curve Group Law

Discrete points on the elliptic curves follow Group Laws [**2**, **11**] which states

i. Identity: $P + \mathcal{O} = \mathcal{O} + P = P \; \forall P \in \mathbb{E}$

ii. Negatives: If $P \in \mathbb{E}$, then $P + (-P) = \mathcal{O}$ and also, $-\mathcal{O} = \mathcal{O}$

iii. Point Addition and Point Doubling: Suppose (x_1, y_1), (x_2, y_2) and (x_3, y_3) denote the coordinates of P, Q and $P + Q$ respectively, then x_3 and y_3 are given by,

$$\begin{aligned} x_3 &= \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2 \\ y_3 &= -y_1 + \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x_1 - x_3) \end{aligned} \tag{1.3}$$

If $P = Q$ then $P + Q = P + P = 2P$ and therefore x_3 and y_3 are given by,

$$\begin{aligned} x_3 &= \left(\frac{3{x_1}^2 + a}{2y_1}\right)^2 - 2x_1 \\ y_3 &= -y_1 + \left(\frac{3{x_1}^2 + a}{2y_1}\right)(x_1 - x_3) \end{aligned} \tag{1.4}$$

1.4.4 Point Counting on Elliptic Curve

The order of an elliptic curve is defined by the number of points which forms the elliptic curve group. Following are the popular point counting theorems/algorithms on elliptic curves:

Hasse's Theorem

Hasse's Theorem [**2**] is a fundamental theorem that provides a rough estimate of the bounded order of elliptic curve.

Theorem 2 *(Hasse) Let an elliptic curve* $\mathbb{E}$ *be defined over a finite field with* q *elements (*$\mathbb{F}_q$*) (where* $q = p^m$*, where* p*, a prime, is the characteristic of* $\mathbb{F}_q$*), then order of* $\mathbb{E}(\mathbb{F}_q)$ *satisfies*

$$|N - q - 1| \leq 2\sqrt{q} \tag{1.5}$$

where N *be the number of* $\mathbb{F}_q$*-points on* $\mathbb{E}$.

The proof of Hasse's theorem can be seen in [**11**].

Shank's Baby-Step-Giant-Step (BSGS) Algorithm

It is a deterministic algorithm to find the order of a point on an elliptic curve which requires approximately $\sqrt{N}$ steps and around $\sqrt{N}$ storage where N is the order of the elliptic curve [**11**]. It is a fully exponential time algorithm that works on any group [**12**]. The main drawback of this algorithm is that it has to store $O(\sqrt{N})$ group elements and cannot be parallelized in an efficient way [**13**]. Details of BSGS can be seen in [**14**, **15**].

Naive Approach

The naive way [**16**] of counting rational points on elliptic curves defined over small finite fields of odd characteristic p is to evaluate the sum $p + 1 + \sum_{x=0}^{p-1} \frac{x^3+ax+b}{p}$.

The naive approach works well with small p and Cohen [**17**] suggested that this approach is appropriate for $p < 10000$.

Mestre's Algorithm

Mestre's algorithm is a simplification of certain group theoretical computations in the Baby-Step-Giant-Step algorithm [**15**]. This algorithm can be viewed as combination of the following Theorem 3 [**15**] and Theorem 4 [**15**].

Theorem 3 *Let $p > 457$ be a prime and $\mathbb{E}$ be an elliptic curve over $\mathbb{F}_p$ then either $\mathbb{E}$ or its quadratic twist $\mathbb{E}'$ admits an $\mathbb{F}_p$-rational point of order at least $4\sqrt{q}$.*

The proof can be found in [**15**].

Theorem 4 *Let $p > 229$ be a prime and let $\mathbb{E}$ be an elliptic curve over $\mathbb{F}_p$ then either $\mathbb{E}$ or its quadratic twist $\mathbb{E}'$ admits an $\mathbb{F}_p$-rational point P with the property that the only integer $m \in (p+1-2\sqrt{p}, p+1+2\sqrt{p})$ for which $mP = 0$ is the order of the group of points.*

The proof can be seen in [**15**]. Theorem 3 and Theorem 4 overcomes the failure of the Baby-Step-Giant-Step strategy when the value of m is more than one such that $mP = 0$ for a number of points on the elliptic curve $\mathbb{E}$. By replacing $\mathbb{E}$ by its quadratic twist $\mathbb{E}'$, one can avoid multiple values for m for which $mP = 0$. More details can be found in [**15**]. Mestre's algorithm works in field characteristic 2 and is based on a $2-adic$ version of the Arithmetic-Geometric-Mean (AGM) [**13**].

Schoof's Algorithm

Schoof [**18**] proposed the first polynomial time algorithm to compute cardinality $\#\mathbb{E}(\mathbb{F}_q)$ of an elliptic curve using $l-adic$ approach. Schoof's algorithm proceeds with computing trace of the Frobenius Endomorphism t modulo suficiently many primes l such that $\prod l \geq B$ where $B > 4\sqrt{q}$. The algorithm uses Chinese Remainder Theorem (CRT) to compute cardinality of the elliptic curve [**2**]. The observed time complexity of the algorithm is $O(log^{3\mu+2}q)$ with space complexity

as $O(log^3 q)$ where μ is a constant such that multiplication of two m-bit integers can be computed in $O(m^\mu)$ time. It works well when elliptic curve is defined over small prime sizes [**2**]. Schoof's algorithm for computing order of the elliptic curve $\mathbb{E}$ over $\mathbb{F}_q$ where $q = 2^{135}$ has running time estimated by Koblitz [**12**] as approximately equal to 3×10^{19} which is certainly not a practical choice to use elliptic curve in public key cryptosystem. Later, Elkies and Atkin [**19**] improved this running time complexity of Schoof's Algorithm with a new algorithm named as SEA algorithm.

Satoh's Algorithm

Satoh [**18**] proposed p-adic methods to find group order induced by an elliptic curve. The algorithm proceeds with lifting the elliptic curve and the Frobenius endomorphism to a p-adic ring. In the next step the trace of the Frobenius Endomorphism $t \ modulo \ p^m$ with $p^m > 4\sqrt{q}$ is recovered from the lifted data. The time complexity of this algorithm is $O(n^{2\mu+1})$ for a fixed p whereas the space complexity is found to be $O(n^3)$. Satoh's algorithm is useful in case of small value of p only as the time complexity grows as $O(p^2 log^\mu p)$ [**18**] as well as in case of small field characteristic greater than 5 [**20**].

SEA (Schoof-Elkies-Atkin) Algorithm

Elkies and Atkin improved Schoof's algorithm to find elliptic curve group order by reducing the time complexity to $O(log^{2\mu+2} q)$ and space complexity to $O(log^2 q)$ [**18**]. They used isogenies to improve the efficiency of Schoof's algorithm [**11**]. Using l-adic algorithm, SEA algorithm takes $O((logq)^{4+\epsilon})$ bit operations where ϵ is a positive constant, to compute order of the elliptic curve with fast arithmetic and consumes $O((logq)^2)$ memory [**21**]. SEA algorithm uses BSGS which demands good resource in terms of space. Details on SEA algorithm can be seen in [**16**].

1.4.5 Random Number Generation

A random number generator (RNG) is classified in two basic classes [**22**]: first, a deterministic random number generator (DRNG) or a pseudorandom number generator (PRNG) which needs a seed value as input and produces random looking bitstreams using some deterministic algorithm. Second, a true random number generator (TRNG) which uses physical and non-physical sources to generate true randomness. It is imperative to note that unlike PRNG or DRNG, TRNG does not need any seed value but uses non-deterministic effects or physical experiments to generate the true random bits [**22**]. The significant differences between PRNG and TRNG are that the PRNG generates random sequences at very fast rate which has large period and properties of independence and equally likeliness whereas TRNG is slow, having infinite period, costly in deployment and has the possibility of manipulation. Unlike TRNG, PRNG has less development and deployment cost (no need of dedicated hardware) but can produce reasonably good random looking bitstreams. Figure 1.3 is shown to differentiate between TRNG and PRNG [**23**]:

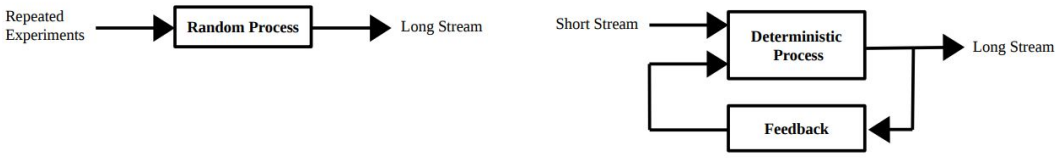


Figure 1.3: TRNG and PRNG

PRNGs have vital role in generating keys, initialization vectors (IVs), nonce, session keys etc. for cryptographic applications. A PRNG is provably secure, if its security can be reduced to a well-established conjectured hard problem. These PRNGs are supposed to produce output bits which are reasonably random i.e. each bit has probability of 0.5 of occurance making them completely unpredictable.

1.4.6 RNG Requirements

Koc [**22**] and Schneier [**24**] collated the properties that various classes of RNG exhibit and formulated the following requirements:

1. $R1$: A random sequence generated by a RNG should have good statistical properties.

   This requirement enables a RNG with a large period.

2. $R2$: A random sequence generated by a RNG should be unpredictable.

   This requirement makes the prediction of the next bit infeasible in the stream, given the complete knowledge of the algorithm or hardware which generates the sequence and all of the previous bits in the stream. This gives the notion of Backward Secrecy.

3. $R3$: A random sequence generated by a RNG should not allow to compute previous internal state or values of the generator even if the internal state is known. This gives the notion of Forward Secrecy.

4. $R4$: A random sequence generated by a RNG should not be reliably reproduced.

   If the RNG is run twice with exactly the same input, it should produce two completely unrelated random sequences.

From definition [**24**], a PRNG meets only $R1$ requirement whereas CSPRNG meets $R1$, $R2$ and $R3$ requirements of RNG. However, a TRNG meets $R2$, $R3$ and $R4$ requirements of the RNG. In this thesis, the proposed KCS-PRNG is designed in such a way that it meets the $R1$, $R2$ and $R3$ requirements along with the $R4$ requirement of RNG to a practical extent.

1.4.7 Randomness for Kernel Applications

One of the most important kernel applications that requires high quality randomness is Address Space Layout Randomization (ASLR) [**25**] which is an efficient mitigation technique against remote code execution attacks by randomizing the memory address of processes to disable memory exploitation. The ASLR currently uses CSPRNG to randomize the logical elements contained in the memory objects at the time of pre-linking (at the time of installation of the application), per-boot (on every time the system boots), per-exec (when new executable image is loaded in memory called pre-process randomization), per-fork (every time a new process is created) and per-object (every time a new object is created). Figure 1.4 [**25**] shows the Per-boot versus Per-exec randomization to point out when randomization takes place in both the per-boot and per-exec processes.

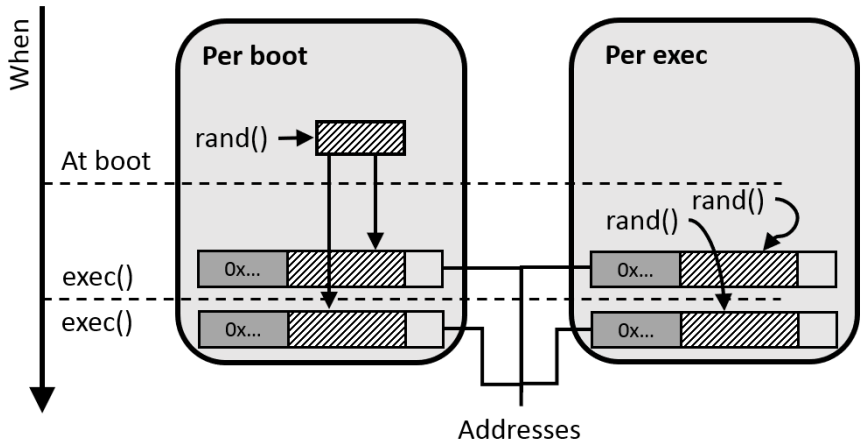


Figure 1.4: Per-boot versus Per-exec randomization

Similarly, Figure 1.5 shows that $mmap()$ system call allocates all the objects side by side in the $mmap\_area$ area during the per-object randomization. The $rand()$ provides random bits of desired length to the objects as shown in Figure 1.5 [**25**].

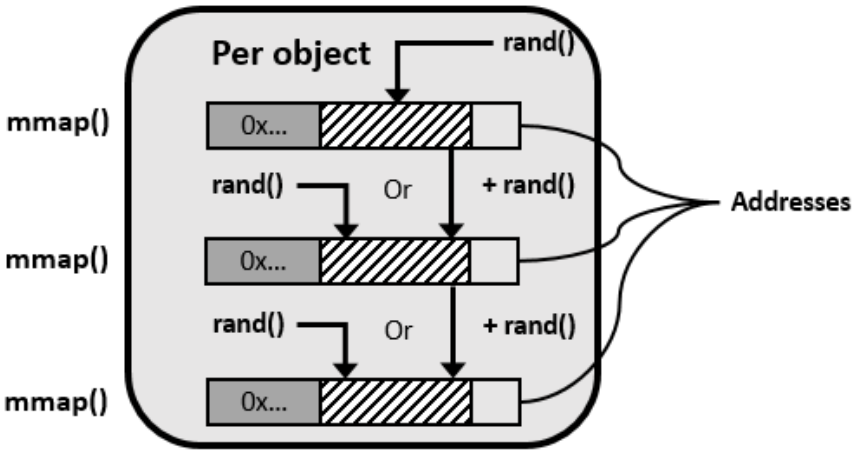


Figure 1.5: Per-object randomization

Another important kernel application is the Morris-Thompson scheme [**3**, **8**] which associates a n-bit random number with each password and concatenates and then encrypts together before storing it in the password file. A CSPRNG is used whenever a password is changed and a random number is required.

1.5 Motivation of the Thesis

1.5.1 Motivation for New Trusted Elliptic Curves

Non-standard elliptic curves are desirable for building critical strategic applications such as kernel applications. There are many incidents reported which are discussed in this thesis which proved that the standard elliptic curves are claimed to be cryptographically secure but they seriously lack trust for use in cryptography. Hence, new trusted elliptic curves are required whose parameters i.e., curve coefficients and prime, are free from any intentionally vulnerable non-disclosed properties.

1.5.2 Motivation for Computational Resource Estimation of Elliptic Curves

Computation of elliptic curves over large prime fields is a resource intensive operation and their computation within stipulated time is a big challenge which is often required in cryptographic interests. Thus, the motivation comes from the possibility to see the feasibility of sufficient financial allocation to arrange the computational resources (in terms of the CPU processor) required for generation of large order elliptic curves within stipulated time.

1.5.3 Motivation for Designing new CSPRNG for Operating System Kernels

The degree of security provided by ASLR technique depends on the predictability of the random memory layout of a program in an operating system kernel. Therefore, 'non-reproducibility' of the random sequences used in ASLR is essential. The motivation of the thesis is to use trusted large order elliptic curves in the design of a competent kernel CSPRNG such that it can generate non-reproducible pseudo random bitstreams for kernel applications.

1.6 Organization of the Thesis

The thesis covers seven research problems in applied cryptography and computer application domains. The thesis is overall organized in seven chapters as shown in Table 1.1.

Table 1.1: Organization of the Thesis

<table>
<tr><td colspan="2">Chapter 1 - Introduction</td></tr>
<tr><td colspan="2">Chapter 2 - Survey of Related Literature</td></tr>
<tr><td colspan="2">Chapter 3 - Problem Statements</td></tr>
<tr><td>Part I Evaluation and Computation of Novel Short Weierstrass Elliptic Curves
Chapter 4 - The Proposed Cryptographically Secure and Trusted Elliptic Curves Over 256 bit and 384 bit Prime Fields
Chapter 5 - The Proposed Computational Resource Estimation of Short Weierstrass Elliptic Curves</td><td>Part II Construction of Novel CSPRNG using elliptic curves for kernel applications
Chapter 6 - Design and Implementation of The Proposed KCS-PRNG</td></tr>
<tr><td colspan="2">Chapter 7 - Conclusion and Future Research</td></tr>
</table>

Chapter 2

Survey of Related Literature

"It is possible to write endlessly on elliptic curves."

- Serge Lang

The thesis evaluates Short Weiertrass elliptic curves from computational, security and trust perspectives and demonstrates the trend in the computation of elliptic curves in its standardization. The thesis argues that though standard elliptic curves provide compatibility and interoperability across diverse applications, they are not preferable in strategic applications due to the trust issues related with the procedure of computing curve parameters. Hence, non-standard or custom elliptic curves which are randomly generated in a closed environment are desired for development of mission critical applications such as operating system kernel applications, in particular. This chapter comprehensively surveys the deterministic and random approaches of computation of the Short Weierstrass elliptic curves and evaluates them for their implementation in kernel or strategic applications. This chapter solves the first problem[1] of the thesis which is mentioned in Chapter 3.

[1]Evaluation of computational approaches and selection criteria of elliptic curves over prime fields from computation, security and trust perspectives.

2.1 Publications from this chapter

The survey and literature review of this chapter contributes the "Literature Survey and Review" components of the following journal papers:

1. **Kunal Abhishek** and E. George Dharma Prakash Raj, *Evaluation of Computational Approaches of Short Weierstrass Elliptic Curves for Cryptography*, Cybernetics and Information Technologies (2021). (DOI: 10.2478/cait-2021-0045)

2. **Kunal Abhishek** and E. George Dharma Prakash Raj, *Computation of Trusted Short Weierstrass Elliptic Curves For Cryptography*, Cybernetics and Information Technologies (2021). (DOI: 10.2478/cait-2021-0020)

3. **Kunal Abhishek** and E. George Dharma Prakash Raj, *Computational Investment in Generation of Elliptic Curves Randomly over Large Prime Fields*, Concurrency and Computation Practice and Experience (2022). (Status: Under Revision)

4. **Kunal Abhishek** and E. George Dharma Prakash Raj, *On Random Number Generation for Kernel Applications*, Fundamenta Informaticae, IOS Press (2022). (Status: Accepted - In press)

2.2 Introduction

Computation of elliptic curve requires a lot of mathematical research to compute curve's parameters over large prime field for its use in cryptography [**26**]. There are several agencies like National Institute of Standards and Technology (NIST), Standards for Efficient Cryptography Group (SECG), Brainpool and others who have recommended standard elliptic curves over various prime field

orders. However, it is important to note the rationale behind the approaches adopted for selection of elliptic curve parameters from computational and security perspectives. In this thesis, a comprehensive review on the computational approaches and the selection criteria of elliptic curve parameters for use in cryptography is presented. The scope of this survey and subsequently, of this thesis, is limited to the Short Weierstrass form of elliptic curves which are used for constructing most of the present cryptosystems such as Public Key Infrastructure (PKI) [**27**], Secure Shell (SSH), Transport Layer Security (TLS), IPSec, JSON Web Encryption (JWE) [**28**] etc.

This chapter encompasses authentic observations, theories and results contributed by renowned researchers and scientists through their publications in various reputed journals, conferences, workshops, text books and their valuable comments or public statements on the subject during 1978 - 2021. The chapter also includes the experiences and observations made during the research execution of this thesis.

The key outcomes of the chapter are as follows:

- The chapter evaluates the approaches and selection criteria for computation of cryptographically secure Short Weierstrass elliptic curves and discusses the evolution of elliptic curve cryptography (ECC) with theoretical advancements in cryptographic mathematics and their significant impact on standardization of computational methods by various agencies. Subsequently, the chronology of attacks on ECDLP and their countermeasures is presented which is crucial in deciding the selection criteria of cryptographically secure elliptic curves. Additionally, the selection criteria and verification criteria of cryptographically secure Short Weierstrass elliptic curves are discussed and a new cryptographically secure

Short Weierstrass elliptic curve is computed using random approach for demonstration purposes in this chapter.

- The chapter demonstrates a trend in computational approaches of Short Weierstrass elliptic curves in standards recommended by various agencies. A comparative study of standard and non-standard elliptic curves from computational, trust and security perspectives is also presented.

- The chapter presents a review on the work of Koblitz [**12**] who probabilistically estimated the number of searches required to successfully generate suitable elliptic curve over the binary field which motivated this thesis to work further on deriving such estimates for elliptic curves over the prime fields. The status of ECC-based cryptosystems in presence of quantum computers is also reviewed in the light of recent work of Roetteler et. al. [**39**, **40**].

- The chapter presents a factual study of three most popular kernel CSPRNGs called /dev/(u)random, Yarrow and Fortuna respectively.

2.3 Computational Approaches of Elliptic Curves

2.3.1 Evolution of Elliptic Curves for Cryptography

Elliptic curves were extensively studied and reviewed for cryptography soon after the proposals of Neal Koblitz and Victor Miller during 1985-1987. Since then, numerous advancements in the theory of elliptic curve cryptography and its cryptanalysis took place which are described in Table 2.1 with their significant impacts on evolution of elliptic curve computational standards.

Table 2.1: Evolution of Short Weierstrass Elliptic Curves for Cryptography

Year	Event in Elliptic Curve Cryptography	Impact on ECC Standardization
1985	Elliptic curves were proposed for use in cryptography.	ECC were extensively studied to develop cryptosystems.
1987	Efficient point counting algorithm on elliptic curves by Schoof, Elkies and Atkin called SEA algorithm was developed [**18**, **19**].	Uses complexity $O(ln^5 p)$ for point counting.
1992	Elliptic Curve based Digital Signature Algorithm (ECDSA) was developed [**4**].	Considered as a mature signature scheme in NIST standard.
1993	Reduction of ECDLP of supersingular elliptic curves having trace zero to logarithm in a finite field [**4**].	Became selection criteria for safe elliptic curve in all standards.
1994	Proposal of Shor algorithm [**29**] generalizes to solve ECDLP random quantum polynomial (RQP) time using quantum computers.	Led to realization that elliptic curves will be unsafe once sufficient quantum capability is built.So, new computational standard is required for quantum resistance.
1996	It was proved that the condition $N \mid (q^k - 1)$ is sufficient to realize the MOV algorithm under mild condition. Further, it was proved that randomly generated curves have $k > log^2 q$ [**30**].	Became selection criteria for safe elliptic curve in all standards.
1997	Proposal of a linear algorithm to solve ECDLP of trace one [**9**, **31**].	Became selection criteria for safe elliptic curve in all standards.
1999	NIST recommendation of 15 elliptic curves [**32**].	Widely accepted standard later.
2000	SECG recommendation of elliptic curves [**33**].	Widely accepted standard later.
2005	Recommendation of Brainpool first set of elliptic curves for standardization[**34**].	International effort for elliptic curve standardization.

Continued to next page..

Table 2.1 – continued from previous page

Year	Event in Elliptic Curve Cryptography	Impact on ECC Standardization
2010	Brainpool revised their specifications and published. Request for Comment (RFC) 5639 [**35**].	Standard established.
2014	Review of existing elliptic curves generation mechanisms by Tanja and Bernstein [**36**] who coined two terms: ECDLP security and ECC security. They observed that Short Weierstrass form of elliptic curves are dominant in both the software and hardware implementations.	Two new terms ECDLP security and ECC security became important verification criteria for curve selection with side channel attack resistance.
2014	NUMS-curve (Nothing Upon My Sleeves) were proposed under IETF standard [**37**].	Curves with better performance proposed under IETF Standard.
2015	NIST Call for next generation elliptic curves with new models and optimized parameters resistant to side channel analysis was placed [**37**].	NIST wanted to replace its standard elliptic curves.
2016	NIST report [**38**] on Post Quantum Cryptography (PQC). Resistance of elliptic curve cryptosystems was looked for quantum computing.	Isogenies of supersingular elliptic curves were discussed as resistant to PQC instead of ECDLP.
2017-2021	Proposal of Quantum resources required to run Shor algorithm to solve ECDLP in polynomial time [**39**, **40**].	Roeteller et. al. suggested quantum resource estimates to break ECDLP.

Note: N=Curve order, q=prime power, k= embedding degree

2.3.2 Chronology of Attacks on ECDLP and their Countermeasures

Elliptic curves are expected to have proper implementation of the countermeasures to resist important attacks on its ECDLP. Table 2.2 [**9**] briefly depicts such

countermeasures for important discrete logarithm (DLP) based attacks and pairing based attacks which resulted in the evolution of cryptographically safe elliptic curve selection criteria.

Table 2.2: Chronology of Attacks on ECDLP and their Countermeasures

Year	Attack/Type	Description	Countermeasure
1978	Pohlig-Hellman, DLP attack	Private key can be recovered using Chinese Remainder Theorem [**41**].	N must be a prime or near prime with small cofactor, $N \geq 2^{160}$ [**2**].
1978	Pollard-rho, DLP attack	A parallelized Pollard-rho on r processors can solve ECDLP in $\frac{(\sqrt{\pi n})}{\sqrt{2}r}$ steps [**2**, **42**].	$n \geq 2^{160}$ [**11**, **42**].
1978	Pollard's Lambda, DLP attack	Faster method than Pollard-rho when ECDLP lies in subinterval $[1, b]$ of $[1, n-1]$, where $b < 0.39n$ [**11**].	Private key should be selected uniformly at random within interval $[1, n-1]$ [**39**].
1979	Index-Calculus, DLP attack	ECDLP can be solved using multiplicative group $\mathbb{F}_q^*$ of the finite field $\mathbb{F}_q$ [**11**].	Small prime fields should be avoided i.e., $n \geq 2^{160}$ [**11**].
1985	Exhaustive Search, DLP attack	Computes successive multiples of base point till public key is achieved.	n should be sufficiently large [**7**].
1985	Shanks' Baby step Giant step, DLP attack	Fully exponential deterministic algorithm to determine n on $\mathbb{E}(\mathbb{F}_p)$ which requires approximately $\sqrt{N}$ steps and around $\sqrt{N}$ storage.	$n \geq 2^{160}$ [**11**].

Continued to next page..

Table 2.2 – continued from previous page

Year	Attack/Type	Description	Countermeasure
1993	Weil pairing and Tate pairing attacks, Pairing based attack	ECDLP of $\mathbb{E}(\mathbb{F})_q$ can be reduced to ordinary DLP on extension field $\mathbb{F}^*_{q^k}$ for some $k \geq 1$ where the number field sieve algorithm can be used to solve ECDLP [**1**, **4**]. MOV reduction attack [**4**].	$n \nmid (q^k - 1)$ [**6**, **18**] $\forall 1 \leq k \leq 20$ [**2**]. $p \nmid t$ and $t^2 \neq 0, q, 2q, 3q$ or $4q$ [**4**]. (Non-supersingularity)
1997	Multiple logarithm, DLP attack	Multiple instances of ECDLP for the same elliptic curve parameters.	$n > 2^{160}$
1998	Prime field anomalous curve, Pairing based attack	Trace of $\mathbb{E}(\mathbb{F})_p = 1$ i.e., $\#\mathbb{E}(\mathbb{F}_p) = p$ [**7**, **43**].	$N \neq q$ [**4**].

Note: q=size of underlying field, p=prime characteristic, n=order of a point on $\mathbb{E}$, N=order of $\mathbb{E}$, r=number of processors, k=embedding degree, t=trace of curve.

Once a suitable cryptographically secure elliptic curve is selected, the public key cryptosystem can be developed using that elliptic curve to get performance gain and competitive security with much smaller key size than the legacy RSA or ElGamal based cryptosystems.

2.3.3 Approaches for Computation of Short Weierstrass Elliptic Curves

Computation of Short Weierstrass elliptic curve over prime field involves rigorous mathematical validation of its parameters to certify its suitability for cryptography. These validations are meant to certify that the elliptic curve has the claimed order, resists all known attacks on ECDLP and base point order has also the claimed order

[**2**]. There are usually two approaches either of which can be used to compute an elliptic curve over prime field: first, the deterministic approach and second, the random approach. Though in both the deterministic and random approaches, following conditions are critical for the elliptic curve to meet cryptographic requirements [**1**, **2**, **44**] :

1. $C1$: Resistance to Pohlig-Hellman and Pollard's Rho attack i.e., $n > 2^L$ where n is sufficiently large prime that divides order of the elliptic curve group $\#\mathbb{E}(\mathbb{F}_q)$. Here, $L \geq 160$, the length in bits.

2. $C2$: Resistance to Semaev–Smart–Satoh–Araki attack (Smart-ASS) [**9**, **43**] i.e. $L \leq \lfloor log_2 q \rfloor$ ensures $2^L \leq q$ or $\#\mathbb{E}(\mathbb{F}_q) \neq q$. It avoids the attack on prime-field-anomalous curves.

3. $C3$: $n > 4\sqrt{q}$ guarantees that $\mathbb{E}(\mathbb{F}_q)$ has a unique subgroup of order n as $\#\mathbb{E}(\mathbb{F}_q) \leq (\sqrt{q}+1)^2$ by Hasse's theorem and so, $n^2 \nmid \#\mathbb{E}(\mathbb{F}_q)$.

2.3.4 Evaluation of Deterministic Approach

Generating elliptic curves in the cryptographic context is an intricate task. It involves consideration for standardization of elliptic curves to be used in cryptographic applications for compatibility and interoperability purposes. The standardization of elliptic curves further involves fixing of various criteria related to selection of the curve parameters. In this section, the deterministic approach of computation of Short Weierstrass elliptic curve is evaluated on the basis of their computational method, computational complexity, security, trust and specific gains of elliptic curves computed by the deterministic method.

- Computational method

  Complex Multiplication (CM) is a popular deterministic approach to select cryptographically safe elliptic curves over prime fields and widely accepted

approach for standardization of elliptic curves. The CM method is called the Atkin-Morain method when the elliptic curve is derived over prime field [**45**].

The CM method proceeds with fixing the prime p first and then constructs an elliptic curve over the field $\mathbb{F}_p$ [**44**]. The CM method gives a choice for selecting primes of special forms. The CM method takes p as input and determines the CM discriminant D. Then p is selected such that it meets the conditions $C1$, $C2$ and $C3$. The CM method is efficient when p and the elliptic curve order $\#\mathbb{E}(\mathbb{F}_p) = p + 1 - t$ are chosen such that CM-field of $\mathbb{E}$ i.e., $\mathbb{Q}(\sqrt{t^2 - 4p})$ has small class number [**1**, **2**].

A crucial step of CM method is to compute the roots of a special type of class field polynomials called the Hilbert and Weber polynomials [**46**]. These polynomials are uniquely determined by the CM discriminant D.

Equations (2.1), (2.2) [**47**] and equation (2.3) [**48**] constitute the basis of computation of Short Weierstrass elliptic curves for use in cryptography.

Definition 6 *(Twist) Given $E : y^2 = x^3 + ax + b$ with $a, b \in \mathbb{F}_p$ the twist of $\mathbb{E}$ by c is the elliptic curve given by*

$$\mathbb{E}_c : y^2 = x^3 + ac^2x + bc^3 \tag{2.1}$$

where $c \in \mathbb{F}_p$.

Theorem 5 *If the order of an elliptic curve $\#\mathbb{E}(\mathbb{F}_p) = p + 1 - t$, then the order of its twist is given as*

$$\mathbb{E}_c(\mathbb{F}_p^*) = \begin{cases} (p+1-t) & if\ c\ is\ square\ in\ \mathbb{F}_p \\ (p+1+t) & if\ c\ is\ non-square\ in\ \mathbb{F}_p \end{cases} \tag{2.2}$$

Algorithm 1 Elliptic curve generation over prime field using CM approach

Require: Nil

Ensure: Elliptic curve over a prime field $\mathbb{E}(\mathbb{F}_p)$

1. Choose p, a prime
2. Find smallest CM discriminant D from equation (2.3) along with trace t
3. Construct the orders of the two elliptic curves $\#\mathbb{E}(\mathbb{F}_q) = p + 1 \pm t$
4. **if** one of the order of the curve is a prime or nearly a prime ▷ Fix elliptic curve order
5. **else** Repeat step 1 to determine D and t
6. **end if**
7. Construct the class polynomial $H_D(x)$ ▷ Class polynomial is independent of p
8. Find a root j_0 of $H_D(x)(mod\ p)$ ▷ j_0 is the j-invariant of the desired elliptic curve
9. Set $k = j_0/(1728 - j_0)(mod\ p)$ ▷ so that the elliptic curve $\mathbb{E} : y^2 = x^3 + 3kx + 2k$
10. **if** $\#\mathbb{E} \neq (p + 1 - t)$
11. Construct the twist $\mathbb{E}_c$ ▷ using a randomly selected non-square $c \in \mathbb{F}_p$ following equations (2.1) and (2.2)
12. **return** $\mathbb{E}_c$
13. **else**
14. **return** $\mathbb{E}$
15. **end if**

Theorem 6 *(Atkin-Morain) Let p be an odd prime such that*

$$4p = t^2 + Ds^2 \tag{2.3}$$

for some $t, s \in \mathbb{Z}$. Then there is an elliptic curve $\mathbb{E}$ defined over $\mathbb{F}_p$ such that $\#\mathbb{E}(\mathbb{F}_p) = p + 1 - t$.

Equation (2.3) observes that D be the integer which can be determined from a given prime p called the CM discriminant of p. Algorithm 1 describes a general CM method [**17**] for constructing an elliptic curve over a given prime field.

CM method adheres to "Performance over slightly sacrificed security" principle for computation of Short Weierstrass elliptic curves. Fast elliptic curve computation is possible in CM method due to elimination of the need for a point counting algorithm and fixing of certain parameters like prime p with special structures [**49**]. CM method allows much faster arithmetic with elliptic curves as compared to random approach to achieve higher performance of elliptic curve cryptosystems [**2**]. It provides smaller, faster and easily implementable software code due to offline precalculations while adopting deterministic computational approach [**48**]. Prime order elliptic curves generated using CM method with $a = -3$ are backward compatible with implementation supporting most of the standardized elliptic curves [**50**].

- Computational complexity

  The bit complexity $(\mathcal{B})$ of CM method depends on b and h where b=length of field order p, h=class number, h_c=cross over class number for which the random approach and CM approach have the same runtime. When $h(D) < h_c(b)$ where D is the CM discriminant, then CM method is faster than random approach [**44**]. CM method can generate a prime order elliptic curve in time $\tilde{O}((logN)^4)$ [**48**].

- Security

  Deterministic approach is vulnerable due to non-disclosed attacks. The standards developed by various agencies have deterministic way of computing elliptic curves which are supposed to be (dis)trusted for ultra security sensitive applications. Standards are sometimes purposely designed in such a way that it can be manipulated by the agency who recommended those standards [**51**]. Also, sufficient information about the computational mechanisms of curve parameters have not been made

publically available [**6**]. It is always a concern for researchers that the ECDLP of deterministically computed elliptic curves can be solvable by using very efficient sub-exponential or polytime algorithm using non-guessable very high computing power unknown to outside world.

- Trust

  The elliptic curve parameters which are selected deterministically are sometimes distrusted due to lack of sufficient proofs of their computational mechanisms [**49**]. Moreover, trust in the curve parameters is doubtful due to possibility of intentional non-disclosed properties of the curve parameters. There are some serious statements of distrust expressed by many reputed scientists and researchers on NIST recommended elliptic curves which was generated through deterministic approach. Some of such statements of distrust are given as below:

  – "I no longer trust the constants. I believe the National Security Agency (NSA) has manipulated them through their relationships with industry." - Bruce Schneier [**52**]

  – "NIST should generate a new set of elliptic curves for use with ECDSA in FIPS 186... The set of high-quality curves should be described precisely in the standard, and should incorporate the latest knowledge about elliptic curves." - Edward Felten [**50**, **53**]

  – "However, in practice the NSA has had the resources and expertise to dominate NIST, and NIST has rarely played a significant independent role." - Koblitz, Koblitz and Menezes [**6**]

  – "We don't know how $Q = [d]P$ was chosen, so we don't know if the algorithm designer [NIST] knows [the backdoor] d." - Shumow and Ferguson [**54**]

  - "Consider now the possibility that one in a million of all curves have an exploitable structure that "they" know about, but we don't.. Then "they" simply generate a million random seeds until they find one that generates one of "their" curves." - Scott [**55**]
  - "NIST should ensure that there are no secret or undocumented components or constants in its cryptographic standards whose origin and effectiveness cannot be explained." - Steve Lipner [**50**, **53**]
  - Many more..

- Specific gains of deterministic approach

  CM method can only be adopted to construct ordinary elliptic curves with low embedded degree $k > 6$ [**6**]. CM method is not efficient if there is no restriction on the class number of the elliptic curve [**7**]. This method is useful in deriving elliptic curves with small class numbers for which ECDLP is hard and gives the same security level as given by the elliptic curves which are generated randomly [**2**, **7**].

2.3.5 Evaluation of Random Approach

Random approach allows to obtain elliptic curves which are ordinary and avoids any special form or structure. This approach uses 'early-abort strategy' to obtain desired elliptic curve [**2**]. A general observation is that elliptic curves generated using random approach have not been given preference for standardization like those elliptic curves which are generated using deterministic approaches. We evaluate random approach for computation of elliptic curves in various contexts as given below:

- Computational method

  In random approach, the elliptic curve generation algorithm computes curve parameters keeping ECDLP security and procedural transparency in consideration. The elliptic curve computation algorithm considers a list of security criteria and prefers security of the crypto applications over their performance during elliptic curve generation in general. Algorithm 2 describes a general random approach as preferred in [**1**, **2**, **3**, **4**, **9**, **17**, **18**, **19**, **29**, **36**] to derive cryptographically safe elliptic curve over prime field. Here the prime p is fixed and the coefficients a and b are kept varying till a suitable elliptic curve $\mathbb{E}$ with prime order N is obtained. Some validations to meet the cryptographic requirements $C1$, $C2$ and $C3$ are also conducted. It is observed that all the elliptic curve parameters such as p, a, b and $G_{x,y}$ are randomly generated in order to avoid any special structure or known values whose choices are ambiguous.

  A sample Short Weierstrass elliptic curve is computed using random approach as shown in Algorithm 2. The curve generation process using random approach is comprehensively discussed in Algorithm 5 of this thesis.

  Random approach adheres to the principle of "security over performance" for computation of elliptic curve parameters. Computing order of the elliptic curve is a time-intensive task and hence, selecting elliptic curve using random approach is a slower process as compared to the deterministic approach where one starts with fixing the order of the elliptic curve. Point compression and decompression also requires more computation in randomly generated elliptic curves [**49**].

Algorithm 2 Elliptic curve generation over prime field using random approach

Require: Randomness
Ensure: Elliptic curve over a prime field $\mathbb{E}(\mathbb{F}_p)$, $G_{x,y}$, N

1: Select randomly a prime p of desired size
2: Fix $K = GF(p)$ ▷ Generate a prime field K
3: Choose randomly coefficient a
4: Choose randomly coefficient b
5: Generate elliptic curve $\mathbb{E}(K)$ ▷ Elliptic curve over $\mathbb{F}_p$
6: **if** $4a^3 + 27b^2 \neq 0$ ▷ Non-singularity check as stated in equation (1.1)
7: **else** go to step 3
8: **end if**
9: Compute cardinality or order N of $\mathbb{E}(K)$
10: **if** N is prime ▷ Prime cardinality only to resist Pohlig-Hellman attack
11: **else** go to step 3
12. **end if**
13: **if** $\mathbb{E}$ is non-supersingular ▷ Non-supersingularity check to resist MOV attack
14: **else** go to step 3
15: **end if**
16: **if** $N \neq p$ ▷ Non-anomalous check as per criteria $C2$
17: **else** go to step 3
18: **end if**
19: Select randomly a base point $G_{x,y}$ on $\mathbb{E}$
20: Compute base point order n ▷ Such that size of $n \geq 160$ bits as per criteria $C1$ and $n > 4\sqrt{p}$ as per criteria $C3$
21: **if** $n \neq N$ ▷ Check for cofactor as 1
22: **else** go to step 18
23: **end if**
24: Compute Twist $\mathbb{E}_c$ ▷ Twist security of elliptic curve
25: **if** $\mathbb{E}_c$ is non-singular
26: Compute Cardinality N' of $\mathbb{E}_c$
27: **else** go to step 3
28: **end if**
29: **if** N' is prime
30: **else** go to step 3
31: **end if**
32: **if** $\mathbb{E}_c$ is non-supersingular
33: **else** go to step 3
34: **end if**
35: **return** $\mathbb{E}(\mathbb{F}_p), G_{x,y}, N$ ▷ Return elliptic curve parameters

- Computational complexity

  For random approach, the bit complexity $(\mathcal{B})$ depends on the length of prime (r_0) only and falls in the range $O(log^{5+\epsilon}k_0r_0)$ to $O(log^7k_0r_0)$ where $\epsilon > 0$ and k_0 is the cofactor [**44**].

- Security

  Random approach does not allow any special structure of the curve parameters in order to eliminate doubts on intentional non-disclosure of backdoors [**2**]. Elliptic curves which are randomly computed have no hidden goals which can be proved in determination of the curve parameters. It ensures that the elliptic curve parameters are trusted and not suspected to belong to a (not publicly known to be) vulnerable class. This approach is favourable when long term security is desired with an ignorable sacrifice of efficiency [**6**]. Elliptic curves can be frequently changed for security reasons when computed randomly [**49**]. The only way to compromise elliptic curve security in such case is to solve ECDLP rather than just attacking particular classes of weak elliptic curves [**49**]. Hence, random approach is specifically preferred to obtain elliptic curves for strategic or military grade cryptosystems.

- Trust

  Random approach ensures that no intentional construction with hidden weakness in the elliptic curve parameters is present in order to prevent future exploitation to recover user's private key [**2**]. The trust in derivation of the elliptic curve parameters are maintained due to the use of absolutely new values drawn randomly each time. Moreover, there are no patent issues with randomly selected new curve parameters and therefore, it requires minimum financial investments in using cryptosystems based on such elliptic curves.

Random approach protects against attacks in special classes of elliptic curves which may be discovered in future [**2**]. But random values of elliptic curve parameters are always arguable by others for their origination and random number generation, if not explained adequately.

- Specific gains of random approach

  Elliptic curves are computed with nearly the same probability to ensure that curves are not special in any sense when they are computed randomly [**2**, **44**]. The chances of $\mathbb{E}(\mathbb{F}_p)$ being supersingular is $O(p^{\frac{-1}{2}})$ which is rare in random approach [**6**]. It is computationally difficult to derive elliptic curves over large prime fields using random approach [**49**].

2.4 Selection Criteria of Short Weierstrass Elliptic Curves

Elliptic curves requires certain mathematical validations before their acceptance for implementation in cryptosystems. Table 2.3 shows important selection criteria of Short Weierstrass elliptic curve parameters and their benefits to select elliptic curves with desired properties.

Moreover, Table 2.3 lays the foundation of all the recommended Short Weierstarss elliptic curves suggested by various agencies in their standards. These mathematical validations suggest that the elliptic curve is cryptographically suitable as the elliptic curve coefficients and the prime are selectively chosen such that the discrete logarithm problem due to them are sufficiently hard.

Table 2.3: Elliptic curve parameter selection criteria

Elliptic curve parameter	Criterion	Benefit(s)
Prime p	1. Crandall prime $2^{\alpha} - \gamma$ *where* $\gamma < 2^{10}$ [**3**, **56**] 2. Montgomery-friendly prime $2^{\alpha}(2^{\beta} - \gamma) - 1$ *where* $\alpha, \beta, \gamma \geq 0$ 3. $p \equiv 3(mod\ 4)$ 4. Mersenne prime $p = 2^{k} - 1$ 5. p=random value 6. Length of $p \geq 221$ bits [**36**]	1. For best possible performance by limiting carry propagation during multiply-reduce and γ is small [**56**]. 2. Accelerates Montgomery arithmetic [**3**]. 3. Such primes can compute modular square root in constant time countering constant time attack using Side channels [**3**]. The point compression method allows representing one point (x, y) of E by only its abscissa x and one bit discriminating between the two possible values $\pm y$. However, recovering y requires computing a square root in $\mathbb{F}_p$. This is easier when $p \equiv 3(mod\ 4)$ since in this case, $c^{(p+1)/2}$ is a square root of c if c is a square [**10**]. 4. Mersenne primes are special primes of unique form which enables fast arithmetic [**3**]. Minimizes time for modular multiplication [**57**]. 5. No pre-studied value or special structure vulnerable to cryptanalysis. 6. To counter brute-force attack.
Continued to next page..		

Table 2.3 – continued from previous page

Elliptic curve parameter	Criterian	Benefit(s)
Coefficient a	1. $a = -3$ 2. a=random value	1. For efficiency reasons. Practically all curves have low-degree isogenies to curves with $a = -3$, so this choice does not affect security. P1363 allows $y^2 = x^3 + ax + b$ without the requirement $a = -3$ [**10**]. 2. No pre-studied value or special structure.
Coefficient b	1. Should not be square in $\mathbb{F}_p$ [**26**] 2. b=random value	1. To avoid compressed representations of elliptic curve points as $(0,0)$ and $(0,x)$ would be identical as $x = \sqrt{b}$ with least significant bit as 0 [**34**]. 2. No pre-studied value or special structure.
Elliptic curve order N	1. N should be prime [**11**, **18**] 2. N should be composite	1. Prime order curve selected to resist Pohlig-Hellman and Pollard's Rho attacks [**2**, **10**]. Small subgroup attacks are avoided [**10**, **11**]. 2. Prime group order curves do not have points with $y = 0$ [**36**]. Special points of the form $(x,0)$ exist if the curve has an even order [**10**].
Base point order n	n should be prime to avoid Weil and Tete pairing attacks [**2**, **10**]	$n > 2^{160}$ *and* $n \nmid (q^k - 1)$ where k is the embedding degree of elliptic curve.
Cofactor h	Preferably 1	For optimal bit security, $h = 1$ though $1 \leq h \leq 4$ for performance gain [**2**, **10**, **58**].
Base Point G	Randomly choosen base point [**1**]	Prime order of base point gives maximum elliptic curve group size.

2.5 Verification Criteria of Standard Short Weierstrass Elliptic Curves

The elliptic curves often rely on the use of special primes or special forms of elliptic curve to gain performance benefits. The special primes like Crandall prime, Mersenne prime, Montogomery-friendly prime etc. as discussed in Table 2.3 are used in Short Weierstrass elliptic curves which attract various known and non-disclosed attacks. Side channel attacks are one of the popularly known techniques which work well on special structures of prime. They also take advantage of weak implementation of the elliptic curve. Therefore, secure implementation of the elliptic curve is essential to defend a particular curve from side channel attacks. The verification criteria [**36**] of Short Weierstrass elliptic curves ensure secure implementation of the elliptic curve and is depicted in Table 2.4.

Here, various elliptic curves and terminologies are given respective notations for representation in Table 2.4. One may read Table 2.4 considering A=NIST recommended elliptic curves, B=Brainpool recommended elliptic curves, C=SECG recommended elliptic curves, D1=ANSSI recommended elliptic curve FRP256v1, D=CM discriminant of elliptic curve, t=trace of elliptic curve, p=prime field order of elliptic curve, n=base point order of elliptic curve, s^2 is the largest square dividing $t^2 - 4p$ to affirm that $\frac{t^2-4p}{s^2}$ is square free negative integer.

<This space is intentionally left blank.>

Table 2.4: Elliptic curve parameter verification criteria

Verification criterion	**Details**	**Supported by the curve**
$safeField$	Prime of the forms 1 mod 4 and 3 mod 4	A, B, C, D1
$safeEquation$	Elliptic curve over prime field possessing either Short Weierstrass or Montgomery or Edward equation	A, B, C, D1
$safeBase$	Possessing prime order of base point	A, B, C, D1
$safeRho$	Rho value must be $\geq 2^{100}$	A, B, C, D1
$safeTransfer$	Safe against additive and multiplicative transfers. Additive transfer protects from Smart-ASS attack [**9**, **43**] whereas multiplicative transfer protects from MOV attack	A, B, C, D1
$safeDiscriminant$	Absolute value of complex-multiplication field discriminant $\|D\| > 2^{100}$ where $D = (t^2 - 4p)/s^2$ if $(t^2 - 4p)/s^2 \ mod \ 4 = 1 \implies D = \frac{t^2-4p}{s^2}$ otherwise $D = 4(t^2 - 4p)/s^2$ [**36**]	A, B, D1
$safeRigid$	Allows only fully rigid and somewhat rigid curves	B, C
$safeTwist$	Same ECDLP security requirements for twist of the elliptic curve.	C
$safeCurve$	Elliptic curve is safe if all the above criteria are met	NIL

Among all the Short Weierstrass elliptic curves, Brainpool recommended curves qualify all the verification criteria except the twist security of its recommended curves. However, SECG curves qualify all the verification criteria except the $safeDiscriminant$ criterion and thus, Brainpool and SECG recommended curves could not qualify overall $safeCurve$ validation. Moreover, NIST and ANSSI recommended elliptic curves have also met almost all the verification criteria except the $safeRigid$ and $safeTwist$ criteria as they did not explain the generation of their curve parameters adequately and attracted criticisms as discussed in Section 2.3.4.

There is no sufficient verification data of NUMS-curves [**37**, **50**] and Russian standardized elliptic curves [**59**] available in public domain and therefore, they are not included in Table 2.4.

2.6 Approaches adopted by Agencies for Elliptic Curve Computation

Many agencies have recommended elliptic curves over various security levels for standardization. Table 2.5 depicts the popular standard elliptic curves in Short Weierstrass form with their generation approaches year wise. Here, randomly generated elliptic curves means those elliptic curves whose parameters like prime p, field coefficients a, b and basepoint $G_{x,y}$ are randomly or pseudo-randomly (a secure hash function is used to generate curve parameters from random value given as input to the hash function to confirm that parameters are indeed computed pseudo randomly) generated or otherwise, they are considered to be obtained from the deterministic approach.

Table 2.5: Computational Approach adopted for Short Weierstrass elliptic curve

Name of elliptic curve	**Security Level in bits**	**Approach**	**Agency**	**Year**
NIST [**32**]	112, 128, 192, 256	Deterministic	National Security Agency (NSA)	2001
Brainpool [**34**, **35**]	128, 192, 256	Psuedo-random	European Consortium of Companies and Government	2005
ANSSI FRP256v1 [**51**]	128	Random	ANSSI	2011
SECG [**33**]	112, 128, 192, 256	Deterministic	Certicom	2000
NUMS-Curves [**37**, **50**]	128, 192, 256	Deterministic	Microsoft Research	2014
Russian Standardized Curves [**59**] GOST R 34.10-2001 GOST R 34.10-2012 GOST R 34.11-2012	128, 256	Deterministic	Russian National Cryptographic Standards	2001, 2012

It is obvious from Table 2.5 that CM method i.e., deterministic approach is mostly adopted by the curve recommending agencies except the Brainpool. Clearly the trend demonstrates that deterministic approach of elliptic curve computation is preferred for standardization purposes mainly citing their performance benefits.

2.7 Review of Previous Elliptic Curves Computational Resource Estimates

In this section, Koblitz's work [**12**] is reviewed who approached the problem of estimating the number of searches required for finding near prime order elliptic curve randomly over $\mathbb{F}_{2^n}$ probabilistically. A very brief on the past conclusions made by many researchers on quantum computations attacking ECDLP is also presented. Moreover, this section mainly focuses on the related work of Roetteler *et. al.* [**39**, **40**] who estimated the number of qubits needed to solve ECDLP over certain prime field size using Shor's algorithm.

2.7.1 Koblitz's Approach to derive Estimates for searching Elliptic Curve randomly over $\mathbb{F}_{2^n}$

Focusing on determining the order of elliptic curves of cryptographic interests, Koblitz [**12**] estimated the probability of drawing a good elliptic curve in characteristic 2 with nearly a prime order in terms of number of attempts. These attempts were made primarily to compute the order of the elliptic curve using Schoof's algorithm repeatedly till an elliptic curve with suitable order is found. This probability estimate with underlying assumptions which are detailed in [**12**], are concluded from the experimental data gathered at Hewlett-Packard Laboratories

[**16**]. The experiment was carried out to generate a large number of elliptic curves over various fields of sizes used in real systems [**16**].

A quick recall on the probability estimate given by Koblitz [**12**] is as follows: Let $\mathbb{E}$ be defined over $\mathbb{F}_{2^n}$ and is given by the equation

$$\mathbb{E} : y^2 + xy = x^3 + a_2x^2 + a_6 \tag{2.4}$$

where n is the bit length of the binary field and whose order is a prime or almost prime. Let it be called B-almost prime where B is some constant such that $d|N$ where d is a prime $\geq N/B$ and by doing some variation in coefficients a_2 and a_6, B-almost primality of $N = |\mathbb{E}|$ was assumed to be same as that of a random even integer of the same order of magnitude (not a proven conjecture but it is assumed). As $N \approx q = 2^n$, therefore, for fixed B and large q, the latter probability is asymptotic to

$$\sum_{j=1}^{B/2} \frac{1}{jlog(\frac{q}{2j})} \approx \frac{1}{n} \times log_2(\frac{B}{2}) \tag{2.5}$$

The equation (2.5) implies that, for a prime factor of N of length larger than 134-bit with $B = 2n - 134$, the number of probable trials to find $\mathbb{E}$ with $|\mathbb{E}|$ divisible by a prime whose length is larger than 40 digits or 134 bits will be $\frac{n}{n-135}$. For example, if we select $n = 160$ then Schoof's algorithm has to run 4 times considering actual determination of order of the curve and order of the twist of the curve simultaneously before a suitable elliptic curve over $\mathbb{F}_{2^n}$ is found. The probability estimates suggested by Koblitz work well with those classes of elliptic curves which are defined over $\mathbb{F}_{2^n}$ i.e. in characteristic 2. Two important questions yet need to be answered:

a. Can we have such resource estimate (i.e. the number of searches or attempts made) for computation of elliptic curves randomly in prime characteristic i.e. those elliptic curves which are defined over large prime fields?

b. Also, can we have resource estimate for processor to be used for computation of elliptic curves randomly over large prime fields?

In this thesis, these questions are answered using statistical estimation approach in Chapter 5.

2.7.2 Status of Elliptic Curve-based Cryptosystems in presence of Quantum Computers

Peter Shor [**29**] hoped that the laws of quantum mechanics will be helpful in building quantum computers. In 1994, the author simulated quantum mechanics on a classical computer leading to construct a polynomial time algorithm for factoring. Author showed that integer factorization and discrete logarithm problems (DLP) in finite fields of prime order can be solved in random quantum polynomial (RQP) time with a permissible small probability of (one-sided) error. This DLP computing polynomial time algorithm generalizes to the cases of elliptic curves as well. A good detail on Shor's quantum algorithms to compute ECDLP can be seen in [**2**]. In 2003, Proos *et. al.* [**60**] have shown implementation of Shor's quantum algorithm for computing discrete logarithm problem due to elliptic curve groups. The authors constructed a table with resource estimate for the number of qubits and time depending on the prime field size of the elliptic curve. In 2016, Wohlwend in his report [**61**], conveyed that presence of quantum computers poses a serious threat to ECC based cryptosystem since elliptic curves are basically abelian groups. The author opined that since quantum computers are still in the evolving stage, ECC will be prolonged to be a great choice in cryptographic applications for a reasonably long time.

In 2017, Roetteler *et. al.* [**39**, **40**] precisely estimated quantum resources for quantum circuits required to compute ECDLP induced by an elliptic curve over

an n-bit prime field using Shor's algorithm. The authors proposed an estimate that a quantum computer can solve ECDLP with at most $9n + 2\lceil log_2(n) \rceil + 10$ qubits using a quantum circuit of at most $448n^3log_2(n) + 4090n^3$ Toffoli gates [**39**]. Their resource analysis was carried out by implementing Toffoli, CNOT, and NOT gates circuits to implement the controlled addition of elliptic curve points known as circuit generation time. The authors simulated large parts of quantum circuits on a classical machine. Their results on resource estimation of qubits to solve ECDLP help to plan and acquire practically available quantum resources to target modern elliptic curve-based cryptosystems over certain prime fields sizes. The authors concluded that attacking elliptic curve cryptography is an easier job than attacking RSA by a quantum computer.

In light of this, one of the affordable ways to address the problem of using elliptic curves in cryptography in presence of quantum computers is to scale up the elliptic curve prime field size to a higher possible extent where quantum attacks may not be feasible within reasonable time and with available number of qubits. But generation of elliptic curves randomly over very large prime fields is a cumbersome task which requires huge computational resources as well as time and therefore, proper estimation and allocation of sufficient computational resources are important.

2.8 Cryptographically Secure Random Number Generators for Kernel Applications

2.8.1 /dev/(u)random

Linux and Android kernels use /dev/random and /dev/urandom which are considered as CSPRNG i.e. the PRNG with inputs (meeting the requirement $R2$)

for randomness generation. The limitations of these CSPRNGs are that they do not have enough entropy in the pool and they are not generating keys larger than the hash function that they used internally [**62**]. /dev/random keeps awaiting for the entropy pool to get sufficiently filled in, which results diminished performance of the generator. /dev/random meets the RNG requirements $R1$, $R2$ and $R3$ but does not meet the $R4$ requirement. Though /dev/urandom has provision for unblocked fast supply of random sequences through unblocking pool of entropy, it faces predictability issues [**63**]. /dev/urandom meets the requirements $R1$ and $R3$ but does not meet the requirement $R2$ and $R4$.

2.8.2 Yarrow

Yarrow [**64**] is a PRNG with true random inputs used by MacOS/iOS/FreeBSD kernels. This CSPRNG is too complex and under-specified in entropy handling context and also slow to provide an initial seed [**62**]. It uses Triple DES block cipher for pseudorandom bitstream generation. Like /dev/random, Yarrow meets the requirements $R1$, $R2$ and $R3$ but does not meet the requirement $R4$.

2.8.3 Fortuna

Fortuna [**65**, **66**] is a popular CSPRNG and a refinement over Yarrow, used by the Windows kernel which uses its entropy effectively. It uses AES-like cipher for the generator with 256-bit size of the block cipher key and a 128-bit counter. Fortuna produces a very good throughput of 20 clock cycles per byte on CPU type PC [**65**] and 7.2 Mbps throughput in software [**66**]. Fortuna implicitly accumulates entropy through hash, partitions the incoming entropy into multiple entropy pools and uses its pools at different rate for output generation in order to guarantee that at least one pool will remain available for use [**67**]. Though Viega [**62**] observed that Fortuna completely foregoes the entropy estimation and, Fortuna and Yarrow both do not

exhibit information-theoretic security as well. Like Yarrow, Fortuna also meets the requirements $R1$, $R2$ and $R3$ but does not meet 'non-reproducibility' i.e., the requirement $R4$.

It is imperative to note that the present kernel CSPRNGs do no meet the requirement of 'non-reproducibility' i.e., the requirement $R4$ which is a crucial feature that helps to prevent the kernel better from exploitation as discussed in Section 1.4.7. In this work, the proposed KCS-PRNG is designed in such a way that all the four requirements ($R1$ to $R4$) of an ideal RNG are met to ensure better prevention of the kernel from exploitation.

2.9 Summary

Short Weierstrass elliptic curves are widely used for cryptographic purposes. An evolution chart of events is presented which has significant impact on introducing elliptic curves for use in cryptography. A comprehensive list about important attacks on ECDLP and their countermeasures is presented in this chapter which became the basic selection criteria of elliptic curves for their consideration in cryptography. Two popular approaches i.e., deterministic and random approaches to compute cryptographically secure Short Weierstrass elliptic curves and rationale behind them are evaluated in detail which favoured random approach for the elliptic curve implementation in the kernel applications. This chapter also lays the foundation for trusted elliptic curves which are discussed in Chpater 4 of the thesis. The rationale behind selection criteria and verification criteria to compute cryptographically suitable elliptic curve parameters are also discussed. A trend of approaches for computation of elliptic curve parameters for cryptographic purposes is demonstrated in this chapter which favoured deterministic approach in standardization so far.

Further, Koblitz's estimate to determine the number of searches needed probabilistically to randomly search a near prime order elliptic curve over $\mathbb{F}_{2^n}$ where n is the bit length of the binary field is reviewed in this chapter. This thesis gets motivation from his work and presents novel statistical derivation of such estimate over $\mathbb{F}_p$ where p is a large prime in Chapter 5. Additionally, the processor estimate in terms of the number of CPU clock cycles required to randomly obtain a prime order elliptic curve is also presented as one the research outcomes of this thesis in Chapter 5.

Hence, it is inferred that this comprehensive evaluation and analysis of computational approaches of cryptographically safe elliptic curves will be helpful to those who wish to compute Short Weierstrass elliptic curves for cryptosystems design, in particular, in the design of kernel CSPRNGs with desired properties of the underlying elliptic curves.

Further, in the last section of this chapter, three popular kernel CSPRNGs namely /dev/(u)random, Yarrow and Fortuna were reviewed and it was observed that all of them meet the randomness requirements $R1$, $R2$ and $R3$ but they do not meet the randomness requirements $R4$ which is very crucial for strategic applications such as kernel applications. This thesis covers this critical issue based on the observations made in the survey conducted and resolves it in Chapter 6. The next chapter enlists various problem statements which are observed in this chapter.

Chapter 3

Problem Statements

"A problem well stated is a problem half solved."

- John Dewey

In this chapter, seven research problems are discussed in context of Short Weierstrass elliptic curves and the random number generation in operating system kernels. Based on the survey carried out in Chapter 2, these seven problem statements are ellaborated in two parts of this chapter.

3.1 Part I: Evaluation and Computation of Novel Short Weierstrass Elliptic Curves

Elliptic curves over large prime fields are considered to provide provable security to the cryptographic schemes. Six important problems are encountered in computation of cryptographically secure elliptic curves over prime field which are stated as below:

3.1.1 Problem 1

Evaluation of computational approaches and selection criteria of elliptic curves over prime fields from computation, security and trust perspectives.

A comprehensive evaluation of standard computational approaches and selection criteria of cryptographically secure elliptic curves over the prime fields is presented in Chapter 2 of the thesis. Chapter 2 of the thesis also recommends the preferable computational approach and selection criteria of desired elliptic curves for their implementation in critical cryptosystems of strategic nature.

3.1.2 Problem 2

Computation of cryptographically secure as well as **trusted** elliptic curves over the prime fields.

Chapter 4 discusses about the trust issue of the present standard elliptic curves recommended by various international bodies like NIST, Brainpool, SECG etc. in detail. Chapter 4 also introduces a new security notion called the *trusted security* of elliptic curves and proposes three *trusted security acceptance criteria* to ensure elimination of any possible computational manipulation of the elliptic curve parameters.

3.1.3 Problem 3

Recommendation of new elliptic curves over large prime field sizes whose method of generation is trusted and cryptographically strong.

Chapter 4 recommends two new elliptic curves over 256 bit and 384 bit prime field sizes which are cryptographically secure as well as trusted for implementation in crucial cryptographic applications such as in kernel applications.

3.1.4 Problem 4

Computation of cryptographically secure elliptic curve over large prime field is an intricate and resource intensive task. This leads to two critical problems with respect to estimation of computational resources requirement for randomly deriving elliptic curves over large prime fields which are stated in Problem 4 and Problem 5 respectively.

> To provide an estimate of computational resources in terms of computing processor i.e., number of the CPU clock cycles to compute cryptographically safe elliptic curve randomly over desired prime field size for cryptographic purposes.

Chapter 5 addresses Problem 4 and provides precise statistical estimate of CPU processor in terms of CPU clock cycles required for computation of cryptographically secure elliptic curves randomly over large prime fields.

3.1.5 Problem 5

> To provide an estimate of computational resources in terms of number of attempts or searches to be made in the security parameter space of the elliptic curve to compute cryptographically safe elliptic curve randomly over desired prime field size for cryptographic purposes within stipulated time.

Chapter 5 addresses Problem 5 and provides precise statistical estimate of number of searches or attempts required for computation of cryptographically secure elliptic curves randomly over large prime fields.

3.1.6 Problem 6

In addition, an important question comes into light that how the present elliptic curve cryptography-based applications will be able to co-exist with future quantum computers having certain number of qubits. It was shown by Roetteler et. al. [**39**, **40**] that a quantum computer with certain number of qubits can break ECDLP imposed by the elliptic curves over a certain prime field size. Hence, there is a need felt for recommendation of the reasonable amount of CPU processor and stipulated timelines respectively required for computation of cryptographically secure elliptic curve over desired prime field size which can co-exist in presence of the quantum adversaries.

> To estimate computational investment for cryptographically secure elliptic curves over very large prime fields in order to verify feasibility and to prepare existing ECC-based cryptosystem to be kept resilient to quantum attacks using available number of qubits.

Chapter 5 also addresses Problem 6 and provides a tabular comparison of requirements of the elliptic curve field sizes which will be resilient against certain number of qubits under quantum attack.

We address above mentioned six problems in Part I of this thesis which are covered across Chapter 2 and Chapters 4 - 5.

3.2 Part II: Construction of a Novel CSPRNG Using Short Weierstrass Elliptic Curves For Kernel Applications

The operating system kernel is the lowest level software interacting with the hardware and user programs. The kernel needs software based random number generator (RNG) which can essentially generate statistically validated, unpredictable as well as non-reproducible bitstreams for its critical kernel operations such as Address Space Layout Randomization (ASLR), safe storage of users' passwords and cryptographic key generation, etc. As software based RNG are based on deterministic algorithms, therefore, non-reproducibility property of the generated bitstreams has not been so far possible for sensitive kernel applications in a computer.

3.2.1 Problem 7

A new competitive candidate CSPRNG for kernel or cryptographic usage is highly desirable which could exhibit statistical properties of randomness and unpredictability along with the non-reproducibility property of randomness.

Chapter 6 gives the proposal of new CSPRNG for kernel applications called as KCS-PRNG which is proven to be a viable CSPRNG candidate for adoption in the operating sysem kernels in the thesis.

We address the seventh problem in Chapter 6 in Part II of this thesis.

Part I

Evaluation and Computation of Novel Short Weierstrass Elliptic Curves

Chapter 4

The Proposed Cryptographically Secure and Trusted Elliptic Curves Over 256 bit and 384 bit Prime Fields

"Consider now the possibility that one in a million of all curves have an exploitable structure that "they" know about, but we don't.. Then "they" simply generate a million random seeds until they find one that generates one of "their" curves. Then they get us to use them."

- Michael Scott

In this chapter, a new security notion called *trusted security* of elliptic curve is proposed. Addtionally, two new Short Weierstrass elliptic curves over 256 bit and 384 bit prime field sizes are recommended for cryptographic purposes as the solutions to Problem 2[1] and Problem 3[2] as mentioned in Chapter 3.

[1]Computation of cryptographically secure as well as trusted elliptic curves over the prime fields.

[2]Recommendation of new elliptic curves over large prime field sizes whose method of generation is trusted and cryptographically strong.

4.1 Publications from this chapter

The research outcomes of this chapter contributes the following publications:

1. **Kunal Abhishek** and E. George Dharma Prakash Raj, *Evaluation of Computational Approaches of Short Weierstrass Elliptic Curves for Cryptography*, Cybernetics and Information Technologies (2021). (DOI: 10.2478/cait-2021-0045)

2. **Kunal Abhishek** and E. George Dharma Prakash Raj, *Computation of Trusted Short Weierstrass Elliptic Curves For Cryptography*, Cybernetics and Information Technologies (2021). (DOI: 10.2478/cait-2021-0020)

4.2 Introduction

Short Weierstrass elliptic curves are considered to be as secure for cryptography as the underlying hardness of their elliptic curve discrete logarithm problem i.e., ECDLP which is defined as finding a scalar k knowing any two points P and Q on elliptic curve $\mathbb{E}$ holding the relation $Q = kP$. This is known as the ECDLP security of the selected elliptic curve when used for cryptography [**36**]. The most efficient publicly known method to solve ECDLP or break the ECDLP security is the Pollard's Rho algorithm which takes approximately $0.886 \times \sqrt{n}$ point additions where n is the base point order [**36**, **51**]. One must select an elliptic curve which is ECDLP secure for cryptographic applications. Another notion of security for selecting suitable elliptic curves for cryptography is known as elliptic curve cryptography security i.e., ECC security in short, the term coined by Bernstein and Lange [**36**] which ensures prevention from any information leakage from the implementation flaws of the elliptic curve. Most of the popular standards today such as National Institute of Standards and Technology (NIST) [**32**], Brainpool

[**35**], Standards for Efficient Cryptography 2 (SEC2) [**33**], IEEE P1363 [**68**] etc. recommended those elliptic curves which are ECDLP secure and attain some sort of ECC security (for only some standard curves [**36**]). It is worthwhile to note that an ECC based cryptosystem can be compromised by either compromising the ECDLP security or the ECC security. All the present day standards have recommended Short Weierstrass elliptic curves keeping either or both of these security notions into consideration. This chapter introduces a critical security notion which we call as "trusted security" of elliptic curves which ensures that the selected elliptic curve is free from any manipulation from its computation perspective and can be trusted for use in cryptographic applications. The trusted security notion of computation of elliptic curves minimizes the risks involved in generation of safe curve parameters deterministically where they are vulnerable to (intentionally) non-disclosed attacks with (intentionally) non-disclosed properties of the curve parameters. In such cases, the ECDLP can be solvable by using very efficient sub-exponential or polynomial time algorithm using non-guessable high computing power.

The key outcomes of this chapter are as follows:

- Introduction of a new security notion called as "trusted security acceptance criteria" as an important security evaluation criterion along with the ECDLP security and ECC security criteria for computation of Short Weierstrass elliptic curves aimed for cryptography. The chapter also includes evaluation of standard Short Weierstrass elliptic curves from trust perspective.

- Argument that trust in generation method of elliptic curves can be achieved only through computation of the curve parameters randomly without considering any of their pre-studied values such as -3 or Mersenne primes etc. The randomly selected elliptic curve parameters can be derived using any good quality user trusted random number generator (RNG) along with competitive performance of the elliptic curve.

- Demonstration of two new elliptic curves defined over 256 bit and 384 bit prime field sizes respectively for cryptography which are secure from ECDLP security, ECC security as well as trusted security perspectives and evaluation of these proposed elliptic curves with respect to cryptographic key pair generation, signing and verification from performance perspective.

4.3 Discussion on Distrusted Standardized Elliptic Curves

It is important to select those elliptic curves which are cryptographically secure and trusted for constructing cryptographic systems. Transport layer security (TLS), secure shell (SSH) and Internet Protocol Security (IPSec) [**28**], public key infrastructure (PKI) [**27**] etc. are some of the popular applications which require safe elliptic curves in their cryptosystem design. Most of such commercial applications use standard elliptic curves over prime field of 256 bit sizes for sufficient security and interoperability purposes. However, Bernstein et. al. [**51**] has recently pointed out some mechanisms such that a new elliptic curve can be proposed to sabotage the public standards. They demonstrated convincing methods by which they were able to implant vulnerability in the elliptic curves known as BADA55 curves by utilizing the gain of many bits of freedom [**51**] which satisfies the public standards and can be put forward for standardization to fool the users. This essentially proves that an attacker can exploit unknown (his known) vulnerability to sabotage existing public standards and justify his selection of elliptic curve parameters citing performance gain and his own way of getting randomness i.e. verifiably random etc. which is used in the generation of the vulnerable curve parameters. Bernstein et. al. [**51**] comprehensively demonstrated how a wrong or non-trustable elliptic curve can be derived using the procedure

led by the public standards and their recommended public criteria. They showed that plausible variations in the Brainpool curve generation procedure and Microsoft curve generation procedure respectively can be used to sabotage public standard. Further, the Agence nationale de la securite des systemes d'information (ANSSI) standard recommended FRP256V1 elliptic curve which has low twist security of order 279 which means that there are 279 elliptic curve additions required to mount the twist attack to get user's secret key [**51**]. Also, there is no reasonably sufficient documentation available for this curve. Furthermore, Bernstein et. al. demonstrated computation of the BADA55-R-256 curve which meets the public security criteria for ECDLP security and ECC security but still being a manipulated curve. Finally, it is understood that computation of an elliptic curve can be manipulated by any deterministic method of computation of the curve parameters and variety of reasons can be cited with selection of the curve parameters adhering to some public standard of proposer's convenience. Summarizing, the problems pertained with the trust factor consists of one or more issue(s) from the following:

- No sufficient explanation on the RNG used for seed or randomness generation.
- Intentional variation in standard elliptic curve generation procedure recommended by the curve proposing agencies by themselves.
- Intentional hiding of information about the curve parameters even providing detailed documentations on curve generation process of standard elliptic curves.
- Sabotaged standards.
- Root problem of the lack of trust is the deterministic approach adopted by all the agencies in standardizing their proposed elliptic curves.

With the above prevalent issues, an obvious question arises that **"because you can explain, does not mean that you will explain everything"**. This question is answered by introducing a set of three important security evaluation criteria in the thesis called "trusted security acceptance criteria" for computation of suitable elliptic curves for cryptography which can be additionally invoked along with the ECDLP security and ECC security criteria to mitigate the trust issues in curve generation process to a great extent. Before proceeding further, it is important to get an insight into the usability of standard and non-standard i.e., self-derived elliptic curves to know their purpose and benefits of using them under appropriate circumstances. Following section gives a new insight for the same.

4.4 Standard Elliptic Curves and Non-standard Elliptic Curves

Elliptic curves are standardized to enable compatibility and interoperability across diverse applications. Moreover, non-standard elliptic curves are mostly used by strategic or military applications and sometime non-military but other critical infrastructures applications such as Command and Control systems of nuclear reactors etc. These applications do not really believe in Kerckhoffs's principle [**69**] which says "A cryptographic system should be secure even if everything about the system, except the key, is public knowledge.". Unlike Kerckhoff's principle, the strategic applications do believe that not only the keys but the algorithm should also be kept private to protect critical information infrastructure better. In such cases, they compute elliptic curves preferably using random approach instead of deterministic approach. The thesis contributes a new insight to observe some remarkable differences between the standard and non-standard elliptic curves from computation, trust and security perspectives as portrayed in Table 4.1.

Table 4.1: Comparison of Standard Elliptic Curves with Non-standard Elliptic Curves

Standard Elliptic Curve	Non-standard Elliptic Curve
Prefers deterministic approach of computation to get performance benefits in elliptic curve arithmetic. This helps in standardization of elliptic curves by global acceptance.	Prefers random approach of computation for long term security so that any special kind of curve is avoided which may lead to vulnerability to an unanticipated attack.
Adheres to Kerckhoffs's principle of security and fixes elliptic curves for compatibility and interoperability among diverse applications across the globe.	Adheres mostly to strategic principle of security which says that keys and algorithm both need to be kept secret.
Standard elliptic curves are subject to public exposure and often attract cryptanalysis as more people use it. Hence, there is always a high chance of collision with the secret key [**70**].	Negligible chance of collision with the secret key that's why random approach is preferred.
Distrust comes with presence of special structures of the curve parameters.	Trusted new values of curve parameters known to designer only. Prefers random approach to compute elliptic curve parameters.
Standard elliptic curves are globally accepted and trusted.	Not published and mostly not supported by the standards. Hence, trusted by their proposers or/and in closed group only.
	Continued to next page..

Table 4.1 – continued from previous page

Standard Elliptic Curve	Non-standard Elliptic Curve
Compatible across applications and interoperable due to standardization.	Not compatible. Applications need to be made interoperable explicitly.
Better approach in case where elliptic curve needs to be computed over large prime fields.	Better approach in case where elliptic curve needs to be transparently computed without any special structures known to others.
Curve parameters and compression techniques have patent issues.	No patent issues.
Already published and analyzed thoroughly. Non-deniable chances of hiding backdoors.	Derivation procedure of curve parameters are known to the proposers only and hence, negligible chances of backdoors. High degree of trust observed by the proposers of non-standard elliptic curves.
Standard elliptic curves are fixed to maintain compatibility among applications.	Non-standard elliptic curves have edge over the standard ones as they can be replaced frequently for added security.
More prone to get attacked by sophisticated advancements in mathematics and discoveries.	In case of randomly selected curve parameters, curve is safe until sub-exponential algorithm is known to break it in particular [**3**].

4.5 Trusted Security Acceptance Criteria for Elliptic Curves for Cryptography

Standard elliptic curves followed deterministic approach in computation of their coefficients and primes. Most of them used pre-studied values whose credibility and trustworthiness are doubted [**51**, **71**, **72**, **73**] due to origination of the curve parameters and lack of proof for the randomness used in the curve generation process such as use of computationally convenient primes like powers of two etc. Hence, there is a need to introduce additional security acceptability criteria to invoke trust in the computation of elliptic curve parameters for use and in standardization. In this chapter, a set of three new security evaluation criteria of cryptographically safe elliptic curve called the "trusted security acceptance criteria" for elliptic curves used for cryptography is introduced which are as follows:

1. *T*1: *User trusted random number generator (RNG) to provide (pseudo)randomness.*

   A RNG should be selected preferably by its user for assuring that user is fully aware of the technicality of the RNG and hence he/she trusts it completely. Apart from the trust aspect, the RNG should adhere to the following properties as indicated by Koc [**22**] and Schneier [**24**] and discussed in Section 1.4.6:

   - The bitstream generated by a pseudo random number generator (PRNG) or cryptographically secure PRNG (CSPRNG) should be statistically sound i.e., it has a large period.
   - The bitstream generated should be unpredictable i.e., the RNG should be forward secure as well as backward secure.

The curve parameters should be chosen randomly in a trustworthy way to avoid any uneasy explanation about the generation of the curve constants and hence, the requirement of user trusted and strong RNG is critical in trust building.

2. $T2$: *No pre-studied values of the curve coefficients and prime.*

   The well-known constants are accepted by everyone without hesitation but their non-exposed property may be used for construction of vulnerable elliptic curves. BADA55-VPR-224 is such an example which used $cos(1)$ constant [**51**]. The elliptic curve coefficients a, b must not use any pre-studied values to avoid the scope of manipulation. Moreover, the prime field order p can only have special structure if it is randomly selected with suitable size (normally ≥ 224) bits for fast reduction on the elliptic curve.

3. $T3$: *Reproducibility of new elliptic curves of nearly the same cryptographic strength and suitability using the same method and apparatus.*

   One must get new elliptic curves of nearly the same cryptographic strength using the same method and apparatus. The Pollard's rho values of the elliptic curves and their respective twisted curves are considered as the measurement of their cryptographic strengths which is the number of elliptic curve point additions to solve the ECDLP. Generally, $0.886 \times \sqrt{n}$ elliptic curve point additions are required to break the ECDLP where n is the order of the base point [**36**, **51**].

<This space is intentionally left blank.>

4.6 Evaluation of Standard Elliptic Curves from Trust Perspective

Standard Short Weierstrass elliptic curves claimed to have followed rigorous ECDLP security validations and sometime ECC security validations together to arrive at the curve parameters for recommendation. They claimed that they used seeds which were randomly generated and some of them adhered to verifiably random way of obtaining the curve parameters. Table 4.2 evaluates standard elliptic curves from trust perspectives for use in cryptography:

Table 4.2: Evaluation of standard Short Weierstrass elliptic curves from trust perspectives

Elliptic curve	**Trusted Security** ($T1, T2, T3$)	**Remarks**
NIST P224r1	None	Deterministic approach with pre-studied coefficients and prime [**32**]
NIST P256r1	None	Deterministic approach with pre-studied coefficients and prime [**32**]
NIST P384r1	None	Deterministic approach with pre-studied coefficients and prime [**32**]
secp224r1	None	Special structure of prime p (Mersenne prime) and insufficient documentation [**33**]
secp256r1	None	Special structure of prime p (Mersenne prime) and insufficient documentation [**33**]
secp384r1	None	Special structure of prime p (Mersenne prime) and insufficient documentation [**33**]
secp521r1	None	Special structure of prime p (Mersenne prime) and insufficient documentation [**33**]
ANSSI FRP256v1	None	Pre-studied value of coefficient a and insufficient documentation [**51**, **74**]
Brainpool	$T2$	None of the Brainpool curves are generated by their own stipulated procedure [**51**, **35**]
NUMS curves	None	Deterministic approach with pre-studied coefficients and prime.[**51**, **3**]

It is imperative to note from Table 4.2 that, there is an ardent need for new

elliptic curves which are cryptographically secure as well as trusted. Following section will focus on the generation details of trusted Short Weierstrass elliptic curves to be used for cryptography.

4.7 Cryptographically Secure Elliptic Curve Generation using the Proposed Trusted Security Acceptance Criteria

Short Weierstrass elliptic curves can only exhibit prime order [**50**] which does not loose any bit of security of ECDLP [**3**]. However, elliptic curves of cryptographic interests must get validated against their ECDLP security, ECC security as well as trusted security. It is now observed from previous sections that random approach of computing safe elliptic curves is the only way to achieve all of these three security notions. A standard procedure is shown as the flow chart in Figure 4.1 to get bird's eye view of generation of the trusted Short Weierstrass elliptic curves intended for cryptography.

<This space is intentionally left blank.>

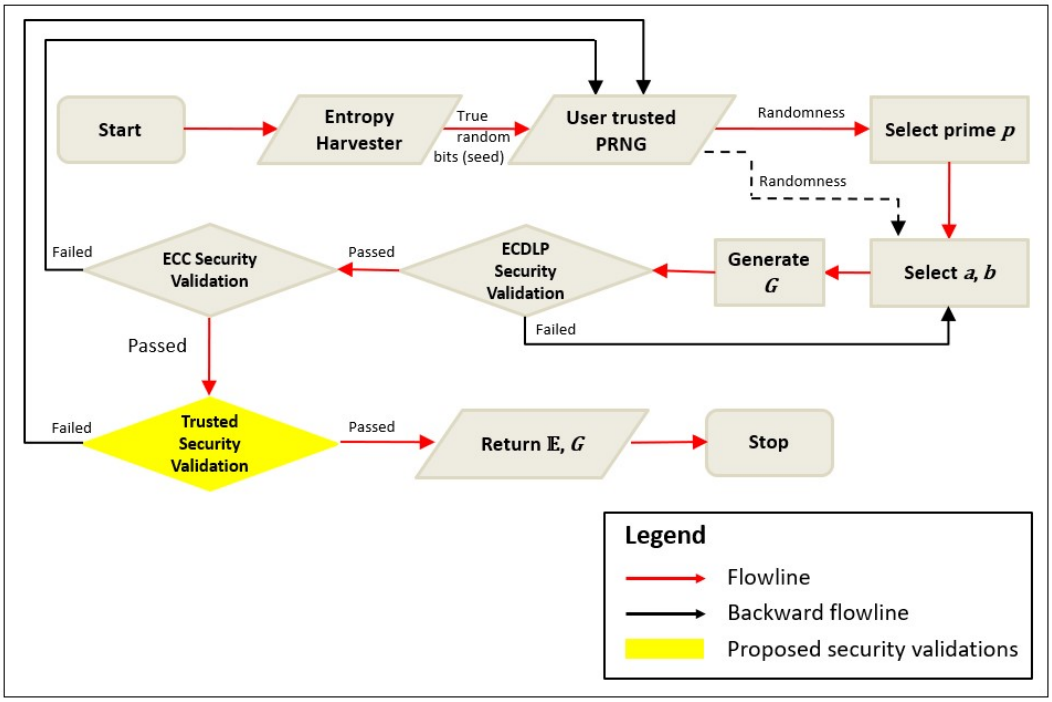


Figure 4.1: Flow chart of generation of cryptographically secure and trusted Short Weierstrass elliptic curve

An entropy harvester which is used to obtain sufficient number of true random bits from various physical noise sources like device randomness, disk randomness, Human Interface Device (HID) (key board, mouse, etc.), interrupt randomness, etc. is used to seed a user trusted (means user is aware of the technicality of the RNG and associated security risks completely) PRNG/CSPRNG as depicted in Figure 4.1. The user trusted PRNG supplies desired number of (pseudo)random bits to generate suitable p, a and b. An elliptic curve $\mathbb{E}$ is constructed over prime field $\mathbb{F}_p$ (where p is fixed in this case, but one can choose other way also to generate suitable elliptic curves by fixing the curve order N randomly etc.) with coefficients a and b. Now $\mathbb{E}$ is subjected to ECDLP security validation failing which will re-generate the coefficients a and b until it gets suitable curve coefficients for $\mathbb{E}$ to be ECDLP secure. A base point $G_{x,y}$ is also selected randomly over elliptic curve $\mathbb{E}$ and gets verified for its prime order for acceptability. Once $\mathbb{E}$ is validated for ECDLP security, it is subjected to security validation from ECC security perspective which

expects $\mathbb{E}$ to have its twist $\mathbb{E}'$ also to be as secure as $\mathbb{E}$ is. In case of the fact that ECC security validation does not pass, one needs to re-generate the prime p and subsequently coefficients a and b to get ECDLP security and ECC security validated successfully. Finally, the ECDLP secure and ECC secure elliptic curve $\mathbb{E}$ is verified with the proposed trusted security acceptance criteria (indicated in yellow decision box in Figure 4.1 failing which the process is re-initiated with deriving prime p and coefficients a and b as fresh until one gets an acceptable $\mathbb{E}$. Lastly, $\mathbb{E}$ and G are returned as the output. The elliptic curve generation procedure is detailed in Algorithm 3.

4.7.1 Assumptions

Following assumptions are made while computing the curve parameters using Algorithm 3:

i. User trusted cryptographically strong RNG is available to provide randomness required in computation of secure elliptic curve.

ii. Sufficient entropy is available in the system. Generally, more than 2000 bits of entropy is expected to be available with the system to seed the RNG sufficiently to uninterruptedly generate elliptic curves up to over 384 bit prime field sizes. Also, the operating system is not used for the first time after installation as sufficient entropy will not be available with the machine.

iii. Compilers, CPU Processors, SAGE and other participating modules in the curve parameter generation are trusted.

<This space is intentionally left blank.>

Algorithm 3 Generation of cryptographically safe and trusted Short Weierstrass elliptic curve

Require: Prime field size (l) in bits and randomness from user trusted RNG
Ensure: Trusted cryptographically safe elliptic curve $\mathbb{E}$ over prime field $\mathbb{F}_p$ with base point $G_{x,y}$
1: Input prime field size l in bits
2: Obtain seed S as true random bits of desired length from entropy harvester
3: Set seed S for user trusted RNG
4: Select randomly prime p such that $p \equiv 3 \ mod \ 4$ ▷ for fast arithmetic on $\mathbb{E}$
5: Choose randomly the coefficient a of $\mathbb{E}$
6: Choose randomly the coefficient b of $\mathbb{E}$
7: Construct the elliptic curve $\mathbb{E}$ with curve parameters p, a and b
8: **Enforce ECDLP security validation:**
9: **if** discriminant = $4a^3 + 27b^2 \neq 0$ AND curve order N is prime AND $\mathbb{E}$ is non-anomalous case AND $\mathbb{E}$ is not supersingular curve **then**
10: continue
11: **Else** go to step 5
12: **end if**
13: Generate randomly the base point $G_{x,y}$ on $\mathbb{E}$
14: **if** base point order n is prime **then**
15: continue
16: **Else** go to step 13
17: **end if**
18: **if** cofactor is 1 AND Pollard's rho value $< 2^{100}$ AND embedding degree $k \geq \frac{(N-1)}{100}$ **then**
19: continue
20: **Else** go to step 5
21: **end if**
22: **Enforce ECC security validation:** ▷ If $\mathbb{E}$ is twist secure i.e., all validations in step 8 applied to the twist $\mathbb{E}'$
23: **if** twist discriminant of $\mathbb{E} = 4a^3 + 27b^2 \neq 0$ AND order of $\mathbb{E}'$ i.e. N is prime AND $\mathbb{E}'$ is non-anomalous case AND $\mathbb{E}'$ is not supersingular curve **then**
24: continue
25: **Else** go to step 4
26: **end if**
27: Generate randomly the base point $G'_{x,y}$ on $\mathbb{E}'$
28: **if** base point order n' is prime AND cofactor of $\mathbb{E}'$ is 1 **then**
29: continue
30: **Else** go to step 4
31: **end if**
32: **if** cofactor is 1 AND Pollard's rho value of $\mathbb{E}' < 2^{100}$ AND embedding degree of $\mathbb{E}'$ i.e. $k' \geq \frac{(N-1)}{100}$ **then**
33: continue
34: **Else** go to step 4
35: **end if**
continued to next page..

Algorithm 3 Generation of trusted.. (continued from previous page)

36: **Enforce trusted security validation:**
37: **if** RNG is trusted **then** ▷ Proposed validation criterion $T1$
38: continue
39: **Else** go to step 2
40: **end if**
41: **if** coefficients a and b have no pre-studied value **then** ▷ Proposed validation criterion $T2$
42: continue
43: **Else** go to step 2
44: **end if**
45: **if** elliptic curves with similar cryptographic strength can be generated with the same method and apparatus **then** ▷ Proposed validation criterion $T3$
46: continue
47: **Else** go to step 2
48: **end if**
49: **return** $\mathbb{E} : p, a, b$ and $G_{x,y}$

4.7.2 Standard Procedure for Elliptic Curve Generation including Trusted Security Acceptance Criteria

The standard procedure shown in Algorithm 3 along with the proposed trusted security acceptance criteria as discussed in Figure 4.1 with detailed security validations of elliptic curve from ECDLP security, ECC security and trusted security perspectives.

The elliptic curve field size (l) in bits is taken as the input in step 1 in Algorithm 3. A seed S is extracted from the entropy harvester in step 2. The /dev/random is used as the PRNG which takes true random bits through a hardware based RNG (HRNG) that extracts entropy directly. The /dev/random PRNG is available with Linux Fedora kernel version 4.13.9 for obtaining randomness in desired bit lengths. The HRNG uses various noise sources like input randomness, device randomness, disk randomness, HID (key board, mouse etc.), interrupt randomness to provide random bits as the seed S to /dev/random in step 3. S is used to initialize /dev/random to provide randomness to the curve generation process as and when

required. As the curve generation program needs a user trusted secure RNG, it is left to the user to select his/her trusted RNG for fulfilling the randomness requirements. Here the focus is to recommend users to use their own trusted RNGs to avoid any possible manipulation in curve computation and here, the demonstration is made to show how a trusted Short Weierstrass elliptic curve can be generated for cryptography. In step 4, the prime p of user desired l bit length is randomly selected and subsequently, checked that it should hold the form of $p \equiv 3 \ mod \ 4$ for fast reduction i.e., fast elliptic curve arithmetic on $\mathbb{E}$. It is noted that p is first chosen randomly and then verified for this form to avoid any pre-studied value. The curve coefficients a and b are then chosen randomly in step 5 and step 6 respectively using different seeds i.e., a and b have independent initializations. Now, an elliptic curve $\mathbb{E}$ is constructed with p, a and b in step 7.

The ECDLP security validations are enforced in steps 8 to steps 22 which includes validations of non-singularity, prime curve order, non-anomalous property, non-supersingularity in step 9 whereas the random selection of base point in step 13 with prime base point order in step 14. The elliptic curve is validated for having small cofactor as 1, high Pollard's rho and high embedding degree in 18 respectively. The non-singularity of elliptic curve confirms that curve is smooth and indeed an elliptic curve [**19**, **75**, **76**]. Prime order elliptic curve with order N is resistant to Pohlig-Hellman attack when $N \geq 2^{160}$ [**2**]. Non-anomalous case of elliptic curve i.e., when curve order $N \neq p$, confirms that curve is resistant to pairing based attacks [2]. Non-supersingularity of elliptic curve prevents the ECDLP from the Menezes, Okamoto and Vanstone (MOV) reduction attack with sub-exponential complexity which takes place when the conditions that p divides trace t or/and $t^2 = 0,\ p,\ 2p,\ 3p$ or $4p$ are met [**4**, **7**]. The cofactor value determines the cryptographic security and gives maximum security when selected as 1 [**2**, **7**]. The Pollard's rho value of elliptic curve determines the number of elliptic curve

point additions to find a collision. This check is very important as a parallelized Pollard-rho on r processors can solve ECDLP in steps [**2**, **42**]. The embedding degree of elliptic curve $k \geq 20$ is considered sufficient to guarantee intractability of the discrete logarithm problem in the extension field [**6**].

The ECC security validations are enforced in step 9 of Algorithm 3 in which it looks for the twist of the selected elliptic curve to be secure against all the ECDLP security validations as described above. The twist security of elliptic curve prevents from any implementation flaws or information leakage about the user's secret key [**36**].

The trusted security validations are carried out in step 36 to ensure the method of generation of elliptic curve is trusted in terms of the randomness used in the curve generation process and that the curve parameters are drawn randomly. Step 37 confirms that the RNG used for randomness is trusted by the user and the curve parameters have no pre-studied or known values. It also ensures that the procedure described in Algorithm 3 can be used to obtain Short Weierstrass elliptic curves of nearly the same cryptographic strength each time on its execution which is shown in step 45. Finally, a trusted and secure elliptic curve $\mathbb{E} : p, a, b$ and base point G is returned as the outcome in step 49.

4.7.3 Creation of Database of Trusted and Secure Elliptic Curves

A database of 500 elliptic curves over 256 bit prime field were created using the proposed method as discussed in Section 4.7.2. It took around six months time using computational resources and programing tools as mentioned in Section 4.8.1 in the creation of the database. All the elliptic curves have undergone thorough security analysis for their cryptographic security which is discussed in Section 4.9 with Pollard's Rho value in the range of 127.0 to 127.8 on an ideal 128

point scale. The database is kept growing by adding more such elliptic curves in order to supply cryptographically secure and trusted curves for generation of non-reproducible pseudorandom bitstreams by the proposed KCS-PRNG which is comprehensively discussed in Chapter 6 of the thesis.

4.8 Demonstration of Trusted Short Weierstrass Elliptic Curves

Algorithm 3 is used to derive and propose two trusted Short Weierstrass elliptic curves KG256r1 and KG384r1 defined over 256 bit and 384 bit respectively for demonstration. The details of the proposed KG256r1 and KG384r1 are shown in Table 4.3 and Table 4.4 respectively. These elliptic curves have undergone security analysis in Section 4.9 to ensure that the elliptic curves generated using Algorithm 3 have nearly the same cryptographic strength in terms of Pollard's rho complexity and other criteria like big discriminant, embedding degree, trace etc. while being compliant with the three security notions i.e. ECDLP security, ECC security and trusted security.

<This space is intentionally left blank.>

Table 4.3: The proposed KG256r1 elliptic curve

KG256r1	
p	105659876450476807015340827963890761976980048986351025 435035631207814085532543
a	577801306981151765834884991713447710888985073378732385 9040095537112968513882 6
b	102451950841073747949316796495896937960702115486975363 7 9832359679732709081346 2
N	105659876450476807015340827963890761976544313325663770 76239923539474412135987 1
$G_{x,y}$	(538516633311464649781099807461241598582198637115148595 4545860140786887919600 64, 884401665317899467231260835467506331798660390928837647 8404161106554792616 0 80)
h	1 (smallest cofactor)

Table 4.4: The proposed KG384r1 elliptic curve

KG384r1	
p	308504936566801493400799664217561138887972017059009663818 4 028808688880241117658797202073501252346926756450542076405 1
a	268937684885793435941799884521325825414071666675195106719 6 901653139051892648485257788827989185822359193013251735562
b	282679914441081045194064979674986566053141057529253438397 67 457243307490975823954516383546612702801272783656774839 39
N	308504936566801493400799664217561138887972017059009663818 4 143875468390039007761732356555487299607397910376591752273 1
$G_{x,y}$	(263821674697227290786867915392591910846306526222054061903 02 146794523414127451183423914120811487055055064792875345576, 202628051316606152195895866462280785015451818341996421 5 119410208934492729588985729356398912702026002012200240404520 4)
h	1 (smallest cofactor)

4.8.1 Resources used

The curve generation programme is written in Python language using Python version 2 and Python version 3.6 compilers ran on a desktop server having 2*Intel(R) Xeon(R) E5-2620v4 processor with 32 CPU cores, 2.1 GHz clock frequency and 128 GB DDR4 memory. The desktop server is equipped with Linux Fedora operating system (kernel version 4.13.9) and SAGE version 8.1 is used for number theory arithmetic support for the curve generation program.

4.9 Security Analysis of the Proposed KG256r1 and KG384r1 Elliptic Curves

4.9.1 Analysis of the ECDLP and ECC Security of the Proposed KG256r1 and KG384r1 Elliptic Curves

SafeCurves verification script [**36**] is used to verify ECDLP security and ECC security of the elliptic curve parameters. Algorithm 4 describes the SafeCurves verification script which was used to verify the KG256r1 and KG384r1 elliptic curves against its ECDLP and ECC security.

It is obvious that ECDLP security is a crucial security requirement for qualifying any elliptic curve for cryptography. However, SafeCurves [**36**] proposed ECC security as another security notion for evaluating elliptic curves to ensure that the ECC implementations do not reveal or leak information about user's secret key. For Short Weierstrass elliptic curves, a twist secure elliptic curve can prevent ECC implementation flaws such as invalid-curve attacks and twist attacks. The elliptic curve $\mathbb{E}'$ is twist secure if its twist $\mathbb{E}'$ is secure which means that all the ECDLP security validations are also successfully compliant by $\mathbb{E}'$.

Algorithm 4 Verification of the proposed elliptic curve parameters for cryptographic security

Require: Elliptic curve parameters $p, a, b, N, G_{x,y}$
Ensure: Safe/Weak Elliptic Curve

if shape of elliptic curve is Short Weierstrass **then**
    continue
    Else return "Not Short Weierstrass elliptic curve"
end if
if p is prime **then**
    continue
    Else return "Weak elliptic curve"
end if
if discriminant $< -2^{100}$ **then**
    continue
    Else return "Weak elliptic curve"
end if
if base point order is prime **then**
    continue
    Else return "Weak elliptic curve"
end if
if GCD (Curve order, base point order)=1 **then**
    continue
    Else return "Weak elliptic curve"
end if
if base point is on curve **then**
    continue
    Else return "Incorrect base point"
end if
if co-factor is 1 or 2 or 4 **then**
    continue
    Else return "Weak elliptic curve"
end if
if $p + 1 - t$ is a multiple of base point order n **then**
    continue
    Else return "Weak elliptic curve"
end if
if embedding degree of curve $\geq \frac{(N-1)}{100}$ **then**
    continue
    Else return "Weak elliptic curve"
end if

continued to next page..

Algorithm 4 Verification of the proposed elliptic.. (continued from previous page)

if elliptic curve is MOV safe **then**
    continue
    Else return "Weak elliptic curve"
end if
if base point order of twist ! $= p$ **then**
    continue
    Else return "Weak elliptic curve"
end if
if twist equation is elliptic **then**
    continue
    Else return "Weak elliptic curve"
end if
if twist shape is Short Weierstrass **then**
    continue
    Else return "Weak elliptic curve"
end if
if co-factor of twist is 1 or 2 or 4 **then**
    continue
    Else return "Weak elliptic curve"
end if
if GCD (base point order of twist, p) = 1 **then**
    continue
    Else return "Weak elliptic curve"
end if
if Pollard's rho value of elliptic curve $\geq 2^{100}$ **then**
    continue
    Else return "Weak elliptic curve"
end if
if rigidity is True **then**
    continue
    Else return "Weak elliptic curve"
end if
if twist rho value $\geq 2^{100}$ **then**
    continue
    Else return "Weak elliptic curve"
end if
if Joint Rho $\geq 2^{100}$ **then**
    continue
    Else return "Weak elliptic curve"
end if
Otherwise, **return** "Cryptographically safe elliptic curve"

Both the KG256r1 and KG384r1 elliptic curves qualified all the ECDLP and ECC security verifications executed in Algorithm 4. The field orders p and curve orders N of both the elliptic curves were verified deterministically for being a prime number using Pocklington's theorem. Any special structure of prime or pre-studied value is avoided in order to prevent from any vulnerability. For example, NIST P-224 prime i.e., $p = 2^{224} + 2^{96} + 1$ was used by BADA55-VPR-224 and standard ANSSI prime 0xF1FD178C0B3AD58F10126DE8CE42435B3961ADBCABC8CA6DE8FCF3 53D86E9C03 was used by BADA55-R-256 curve respectively to demonstrate vulnerable curves to the community [**51**]. Moreover, the discriminants, embedding degrees, cofactor values and Pollard's rho values of both the proposed curves and their respective twist curves are verified successfully possessing more than their expected threshold values. These curves are also verified to confirm that they are not a case of anomalous and supersingular ones as discussed in Section 4.7.2 and thus, they are suitable for cryptography. Table 4.5 and Table 4.6 shows these values possessed by both the KG256r1 and KG384r1 elliptic curves.

<This space is intentionally left blank.>

Table 4.5: Verification results of the ECDLP security of the proposed elliptic curves

Elliptic curve $\mathbb{E}$	**Offered security level in bits**	**Rho complexity (ρ-value)**	**Embedding Degree (k)**	**Trace (t)**	**Discriminant (D)**	**Curve order (N)**	**Co-factor (h)**	**Non-anom-alous?**	**Non-super-singular?**
KG256r1	128	127.8	105659876450476807 015340827963890761 976544313325663770 7623992353947441213 5987	4357356606872546 7263639581306996 4172673	-2327739398073488908 5043658753644854204 3 7302456081553631250 3 5103075438982165243	1056598764504768070 1534082796389076197 6544313325663770762 399235394744121359871 ($N > 2^{256}$)	1	Yes	Yes
KG384r1	192	191.6	3085049365668014934 0079966421756113888 7972017059009663818 4143875468390039007 7617323565554872996 07397910376591752273	-115066779509797 8901029351544819 8604726047115392 60496758679	-122077938252044953 0033023314772621110 4554029829927838928 9312797446442903024 6303129345660706643 5943911501375652123 1163	3085049365668014 93 4007996642175611 38 887972017059009663 818414387546839003 900776173235655548 729960739791037659 17522731 ($N > 2^{384}$)	1	Yes	Yes

Table 4.6: Verification results of the ECC Security of the proposed elliptic curves

Twist of Elliptic curve $\mathbb{E}'$	**Offered security level in bits**	**Rho complexity (ρ'-value)**	**Embedding Degree (k')**	**Curve order (N')**	**Co-factor (h')**	**Non-anom alous?**	**Non-super-singular?**
KG256r1	128	127.8	4402494852103200292305867831828781749058991026959928337819667792536835404384	1056598764504768070153408279638907619774157846470382801076720270208840497052 17 ($N' > 2^{256}$)	1	Yes	Yes
KG384r1	192	191.6	3085049365668014934007996642175611388879720170590096638183913741909370443227555586204759151520508645560252449240053731591520508645560252449240053 72 ($N' > 2^{384}$)	3085049365668014934007996642175611388879720170590096638183913741909370443227555862047591515205086455602524492400537315915205086455602524492400537 2 ($N' > 2^{384}$)	1	Yes	Yes

4.9.2 Analysis of Trusted Security of KG256r1 and KG384r1 Elliptic Curves

Validation of Trusted Security criteria: $T1$

In this thesis, the /dev/random PRNG is trusted and used for curve generation procedure due to the fact that it has faced a lot of successful cryptanalysis [**62**, **63**, **77**] and sustained long with the Linux kernel since 1994 [**63**]. Moreover, the latest versions (version 4.8 or later) of /dev/random have overcome [**78**] the criticism of having possible entropy attacks [**51**]. Also, Linux Fedora kernel version 4.13.9 is used and /dev/random is selected as the PRNG (sometimes /dev/random is referred as true random number generator (TRNG) because it has the direct interface with the HRNG). The actual point is made here that choose your trusted RNG and own the risk associated with your selection.

Validation of Trusted Security criteria: $T2$

To validate the $T2$ criterion, no pre-studied values of the curve coefficients a and b are used as they have been chosen randomly and independently. The prime numbers p in both the proposed curves KG256r1 and KG384r1 are selected randomly first and then chosen with a form of $p \equiv 3 \ mod \ 4$ for performance tuning and there is no evidence of these primes p and coefficients a and b reported in past as the pre-studied ones.

Validation of Trusted Security criteria: $T3$

To validate the $T3$ criterion, we conducted an experiment by taking three trials of executing Algorithm 3 under the same operational environment with same method and apparatus to retrieve three independent elliptic curves of the same field lengths. Subsequently, it is examined if they exhibit nearly the same cryptographic strength

measured in terms of Pollard's rho value for the curves and their respective twists as discussed in Section 4.7.2. Table 4.7 shows the results obtained from this experiment which proves the successful validation of $T3$ criterion by the proposed KG256r1 and KG384r1 elliptic curves.

Table 4.7: Validation of Trusted Security criteria: $T3$

Trial#	**Elliptic curve parameters** $\mathbb{E} : p, a, b$	**Pollard's rho value / Twist rho value**
1	p**:** 870522537066223168006622796313443027136128167421185164457151061638256241869874 a**:** 174615136804881102021896800654674333559821873138099843085301836053906545031466 b**:** 474236453447930708769624430407166643517516693153699581108106722640661632294077 $G_{x,y}$**:** (345624448642634477922898816667823681998089127518316633864441350836419706701034497371709820032463278128673540807706788485141690500194089547672748025843642377	Rho: $2^{127.6}$ Twist Rho: $2^{127.6}$
2	p**:** 838579318862855558184720589505228271952472116393799709521951765665380521489597 a**:**152220314103590540280417930887083748851745810070536720264160697004225001719957 b**:** 757236637128308681589266033304884863127887549151635841163806300108729839314917 $G_{x,y}$**:** (799911456132998508616609226018730465043144210394223103302316207099394952175754048930300595054686355764380599730714484651315014966555 673263252180995491420)	Rho: $2^{127.6}$ Twist Rho: $2^{127.6}$

Continued to next page..

Table 4.7 – continued from previous page

Trial#	Elliptic curve parameters $\mathbb{E} : p, a, b$	Pollard's rho value / Twist rho value
3	p: 1154551736836473367666951985553866160621859574000747009024653986507696171533 83 a: 8924708959453186116722190782467936189647778182777134965463987376079989422 1702 b: 4745608083843859802072220311634358245557960199332409461120771328874426481 9618 $G_{x,y}$: (8738097286190894292660189281220971403853448243215650202717872822185554003 0831, 1090102247036102758077769996625873990104156057568922076505407835493320691476 87)	Rho: $2^{127.8}$ Twist Rho: $2^{127.8}$

4.10 Results and Discussion

The proposed elliptic curves KG256r1 and KG384r1 are compared with other similar standard Short Weierstrass elliptic curves like NIST, SEC2, Brainpool, FRP256v1 and NUMS curves from ECDLP security, ECC security and trusted security perspectives in this section.

4.10.1 Comparison of the Proposed KG256r1 and KG384r1 Elliptic Curves with Standard Elliptic Curves from ECDLP and ECC Security Perspectives

It is imperative to note from Table 4.8 that none of the standard elliptic curves have passed all the SafeCurves verification criteria [**36**] of ECDLP security and ECC security. However, Brainpool recommended elliptic curves have deviated in their own stipulated procedure of generation [**51**] and hence can not be trusted easily.

Also, their verifiably random generation method is under question as such thing can be intentionally implanted to manipulate the standard as demonstrated by Bernstein et. al. through BADA55 curves [**51**].

Table 4.8: Comparison of ECDLP Security and ECC Security of the proposed elliptic curves

Verification criteria	**Details**	**Supported by elliptic curve**
safeField	Prime of the forms 1 *mod* 4 and 3 *mod* 4	A, B, C, D1, KG256r1, KG384r1
safeEquation	Elliptic curve over prime field possessing either Short Weierstrass or Montgomery or Edward equation	A, B, C, D1, KG256r1, KG384r1
safeBase	Possessing prime order of base point	A, B, C, D1, KG256r1, KG384r1
safeRho	Rho value must be $\geq 2^{100}$	A, B, C, D1, KG256r1, KG384r1
safeTransfer	Resistant to Smart-ASS attack (additive transfer) and MOV attack (multiplicative transfer)	A, B, C, D1, KG256r1, KG384r1
safeDiscriminant	Absolute value of complex-multiplication field discriminant $\|D\| > 2^{100}$	A, B, D1, KG256r1, KG384r1
safeRigid	Allows only fully rigid and somewhat rigid curves	B, C, KG256r1, KG384r1
safeTwist	Above security requirements for twist of the curve as well	C, KG256r1, KG384r1
safeCurve	Elliptic curve is safe if all the above criteria are met	KG256r1, KG384r1

Note: A = NIST recommended elliptic curves, B = Brainpool recommended elliptic curves, C = SEC2 elliptic curves, D1 = ANSSI recommended elliptic curve FRP256v1

4.10.2 Comparison of Cryptographic Security of the Proposed KG256r1 and KG384r1 with Standard Elliptic Curves

The proposed elliptic curves KG256r1 and KG384r1 are compared with standard Short Weierstrass elliptic curves from overall security of ECDLP, ECC and trust perspectives in Table 4.9 and shown in Figure 4.10.2.

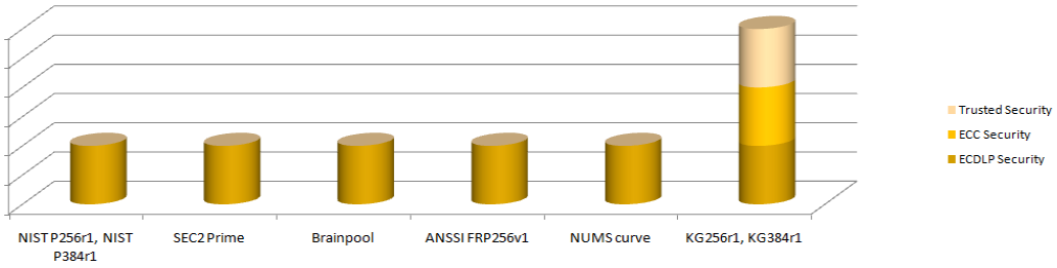


Figure 4.2: Bar chart for comparative security evaluation of the proposed elliptic curves with standard elliptic curves

It is observed from Table 4.9 that only the proposed KG256r1 and KG384r1 elliptic curves are secure from ECDLP, ECC and trust perspectives whereas standard elliptic curves have met the ECDLP security validations only.

4.10.3 Performance of the Proposed Elliptic Curves

The proposed KG256r1 and KG384r1 elliptic curves are demonstrated with cryptographic operations such as key pair generation, signing and verification on desktop machine having x86_64 with Intel(R) Core(TM) i5-10400 CPU with 2.90GHz processor, 16GB DDR4 memory using GNU/Linux version 5.4.0-58-generic and Python Version 3.8.5 software library. Table 4.10 shows the performance metrics of the proposed elliptic curves in various cryptographic implementations. The values indicated are the average outcomes of 10000 trials.

Table 4.9: Comparative security evaluation of the proposed elliptic curves with standard elliptic curves

Elliptic curve	**ECDLP Security**	**ECC Security**	**Trusted Security (T1, T2, T3)**	**Remarks**
NIST P224r1	Yes	No	No	No RNG description. Pre-studied value of coefficient a and special structure of prime p in Mersenne form. Weak twist security [**32**]
NIST P256r1	Yes	No	No	No RNG description. Pre-studied value of coefficient a and special structure of prime p in Mersenne form. Weak twist security [**32**]
NIST P384r1	Yes	No	No	No RNG description. Pre-studied value of coefficient a and special structure of prime p in Mersenne form. Weak twist security [**32**]
SEC2 prime	Yes	No	No	Special structure of prime p (Mersenne prime) and insufficient documentation [**33**]
Brainpool	Yes	No	No	None of the Brainpool curves are generated by their own stipulated procedure [**51**]
ANSSI FRP256v1	Yes	No	No	Pre-studied value of coefficient a and insufficient documentation [**51**]
NUMS curve	Yes	No	No	Deterministic approach with pre-studied coefficients and prime [**50**]
KG256r1	Yes	Yes	Yes	Randomly generated curve parameters with no pre-studied value. User trusted RNG to minimize the risk of manipulation.
KG384r1	Yes	Yes	Yes	Randomly generated curve parameters with no pre-studied value. User trusted RNG to minimize the risk of manipulation.

Table 4.10: Performance of the proposed elliptic curves in cryptographic implementations

Proposed elliptic curve	**Key pair generation**		**Signing**		**Verification**	
	Time elapsed (in seconds)	Number of CPU clock cycles used	Time elapsed (in seconds)	Number of CPU clock cycles used	Time elapsed (in seconds)	Number of CPU clock cycles used
KG256r1	0.021468	62260026	0.0215207	62410198	0.0426380	123650476
KG384r1	0.035866	104012382	0.035838	103931139	0.106852	309871025

4.11 Summary

Three new trusted security acceptance criteria $T1$ - $T3$ are proposed in this chapter to compute cryptographically safe elliptic curves over the prime fields. These trusted security acceptance criteria or simply, the trusted security criteria are invoked along with the ECDLP security and ECC security in order to minimize the scope of manipulation in the curve parameters due to some (intentionally) non-disclosed property or methods exhibited by their proposers and sabotaged standards. It is shown that only randomly drawn curve parameters possess the essential trust factor where a user trusted strong RNG plays a crucial role. The choice of selection of RNG is left with the users who will own the risks associated with their chosen RNG to generate the seed and randomness for curve parameters generation requirements. Two new elliptic curves called KG256r1 and KG384r1 are also introduced after validating the newly proposed trusted security acceptance criteria along with the ECDLP and ECC security validations. Furthermore, it is experimentally proved that if elliptic curves are generated keeping these three security notions into consideration then they would have nearly the same cryptographic strength in terms of Pollard's rho complexity and trustworthiness or suitability. Hence, it is inferred that one must verify trusted security acceptance criteria for randomly generated elliptic curves in addition to ECDLP and ECC security validations for secure implementation of elliptic curve based cryptosystems.

The proposed argument of trusted security and demonstrated KG256r1 and KG384r1 elliptic curves gives the feasibility of future standardization of such randomly generated elliptic curves for trusted cryptographic implementations.

The next chapter discusses the computational resource estimates to compute elliptic curves over large prime fields to solve the Problems 4 - 6 as mentioned in Chapter 3.

Chapter 5

The Proposed Computational Resource Estimation of Short Weierstrass Elliptic Curves

"Science is telling us that we can do phenomenal things if we put our minds and our resources to it."

- Anthony Fauci

This chapter presents statistical estimates of computational resources required for elliptic curves randomly over prime fields of large sizes. This chapter solves the Problems 4[1], Problem 5[2] and Problem 6[3] as mentioned in Chapter 3.

[1]To provide an estimate of computational resources in terms of computing processor i.e., number of the CPU clock cycles to compute cryptographically safe elliptic curve randomly over desired prime field size for cryptographic purposes.

[2]To provide an estimate of computational resources in terms of number of attempts or searches to be made in the security parameter space of the elliptic curve to compute cryptographically safe elliptic curve randomly over desired prime field size for cryptographic purposes within stipulated time.

[3]To estimate computational investment for cryptographically secure elliptic curves over very large prime fields in order to verify feasibility and to prepare existing ECC-based cryptosystem to be kept resilient to quantum attacks using available number of qubits.

5.1 Publications from this chapter

The thesis contributes the following journal paper from this chapter.

1. **Kunal Abhishek** and E. George Dharma Prakash Raj, *Computational Investment in Generation of Elliptic Curves Randomly over Large Prime Fields*, Concurrency and Computation Practice and Experience (2022). (Status: Under Revision)

5.2 Introduction

Elliptic curves are studied extensively in literature for their applications in cryptography. The security of elliptic curve-based system depends upon the number of discrete points exhibited by the elliptic curve group [**79**]. Therefore, the size of the field i.e., prime p needs to be of maximum bit length to offer large number of discrete points in the elliptic curve group. The number of discrete points exhibited by elliptic curve group is known as order or cardinality (which is the set of all discrete points) of the elliptic curve. Here, a prime order or prime cardinality elliptic curve means an elliptic curve $\mathbb{E}$ over a finite field $\mathbb{F}_p$ with $|\mathbb{E}(\mathbb{F}_p)|=$ a prime. But computing elliptic curve over a large prime field in reasonable time requires huge computational resources in terms of the processor and the number of searches. The reason is that, apart from other cryptographic validations of elliptic curve parameters, the searches are made extensively to find a prime order elliptic curve and the order of elliptic curves are validated through factorization method [**16**] which is a time and resource intensive task. Hence, challenges lie with organizations in planning computational resources needed for computing elliptic curves over desired large prime fields within stipulated time. Here, the estimates of the number of CPU clock cycles helps in determining processor requirements whereas the number of attempts or searches helps to decide the number of CPU

cores for speeding up the curve generation process. There is no such research being carried out in literature to estimate the computational resources needed for computation of elliptic curves over given large prime fields. Interestingly, Koblitz [**12**] estimated the probability to draw a suitable random elliptic curve in characteristic 2 in terms of the number of searches for computing a near prime order elliptic curve using Schoof's algorithm. In light of this, one of the goals of this thesis is to derive new computational resource estimates for obtaining suitable prime order elliptic curves randomly over large prime fields.

In particular, the contributions of this chapter are as follows:

- The statistical estimates of computational resources needed to randomly obtain cryptographically suitable prime order elliptic curve over a given prime field size are proposed. These estimates can be predicted from two novel regression equations derived in this chapter. The first regression equation estimates computational resource in terms of the number of CPU clock cycles whereas the second regression equation estimates the number of attempts or searches to be made in the security parameter space of elliptic curve.

- This work is motivated from Koblitz's estimate [**12**] to determine the number of searches needed probabilistically to randomly search a near prime order elliptic curve over $\mathbb{F}_{2^n}$ where n is the bit length of the binary field. In this chapter, a statistical derivation of such estimate is proposed over $\mathbb{F}_p$ where p is a large prime. Additionally, the processor estimate in terms of the number of CPU clock cycles required to randomly obtain a prime order elliptic curve is also proposed in this thesis.

- This chapter present computational resource estimates of computing cryptographically suitable elliptic curves randomly over prime fields of sizes

384, 512, 521 and 1024 bits. A table of computing resource estimates with respect to the elliptic curve prime field sizes is also presented which is used for deciding the prime field size over which an elliptic curve will be resilient to quantum attacks with certain number of qubits as estimated by Roetteler *et. al.* [**39**, **40**].

5.3 The Proposed Approach

The thesis adopts statistical estimation approach to determine computational resource estimates in terms of the number of CPU clock cycles i.e. processor requirements as well as the number of attempts to be made in the security parameter space to randomly generate elliptic curves over large prime fields. The number of searches is crucial for time management in order to obtain cryptographically suitable elliptic curve. Once the number of searches is known, the number of CPU processor cores can easily be decided across which curve searching routine can independently and parallelly run using early-abort strategy. This multiprocessing approach gives controlled speed up in the curve searching process. Hence the number of searches is considered as another important computational resource for estimation purposes in this chapter. A standard procedure as described in Algorithm 5 is followed to generate cryptographically suitable elliptic curves randomly using a trusted RNG as discussed in Section 4.9.2. Further, this section includes the proposed statistical approach for estimation of computational resources for computing elliptic curves randomly over the prime fields.

<This space is intentionally left blank.>

5.3.1 Generation of cryptographically safe elliptic curve over prime field

In this chapter, a novel statistical estimate of computational resources is proposed for computing cryptographically safe elliptic curve randomly over a given prime field size using a standard procedure. This standard procedure taken from various sources [**1**, **2**, **3**, **4**, **9**, **17**, **18**, **19**, **28**, **36**] as mentioned in Algorithm 5 which is also detailed in Algorithm 3 in Chapter 4 with trusted security validations. Algorithm 5 is followed to randomly generate cryptographically safe elliptic curve over desired prime field size. Algorithm 5 follows with fixing a prime p and varying the field elements a and b of Weierstrass equation of the elliptic curve along with varying the order N till a suitable elliptic curve $\mathbb{E}$ is found with N as a prime along with other cryptographic validations. This elliptic curve generation procedure is considered as the standard procedure because every parameter related to the elliptic curve like p, a, b and $G_{x,y}$ are randomly generated and they do not have any special structure or pre-studied values. This standard procedure is represented by Ψ.

In the first step at line 1, the number of CPU clock cycle (CC) which is initially set to 0, starts recording the number of CPU clock cycles throughout the execution of Algorithm 5. The number of CPU clock cycles is determined by the relationship

$$CC = CT \times CF \tag{5.1}$$

where CT is the CPU time and CF is the CPU clock frequency [**80**]. In step 2 and step 3 of Algorithm 5, a prime p is selected randomly of desired length in bits. The prime p is then transformed into field k at line 4 of Algorithm 5. By selecting elliptic curve $\mathbb{E}$ randomly over $\mathbb{F}_p$, we mean that coefficients of $\mathbb{E}(a,b)$ are chosen randomly in the field $\mathbb{F}_p$ [**16**] and are given at lines 5 and 6 in Algorithm 5. An elliptic curve $\mathbb{E}(a,b)$ over field k is generated at line 7.

Algorithm 5 Standard Procedure Ψ: Generation of cryptographically safe random elliptic curve over a given prime field size

Require: Size of prime Field p in bits, number of searches made as *searchCount* set to zero, number of CPU clock cycles as CC set to zero

Ensure: A cryptographically safe randomly generated elliptic curve $\mathbb{E}$ over prime field p, base point, curve order, number of searches and number of CPU clock cycles

1: Start and record CC
2: Select prime field size in bit
3: Select randomly a prime p on selected size
4: Fix $k = GF(p)$ ▷ Generate the field
5: Select randomly coefficient a ▷ Field element
6: Select randomly coefficient b ▷ Field element
7: Generate elliptic curve $\mathbb{E}(k)$ ▷ Elliptic curve over F_p
8: **if** $4a^3 + 27b^2 \neq 0$ ▷ Non-singularity check
9: **else** go to step 5
10: **end if**
11: Compute order N of $\mathbb{E}$
12: $searchCount^{++}$ ▷ Record number of searches made in the cardinality space of the elliptic curve
13: **if** N is prime ▷ Prime cardinality only
14: **else** go to step 5
15: **end if**
16: **if** $\mathbb{E}$ is non-supersingular
17: **else** go to step 5
18: **end if**
19: **if** $N \neq p$ ▷ Non-anomalous check
20: **else** go to step 5
21: **end if**
22: Select randomly a base point $G_{x,y}$ on $\mathbb{E}$
23: Compute base point order n
24: **if** $n \neq N$ ▷ Cofactor check
25: **else** go to step 18
26: **end if**
27: Compute Twist $\mathbb{E}'$
28: **if** $\mathbb{E}'$ is non-singular
29: Compute Cardinality N' of $\mathbb{E}'$
30: **else** go to step 5
31: **end if**
32: **if** N' is prime
33: **else** go to step 5
34: **end if**
35: **if** $\mathbb{E}'$ is non-supersingular
36: **else** go to step 5
37: **end if**
38: Stop recording CC ▷ As desired elliptic curve is computed by now, stop counting the number of CPU clock cycles
39: **return** $\mathbb{E}, G, N, searchCount, CC$

Algorithm 5 checks at line 8 if the elliptic curve $\mathbb{E}$ has non-zero discriminant i.e. $4a^3 + 27b^2 \neq 0$ to ensure that cubic curve is indeed an elliptic curve [**11**, **19**, **75**, **76**]. We need prime order elliptic curves to resist Pohlig-Hellman and Pollard's rho attacks [**2**]. Finding the number of discrete points i.e., the order of the elliptic curve over large prime field for strongest possible system requires a lot of effort as far as computational investment i.e. computational resources are concerned [**16**]. Algorithm 5 runs SEA algorithm at line 11 to determine the order or cardinality of elliptic curve. This step requires $O((logq)^{4+\epsilon})$ bit operations where ϵ is a positive constant and consumes $O((logq)^2)$ memory as given in Section 1.4.4. The number of searches is then counted in step 12. Line 13 checks if the order N is a prime number for ECDLP security [**1**, **3**] which is the most computationally expensive operation. Algorithm 5 uses Multi Polynomial Quadratic Seive (MPQS) method using Gaussian elimination to verify if the order is prime at line 13. MPQS factorization method under plausible assumptions factorizes a number, say N here, in time $O(exp(c \times (lnNlnlnN)^{\frac{1}{2}}))$ where $c \sim 1$ [**81**]. At line 16, only non-supersingular elliptic curve is accepted to retain fully exponential complexity for ECDLP hardness [**2**, **4**, **16**, **51**]. Menezes *et al.* [**4**] showed that supersingular elliptic curves have Trace of Frobenius equal to zero due to which ECDLP can be reduced to the discrete logarithm problem in a finite field to a problem with sub-exponential complexity. Elliptic curve to be computed should not be of anomalous kind where MOV attack is feasible. The anomalous condition [**2**, **7**, **16**, **43**, **82**] is added at line 19 in Algorithm 5. At line 22, a base point $G_{x,y}$ is chosen randomly on $\mathbb{E}$ [**44**]. By random, we mean here that a random element $x_0 < p \in \mathbb{F}_p$ is chosen and we check if $x_0^3 + ax_0 + b$ is a square root in $\mathbb{F}_p$, otherwise we search for another random x_0 till we get a suitable one [**44**]. The bit-complexity for selecting random base point on $\mathbb{E}$ over $\mathbb{F}_p$ and verifying base point order both are $O(log^4 p)$ [**17**]. The base point order n is determined

using Lagrange's theorem at line 23. The base point order is checked to be same as the order of the curve at line 24 so that cofactor value should be 1 for enhanced cryptographic security of the elliptic curve though it is permissible as 2 or 4 as well. At lines 28, 32, 35, we followed Lange *et. al.* [**36**] to impose all ECDLP security requirements of $\mathbb{E}$ on its twist $\mathbb{E}'$. Now the value of CC is noted at line 38 and finally, desired elliptic curve $\mathbb{E}$, base point G and order N along with the number of CPU clock cycles and the number of searches are returned as output of Algorithm 5 at line 39.

5.3.2 Estimation of computational Resources for Computing Random Elliptic Curves over Prime Fields

The Algorithm 6 is proposed to construct two new linear regression equations which are derived independently to estimate the number of CPU clock cycles (η) and the number of searches or attempts (ω) respectively. A comparison table is also presented as one of the outcomes which helps in deciding prime field size of elliptic curve which will be resilient to quantum attacks using certain number of qubits.

In step 1 of Algorithm 6, a new experiment is conducted to create a large real time training data set (Θ) consisting of the number of CPU clock cycles (η) and the number of searches or attempts (ω) made in successful generation of 2400 elliptic curves over different prime fields sizes, recorded as its elements. Similarly, a test data set ($\ddot{\Theta}$) is also created with the number of CPU clock cycles and the number of attempts made in successful generation of 1170 elliptic curves over 21 different prime fields sizes, recorded as its elements. In step 2 of Algorithm 6, statistical modeling using regression is performed on Θ to infer about the estimates of computational resources η and ω required for computation of elliptic curves over desired prime fields sizes. Here, η and ω are the estimates derived as the average

Algorithm 6 Estimating computational resources to compute random elliptic curve $\mathbb{E}$ over given large prime field $\mathbb{F}_p$

Require: A standard procedure Ψ (Algorithm 5), 40 primes $p_1, p_2, \ldots, p_{40}$ of various bit lengths in the range $[18, 252]$ bits with an interval of 6 bits, 18 primes $p_{41}, p_{42}, \ldots, p_{58}$ of different bit lengths in the range $[14, 218]$ bits with an interval of 12 bits and 3 primes p_{59}, p_{60}, p_{61} of bit lengths $254, 266$ and 278 bits respectively

Ensure: Computational resources, statistical test results, verification result of resource estimates and comparison table

1: [Conduct new experiment]

1.1 Create Training Data Set $\Theta = []_{m \times n}$ where m = number of rows = 40 and n = number of columns = 3

1.2 Create Test Data Set $\ddot{\Theta} = []_{r \times s}$ where r = number of rows = 18 and s = number of columns = 3

1.3 **loop** Input p_i with $i = 1, 2, ..., 40$ where p_1 = 18-bit prime, p_2 = 24-bit prime, ..., p_{40} = 252-bit prime:

1.3.1 **loop** 60 times:

1.3.1.1 Run Ψ with input as p_i

1.3.1.2 Record the number of CPU clock cycles η

1.3.1.3 Record the number of searches ω

1.3.2 **end loop**

1.3.3 Compute average (η), average (ω)

1.3.4 Set Θ = [p_i, average (η), average (ω)]

1.3.5 i^{++}

1.4 **end loop**

1.5 **loop** Input p_i with $i = 41, 42, ..., 58$ where p_{41} = 134-bit prime, p_{42} = 146-bit prime, ..., p_{58} = 218-bit prime:

1.5.1 **loop** 60 times:

1.5.1.1 Run Ψ with input as p_i

1.5.1.2 Record the number of CPU clock cycles η

1.5.1.3 Record the number of searches ω

1.5.2 **end loop**

1.5.3 Compute average (η), average (ω)

1.5.4 Set $\ddot{\Theta}$ = [p_i, average (η), average (ω)]

1.5.5 i^{++}

1.6 **end loop**

1.7 **loop** Input p_i with $i = 59, 60, 61$ where p_{59} = 254-bit prime, p_{60} = 266-bit prime, p_{61} = 278-bit prime:

1.7.1 **loop** 30 times:

1.7.1.1 Run Ψ with input as p_i

1.7.1.2 Record the number of CPU clock cycles η

1.7.1.3 Record the number of searches ω

1.7.2 **end loop**

1.7.3 Compute average (η), average (ω)

1.7.4 Set $\ddot{\Theta}$ = [p_i, average (η), average (ω)]

1.7.5 i^{++}

1.8 **end loop**

continued to next page..

Algorithm 6 Estimating computational resources.. (continued from previous page)

2: [Statistical modeling]

2.1 Regression analysis and derivation of desired regression equations for η and ω with best fit based on Θ

3: [Estimation of desired computational resources from Θ]

3.1 **return** Processor estimate η for a given p

3.2 **return** Estimate of searches ω for a given p

4: [Test the statistical model]

4.1 **return** Statistical test results, the tuple $(R^2, R^2{}_{adjusted}, r, p - value)$

4.2 Verify predicted estimates with the estimates obtained from $\ddot{\Theta}$

4.3 **return** TRUE if predicted estimates from $\Theta \leq$ actual estimates from $\ddot{\Theta}$ or FALSE otherwise

5: [Comparison]

5.1 Create comparison table with p of $\mathbb{E}$ as its first column, qubits required to solve ECDLP as its second column and resource estimates η and ω required to compute $\mathbb{E}$ as its third column

5.2 **return** comparison table

number of CPU clock cycles and the average number of searches respectively from 60 observations for each prime field case. In step 3, computational resources based on Θ is estimated in terms of η and ω for a given prime p, are returned. In step 4, the coefficient of determination R^2, $R^2{}_{adjusted}$, correlation coefficient r and $p - value$ of the test statistics are checked. The R^2 value is calculated by squaring the Pearson correlation coefficient that reveals the percentage of variance explained in each of the two correlated variables by the other variable [**84**]. High $R^2{}_{adjusted}$ value indicates a model with small test error [**85**]. High r value shows the strength of association between the two variables whereas suitable $p - value$ denotes the statistical significance of the test. The predicted resource estimates derived from the models based on Θ are compared with the resource estimates obtained from $\ddot{\Theta}$. It is verified if the upper bound of the predicted computational resource estimates are close to the actual computational resource estimates from $\ddot{\Theta}$ under 99% confidence interval. In step 5, a comparison table is returned with the prime field size p, the number of qubits required to solve ECDLP and the computational resource

estimates η and ω required to compute the elliptic curve over $\mathbb{F}_p$.

As indicated in step 1 of Algorithm 6, an experiment is conducted to create a new data set of elliptic curves for training purposes and another data set for testing purposes followed by regression analysis on training data set to obtain the models for desired estimates of computational resources. The detailed experimentation and regression analysis are explained in the following subsections.

Experimentation

In this section, The assumptions and the controls which are used in the experimentation are discussed. Further, the section discusses the computational environment used in the experimentation as well as the methodology by which a large data set is created for modeling and inferencing purposes in this thesis.

i. Assumptions and Control in Experiment

- Algorithm 5 is fixed for experimentation.
- Resource requirements for operating system routines are not considered.
- Communication overheads are not considered.
- X is a non-stochastic controlled variable in this experiment.

ii. Experimentation Environment

An experimentation set up is organized with a Desktop Server having Intel Xeon E5-2620 v4 at 2.1 GHz clock frequency with 32 processor cores with 2 threads per core and 128 GB DDR4 RAM. The SAGE version 8.1 package for elliptic curve generation program on Linux Fedora kernel version 4.13.9 is used. Python versions 2 and 3.6 compilers are used for generating the data sets for experimentation.

MINITAB version 19 and R software were used for statistical calculations pictorial output representations.

iii. Data Set Creation

Algorithm 5 is used to generate elliptic curves over given prime fields sizes. The computational resource estimates η and ω are recorded during elliptic curve generation process till a suitable elliptic curve of prime order is found.

A new data set of elliptic curves over desired prime field sizes was created for modeling purposes. The size of the data set was carefully selected for accuracy of the proposed models. Since the Central Limit Theorem (CLT) proves that even if the population is non-normally distributed, the sampling distribution of the mean will most likely approximate a nice, normal, bell-shaped distribution as long as sample contains at least 30 cases [**84**]. Therefore, it is reasonable to have at least 10 cases for each of the 30 predictor variables in the model [**84**] which means the data set should hold at least $30 \times 10 = 300$ observations in the sample for inferencing. Hence, keeping data size in view, a new sufficiently large data set consisting of η and ω as its elements is created from $40 \times 60 = 2400$ elliptic curves computed under the experimental environment as mentioned in Section 5.3.2 to satisfy the sample size criteria as given in [**84**]. The elliptic curves are defined over 40 prime fields sizes in the range of $[18, 252]$ prime field bit lengths with an interval of 6 bits. Each data element in the data set is the arithmetic mean of 60 distinct observations over each prime field size (see Algorithm 6). This data set is considered as training data set (called Θ) to construct the regression model with best fit. Similarly, a test data set (called $\ddot{\Theta}$) with η and ω as its elements, is also created from $(18 \times 60) + (3 \times 30) = 1170$ elliptic curves defined over 21 different prime fields sizes in the range of [14, 218] bit lengths for interpolation cases and $254, 266$ and 278 bit lengths for extrapolation cases with an interval of 12 bits

each. 18 data elements (interpolation cases) in $\ddot{\Theta}$ is arithmetic mean of 60 distinct observations for each prime field size whereas 3 data elements (extrapolation cases) in $\ddot{\Theta}$ is arithmetic mean of 30 distinct observations for each prime field size. The test data set is created for comparison with predicted values observed from the regression model based on Θ. The maximum prime field size for Θ is limited to 252-bits and that of $\ddot{\Theta}$ as 218 bits for interpolation cases only. However, 3 extrapolation values i.e. 254, 266 and 278 bits to $\ddot{\Theta}$ are added separately to further verify the accuracy of the derived model based on Θ.

Table 5.1 shows Θ with prime field size as input in the first column (ignoring case# column), total number of CPU clock cycles recorded in second column and number of attempts made by the machine in successful generation of the elliptic curve in the third column with a total of 40 entries. Similarly, Table 5.2 shows $\ddot{\Theta}$ with the same attributes as in Table 5.1 in its three columns (ignoring case# column) but having only 21 entries.

Here, the prime field size is considered as X as the Predictor Variable whereas the number of CPU clock cycles (Y) and the number of attempts (Z) made for elliptic curve computation are considered as Regression Variables for two separate models respectively. The data values in Θ and $\ddot{\Theta}$ were recorded from experiments under controlled environment as discussed in Section 5.3.2.

The number of CPU clock cycles and the number of searches made in the security parameter space of elliptic curve to find a prime order elliptic curve depends on the prime field over which elliptic curve is defined. The asymptotic complexity in bit operations for computing the order of elliptic curve using SEA algorithm is $O((logq)^{4+\epsilon})$ where ϵ is a positive constant [**21**]. Further as stated earlier in Section 5.3.1, MPQS method is used to check the order of the elliptic curve N to be a prime with asymptotic complexity $O(exp(c \times (lnNlnlnN)^{\frac{1}{2}}))$ where $c \sim 1$. Algorithm 5 repeats computing a new elliptic curve randomly until a

Table 5.1: Training Data Set (Θ)

Case#	**Prime Field Size in bits** (X)	**Number of CPU clock Cycles** (Y)	**Number of Searches made** (Z)
1	18	1.7852E+11	712
2	24	1.926E+11	865
3	30	2.52581E+11	1196
4	36	1.05707E+13	1504
5	42	5.50157E+11	1627
6	48	1.06851E+12	1873
7	54	2.45426E+12	2001
8	60	3.98226E+12	3166
9	66	1.69909E+13	2943
10	72	2.54716E+13	3456
11	78	3.38216E+13	4164
12	84	3.9335E+13	3827
13	90	6.41849E+13	5642
14	96	8.00742E+13	5536
15	102	1.07634E+14	6654
16	108	1.46966E+14	7389
17	114	2.44204E+14	10860
18	120	2.07239E+14	7845
19	126	2.76189E+14	8428
20	132	6.13986E+14	13714
21	138	6.26786E+14	12135
22	144	8.27762E+14	14344
23	150	9.05017E+14	14013
24	156	1.0066E+15	13661
25	162	1.4215E+15	15117
26	168	2.07947E+15	19891
27	174	2.08301E+15	17982
28	180	2.24864E+15	17267
29	186	2.40415E+15	16479
30	192	4.36018E+15	25488
31	198	5.6743E+15	26916
32	204	5.34167E+15	23379
33	210	5.89211E+15	23242
34	216	6.15421E+15	22411
35	222	9.71521E+15	28740
36	228	1.07283E+16	29068
37	234	1.15439E+16	28272
38	240	1.40148E+16	31181
39	246	1.94564E+16	39541
40	252	1.84172E+16	34062

Table 5.2: Test Data Set ($\ddot{\Theta}$)

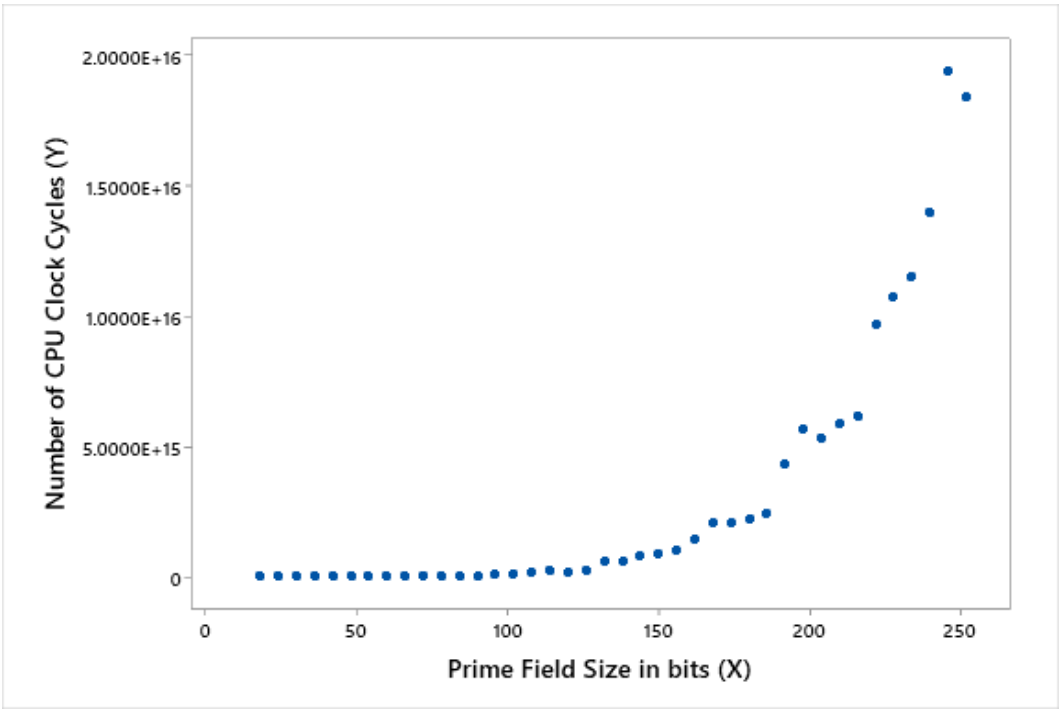
Case#	Prime Field Size in bits (X)	Number of CPU clock Cycles (Y)	Number of Searches made (Z)
1	14	1.56532E+11	538
2	26	2.0379E+11	967
3	38	3.42903E+11	1333
4	50	1.13694E+12	1789
5	62	3.50459E+12	2896
6	74	2.64485E+13	3782
7	86	3.16236E+13	3203
8	98	7.39033E+13	4838
9	110	1.75149E+14	8428
10	122	2.40678E+14	9134
11	134	6.5509E+14	14068
12	146	6.03467E+14	9853
13	158	1.32985E+15	15955
14	170	2.01196E+15	18813
15	182	2.56179E+15	18957
16	194	4.35943E+15	22208
17	206	5.27928E+15	22354
18	218	8.28908E+15	28787
19	254	1.13898E+16	18547
20	266	2.22417E+16	28769
21	278	2.63487E+16	28464

prime order curve is found resulting high cost of CPU clock cycles as observed in Table 5.1 and Table 5.2.

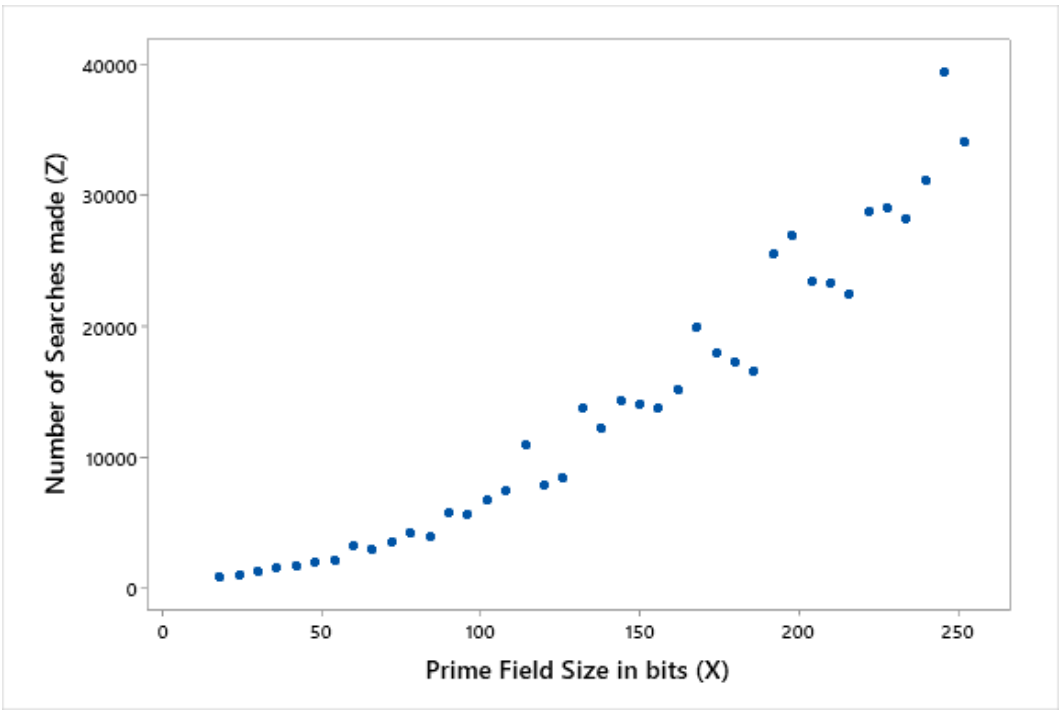
Regression Analysis on Training Data Set

The Scatterplot of Predictor versus Response is the first step for regression analysis [**86**]. Hence, the Scatterplots of X vs. Y and X vs. Z to verify relationship between X and Y as well as between X and Z are plotted to verify if any unusual points or outliers are present in the data set Θ. The desired Scatterplots are presented in Figure 5.1(a) and Figure 5.1(b) respectively.

It is evident from Figure 5.1 that X and Y as well as X and Z have non-linear relationship and no such far outliers are visible and therefore, a polynomial regression model i.e. second order or quadratic model qualifies to be a fit case [**85**].



(a)



(b)

Figure 5.1: Scatterplots for: (a) Prime Field Size in bit Vs. Number of CPU Clock Cycles; (b) Prime Field Size in bits Vs. Number of Searches

i. Model selection with best fit for estimation of the number of CPU clock cycles (η) and number of searches made (ω)

The two regression equations for η and ω respectively are derived with best fits from Θ under the following assumptions:

1. ϵ_i is a random variable and is assumed to have normal distribution, $N \sim (0, \sigma^2)$ and $Cov(\epsilon_i, \epsilon_j) \forall i \neq j (i, j = 1, ..., 40)$. So, our expectation is $E(\epsilon_i) = 0$ and variance $V(\epsilon_i) = \sigma^2$.

2. $E(Y_i) = E(\beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \epsilon_i)$
   $= \beta_0 + \beta_1 X + \beta_2 X_i^2 + 0$
   $= \beta_0 + \beta_1 X + \beta_2 X_i^2$ and,
   $V(Y_i) = E(\beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \epsilon_i)$
   $= V(\epsilon_i) = \sigma^2$.

3. For statistical inferences, we assume $\epsilon_i \overset{iid}{\sim} N(0, \sigma^2)$ and, $Y_i \overset{iid}{\sim} N(\beta_0 + \beta_1 X_i + \beta_2 X_i^2, \sigma^2)$.
   Now $\epsilon_i \overset{iid}{\sim} N(0, \sigma^2)$ leads to consequences like $Y_i \overset{iid}{\sim} N(\beta_0 + \beta_1 X_i + \beta_2 X_i^2, \sigma^2)$.

The polynomial regression (quadratic) models for Figure 5.1 i.e. for estimation of the number of CPU clock cycles and number of searches made can be constructed as

$$Y = \beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \epsilon_i \tag{5.2}$$

where regression coefficient β_0 is known as Intercept, β_1 is called linear effect parameter and β_2 is called quadratic effect parameter. ϵ_i is the error which is normally observed as independent and identically distributed (iid) random variable with $N(0, \sigma^2)$. As desired, $E(Y) = \beta_0 + \beta_1 X_i + \beta_2 X_i^2$ and $Var(Y) = \sigma^2$ in

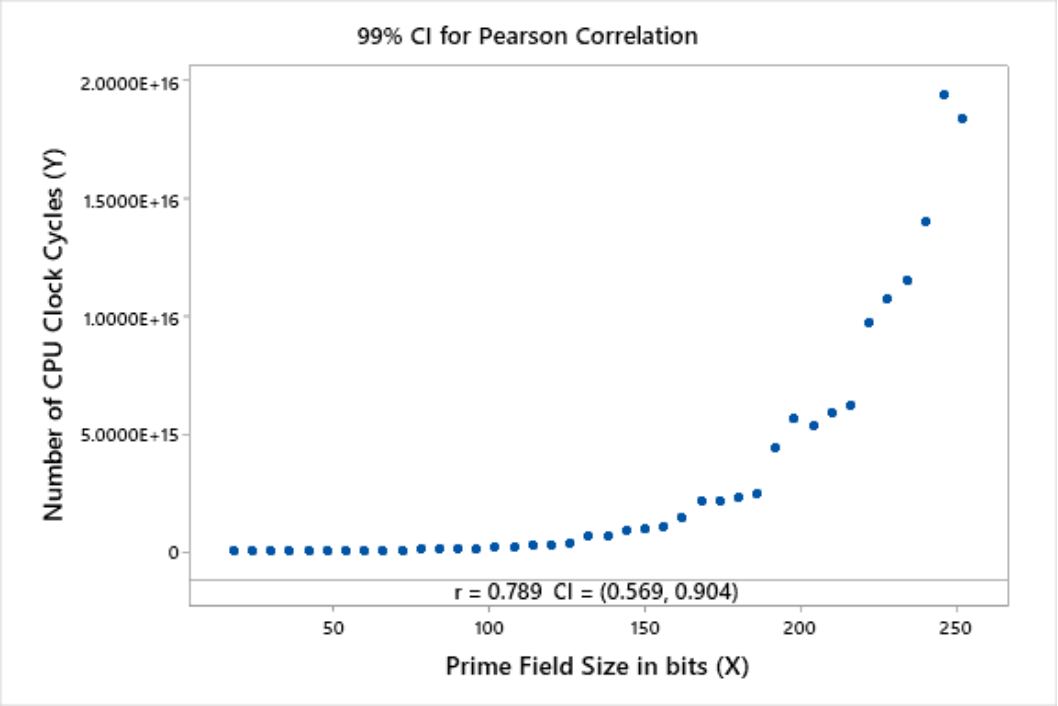
our case. It is also noted that parameters $\beta_0, \beta_1, \beta_2$ and variance σ^2 are unknown whereas ϵ_i is unobserved. Here, the goal is to determine or estimate these unknown parameters while minimizing the error ϵ_i.

Let us first determine how the variables X and Y as well as X and Z are associated with each other by computing Pearson correlation coefficient r which is given by the formula

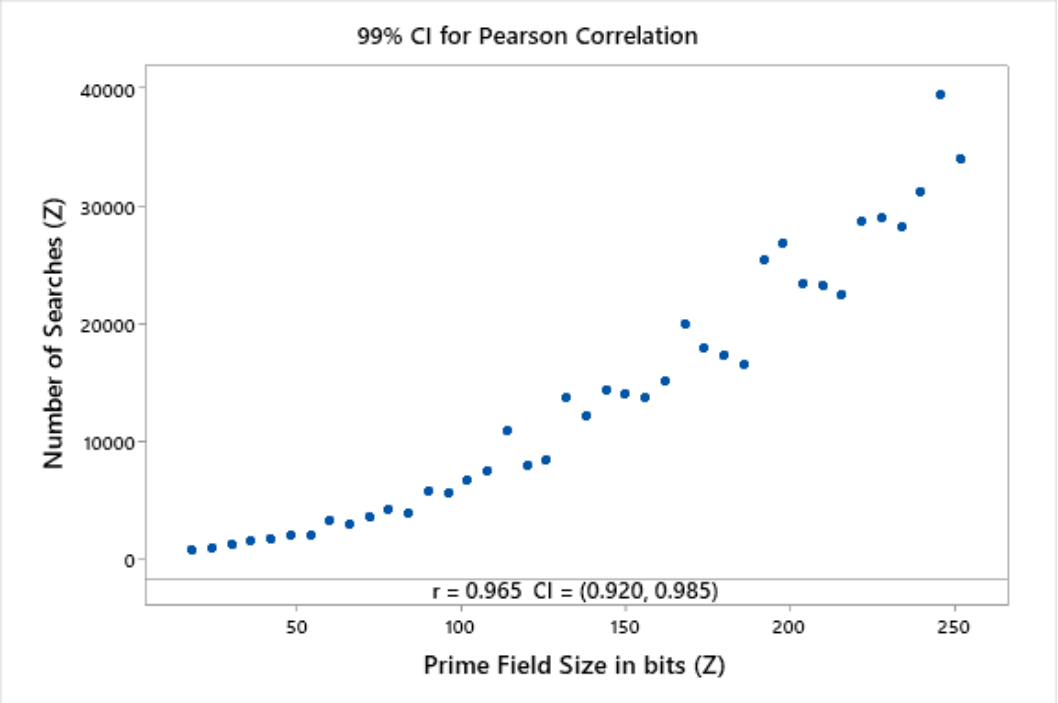
$$r = \frac{\sum (z_x, z_y)}{N} \tag{5.3}$$

where r = Pearson correlation coefficient, $z_x = (X - \mu)/\sigma = z$ score for variable X, $z_y = (Y - \mu)/\sigma = z$ score for variable Y, μ = population mean, σ = standard deviation, N = number of pairs of X and Y scores and observing their Scatterplots of correlation coefficients as shown in Figure 5.2(a) and Figure 5.2(b) respectively.

The value of correlation coefficient r is determined between X and Y as 0.789 from (5.3) in the confidence interval $(0.569, 0.904)$ which shows strong association between both the variables. Similarly, r value between X and Z is determined from (5.3) as 0.965 in the confidence interval $(0.920, 0.985)$ which shows very strong association between both the variables. Further, suitable polynomial regression models for estimating CPU clock cycles and number of searches can be obtained with best fit using Ordinary Least Squares (OLS) method in which parameter estimates are chosen to minimize a quantity called the Residual Sum of Squares (RSS) [**86**]. The quality of the fitted line is assessed by two methods: The "lack of fit" determined by the Residual Standard Deviation/Error (RSD/RSE) and the "measure of fit" determined by the Coefficient of Determination R^2 [**85**]. Furthermore, $R^2_{adjusted}$ is computed to assure that all the correct variables only are included in the model and no noise variables are present [**84**]. R^2 and $R^2_{adjusted}$ statistics together gives adequately the superiority of the improved alternative model. The derivation of polynomial regression (quadratic) model is given in Algorithm 7 which is used to obtain both the models η and ω.



(a)



(b)

Figure 5.2: Correlation Plots for: (a) X Vs. Y; (b) X Vs. Z

Algorithm 7 Finding Polynomial Regression (Quadratic) Model and Test Statistics

Require: (x_i, y_i) where $x_i \in X$ and $y_i \in Y$ where $i = 1, 2, \cdots, n$ from training data set Θ and $n = 40$

Ensure: Regression coefficients β_0, β_1, β_2 and p-value, R^2, $R^2{}_{adjusted}$

1: [Estimation of β_0, β_1 and β_2]

1.1 Compute Residual Sum of Squares(RSS):

$$RSS = \epsilon_1^2 + \epsilon_2^2 + \epsilon_3^2 + ... + \epsilon_n^2$$

where ϵ is the residual or error and n is the number of observations

$$\Rightarrow RSS = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 \times x_i - \beta_2 \times x_i^2)^2$$

1.2 Use Least Square approach to compute β_0, β_1 and β_2 such that RSS is minimum. Take partial derivative of RSS w.r.t. β_0, β_1 and β_2 and equating them to zero:

$$\frac{\partial(RSS)}{\partial\beta_0} = 0 = -2\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 \times x_i - \beta_2 \times {x_i}^2)$$

$$\frac{\partial(RSS)}{\partial\beta_1} = 0 = -2\sum_{i=1}^{n} x_i \times (y_i - \beta_0 - \beta_1 \times x_i - \beta_2 \times {x_i}^2)$$

$$\frac{\partial(RSS)}{\partial\beta_2} = 0 = -2\sum_{i=1}^{n} {x_i}^2 \times (y_i - \beta_0 - \beta_1 \times x_i - \beta_2 \times {x_i}^2)$$

which implies

$$\sum_{i=1}^{n} y_i = n\beta_0 + \beta_1 \sum_{i=1}^{n} x_i + \beta_2 \sum_{i=1}^{n} {x_i}^2 \tag{5.4}$$

$$\sum_{i=1}^{n} x_i y_i = \beta_0 \sum_{i=1}^{n} x_i + \beta_1 \sum_{i=1}^{n} {x_i}^2 + \beta_2 \sum_{i=1}^{n} {x_i}^3 \tag{5.5}$$

$$\sum_{i=1}^{n} {x_i}^2 y_i = \beta_0 \sum_{i=1}^{n} {x_i}^2 + \beta_1 \sum_{i=1}^{n} {x_i}^3 + \beta_2 \sum_{i=1}^{n} {x_i}^4 \tag{5.6}$$

1.3 Compute β_0, β_1 and β_2 using (5.4), (5.5) and (5.6)

continued to next page..

Algorithm 7 Finding Polynomial Regression.. (continued from previous page)

2: [Assessing statistical significance of the test]

2.1 Statistical Significance of the Test

2.1.1 Formulate $H_0 : \beta_1, \beta_2 = 0$ and $H_\alpha : \beta_1, \beta_2 \neq 0$

2.1.2 Compute $p - value$ at $\alpha = 0.01$

3: [Assessing accuracy of the model]

3.1 Compute coefficient of determination R^2

$$R^2 = 1 - \frac{RSS}{TSS}$$

which implies

$$R^2 = 1 - \frac{RSS}{\sum_{i=1}^{n} (y_i - \overline{y})}$$

where $\overline{y} = \frac{1}{n} \times \sum_{i=1}^{n} y_i$

3.2 Compute $R^2{}_{adjusted}$

$$R^2{}_{adjusted} = 1 - \frac{RSS/(n-d-1)}{TSS/(n-1)}$$

where d is the number of variables

3.3 Compute RSD

$$RSD = \sqrt{\frac{1}{n-2} \times RSS} \tag{5.7}$$

4: [Results]

return β_0, β_1, β_2 and $p - value$, R^2, $R^2{}_{adjusted}$

In step 1 of Algorithm 7, the value of RSS is first calculated with a goal to minimize it. Then OLS approach is followed so that RSS will be minimum [**87**].

In step 2, the statistical significance of the test is accessed. The R^2 and $R^2_{adjusted}$ values are obtained in step 3 to analyze the model that fits best with Θ to determine η and ω in computing a suitable random elliptic curve randomly. Thus, accuracy of the model is obtained by observing R^2, $R^2_{adjusted}$ and RSD values in step 3 of Algorithm 7. Thus, accuracy of the model is obtained by observing R^2 and $R^2_{adjusted}$ values in step 3 of Algorithm 7. Finally, the parameters β_0, β_1, β_2, $p - value$, R^2 and $R^2_{adjusted}$ is returned in step 4 as the output of Algorithm 7. The "goodness of fit" is verified for which the models' statistics for η and ω are summarized in Table 5.3 and Table 5.4 respectively. Table 5.3 and Table 5.4 presents the $p - value$ and the degree of freedom which denotes the statistical significance of the models (linear and polynomial quadratic) for η and ω and the approximate number of observations in the data set Θ for determining statistical significance respectively [**84**].

Linear model without the quadratic term has smaller R^2, $R^2_{adjusted}$ and higher RSD values [**85**]. Therefore, the statistical properties of the coefficients' estimates R^2, $R^2_{adjusted}$ and RSD are improved by adding the quadratic term in the regression equations. These values are shown in Table 5.3 and Table 5.4 for η and ω respectively. The Model Statistics for η and ω are then compared based on these statistical properties of the coefficients of the equations. We observed in both the Table 5.3 and Table 5.4 that quadratic model has better R^2, $R^2_{adjusted}$ and minimal RSD values than that of linear model and hence, quadratic model is preferred over the linear model.

a. Model Selelction for η

It is evident from Table 5.3 that polynomial quadratic model is the best fit with higher R^2 value and therefore, it is chosen over the alternative linear model. The resulting fitted line for desired quadratic model is shown in Figure 5.3(a).

The best fitted model for η can be constructed with prime field size (X) and number of CPU clock cycles (Y) as inputs to Algorithm 7. The output of Algorithm 7 is the desired model for η which is represented as

$$Y = 3.99E + 15 - 1.16E + 14 \times X + 6.43E + 11 \times X^2 \tag{5.8}$$

where $\beta_0 = 3.99E + 15$, $\beta_1 = -1.16E + 14$ and $\beta_2 = 6.43E + 11$.

Here, $p - value$, R^2 and $R^2_{adjusted}$ values of the model are computed as < 0.005, 91.71% and 91.26% respectively as shown in Table 5.3.

b. Model Selelction for ω

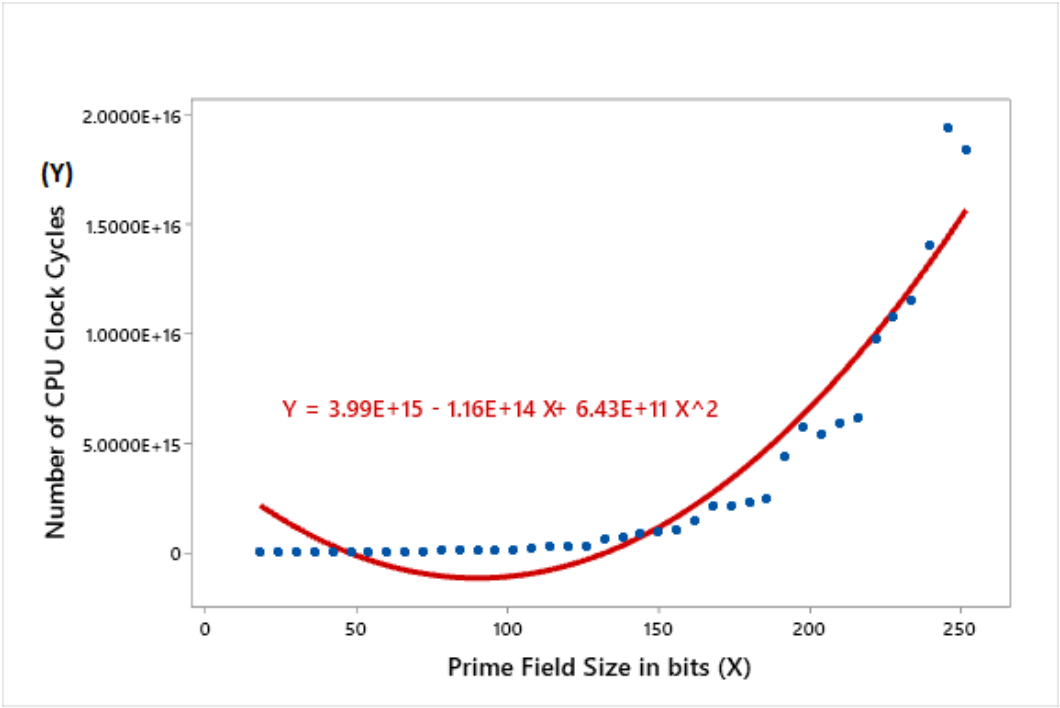
It is evident from Table 5.4 that R^2 values of quadratic model is higher than the corresponding linear model, therefore a quadratic model is selected for estimating ω. The fitted line for the desired quadratic model is shown in Fig. 5.3(b).

The best fitted model for ω can be constructed with prime field size (X) and number of searches or attempts made (Z) as inputs to Algorithm 7. The desired model of ω is the output of Algorithm 7 which is represented as

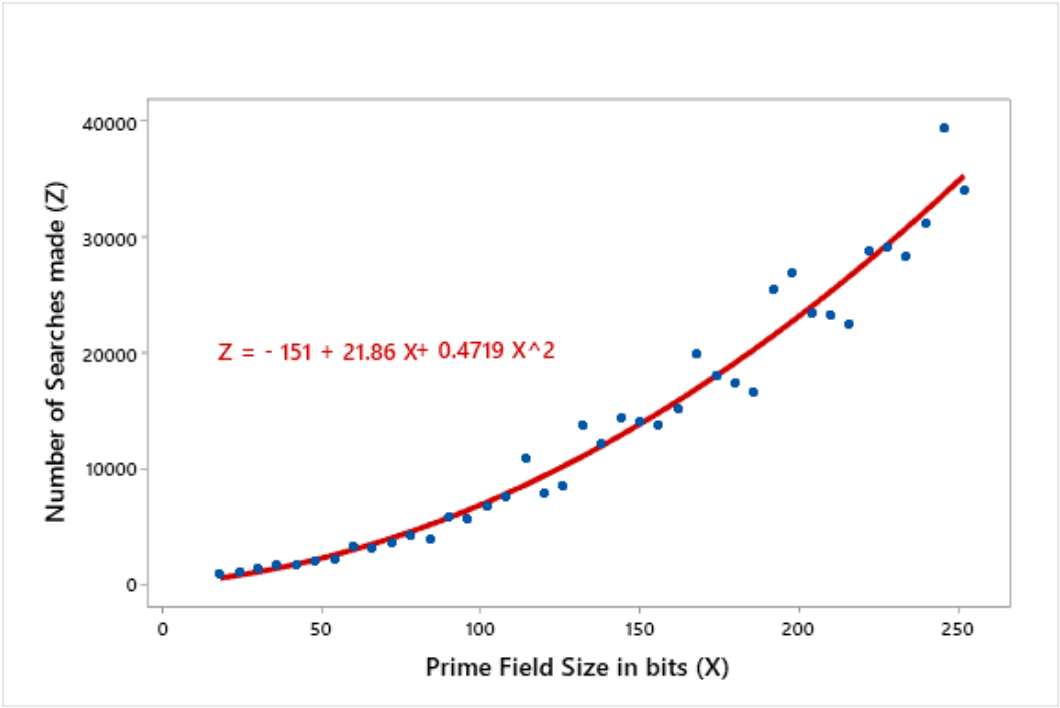
$$Z = -151 + 21.86 \times X + 0.4719 \times X^2 \tag{5.9}$$

where $\beta_0 = -151$, $\beta_1 = 21.86$ and $\beta_2 = 0.4719$.

Here, $p - value$, R^2 and $R^2_{adjusted}$ values of the model are computed as < 0.005, 96.70% and 96.53% respectively as shown in Table 5.4.



(a)



(b)

Figure 5.3: Fitted Line Plots for: (a) Number of CPU Clock Cycles (η); (b) Number of Searches made (ω)

Table 5.3: Comparison of Model Statistics for η

Statistics	**Selected Polynomial Quadratic Model**	**Alternative Linear Model**
R^2	91.71%	62.29%
$R^2_{adjusted}$	91.26%	61.29
$p - value$, model	$< 0.005^*$	$< 0.005^*$
Residual Standard Deviation	1520984304908599.0	3201060156504290.5
Degree of freedom	39	39

*Statistically significant ($p - value < 0.01$)

Table 5.4: Comparison of Model Statistics for ω

Statistics	**Selected Polynomial Quadratic Model**	**Alternative Linear Model**
R^2	96.70%	93.14%
$R^2_{adjusted}$	96.53%	92.96%
$p - value$, model	$< 0.005^*$	$< 0.005^*$
Residual Standard Deviation	2022.578	2879.424
Degree of freedom	39	39

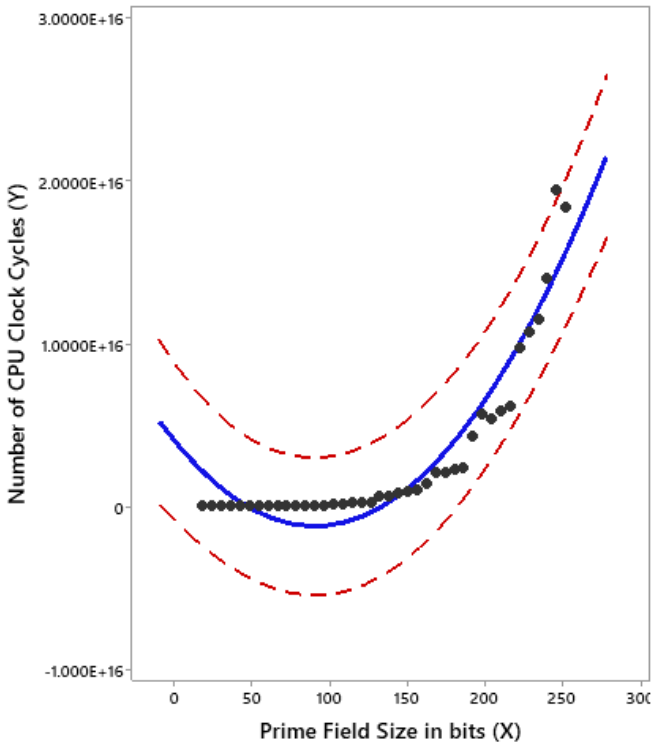
*Statistically significant ($p - value < 0.01$)

5.4 Results and Discussion

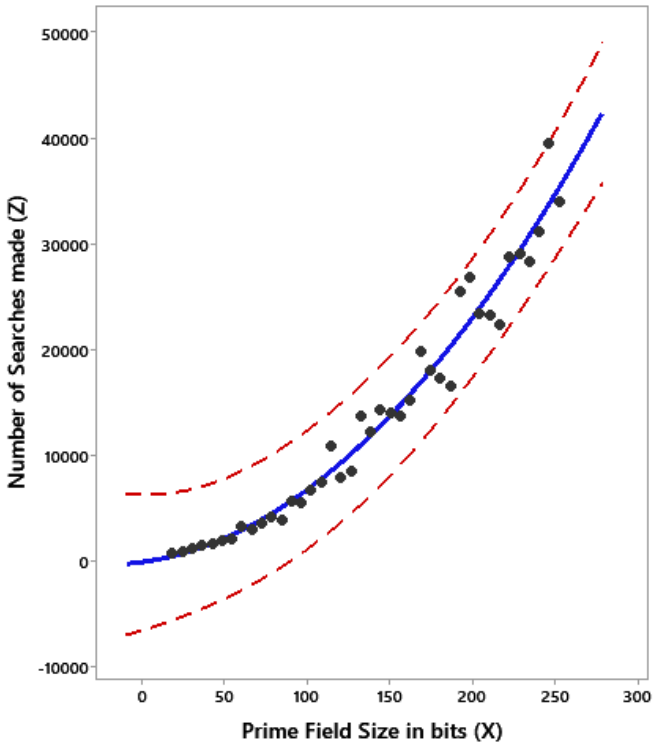
The desired computational resource estimates are derived by equations (5.8) and (5.9) which are novel in light of Koblitz's estimates as discussed in Section 2.7.1. Koblitz gave probabilistic resource estimate of number of searches made over $\mathbb{F}_{2^n}$ where the experimentation was carried out at Hewlett Packards Lab [**16**] to derive a large number of elliptic curves. However, resources in terms of processor were not estimated by the author. In contrast with Koblitz's estimates, proposed resource estimates include processor estimates in terms of the number of CPU clock cycles along with the number of searches made in the security parameter space of the elliptic curve. The proposed resource estimates are based on experiments that were carried out to create a large data set with elliptic curves over $\mathbb{F}_p$ and modeled with regression technique.

Further, the proposed regression models (5.8) and (5.9) are able to precisely predict the computational resources required for random generation of an elliptic curve over a given prime field size. The presented model is also verified with the test data set and found to be within the upper bound of the resource prediction interval. The Prediction Plots for computational resources are shown in Figure 5.4(a) and Figure 5.4(b). The blue fitted line in Figure 5.4(a) shows the predicted value of Y for its corresponding X value whereas the red dashed line shows 99% prediction interval. Similarly, the blue fitted line in Figure 5.4(b) shows the predicted value of Z for its corresponding X value whereas the red dashed line shows 99% prediction interval.

The results of computational resource estimates ($Y_{predicted}$, $Z_{predicted}$) determined by (5.8) and (5.9) with the experimental estimates (Y_{actual}, Z_{actual}) observed from $\ddot{\Theta}$ (as given in Table 5.2) collated with three extrapolated X-values are compared and shown in Table 5.5. It is observed from Table 5.5 that the proposed regression models (5.8) and (5.9) has precisely predicted the number



(a)



(b)

Figure 5.4: Prediction Plots for: (a) Prime Field Size Vs. Number of CPU Clock Cycles (η); (b) Prime Field Size Vs. Number of Searches made (ω)

Table 5.5: Resource prediction for interpolation and extrapolation cases

X	Y_{actual}	$Y_{predicted}$ (99% Prediction Interval Upper Bound) from (5.8)	**Prediction matched?** (Yes/ No)	Z_{actual}	$Z_{predicted}$ (99% Prediction Interval Upper Bound) from (5.9)	**Prediction matched?** (Yes/ No)
Interpolation						
14	1.56532E+11	7.07955E+15	Yes	538	6346.2	Yes
26	2.0379E+11	5.85252E+15	Yes	967	6639.1	Yes
38	3.42903E+11	4.85648E+15	Yes	1333	7128.9	Yes
50	1.13694E+12	4.08310E+15	Yes	1789	7804.5	Yes
62	3.50459E+12	3.52376E+15	Yes	2896	8654.4	Yes
74	2.64485E+13	3.17019E+15	Yes	3782	9667.6	Yes
86	3.16236E+13	3.01496E+15	Yes	3203	10834	Yes
98	7.39033E+13	3.05184E+15	Yes	4838	12146	Yes
110	1.75149E+14	3.27592E+15	Yes	8428	13597	Yes
122	2.40678E+14	3.68367E+15	Yes	9134	15181	Yes
134	6.5509E+14	4.27293E+15	Yes	14068	16897	Yes
146	6.03467E+14	5.04294E+15	Yes	9853	18742	Yes
158	1.32985E+15	5.99423E+15	Yes	15955	20719	Yes
170	2.01196E+15	4.12875E+15	Yes	18813	22828	Yes
182	2.56179E+15	8.44978E+15	Yes	18957	25076	Yes
194	4.35943E+15	9.96199E+15	Yes	22208	27467	Yes
206	5.27928E+15	1.16714E+16	Yes	22354	30010	Yes
218	8.28908E+15	1.35853E+16	Yes	28787	32715	Yes
Extrapolation						
254	1.13898E+16	2.06370E+16	Yes	18547	41910	Yes
266	2.22417E+16	2.34516E+16	Yes	28769	45372	Yes
278	2.63487E+16	2.65094E+16	Yes	28464	49046	Yes

of CPU clock cycles (η) and number of searches made (ω) for various prime fields sizes in both the interpolation and extrapolation cases. A use case of our resource estimates in Table 5.6 is presented that helps to decide suitable prime field size of elliptic curve which will be resilient to quantum attacks using certain number of qubits. For example, Roetteler *et. al.* [**39**, **40**] proposed these qubits estimates required to break ECDLP over prime fields sizes of 192, 224, 256, 384, 512, 521 and 1024 bits as shown in Table 5.6. Now, since a quantum computer with 3500 qubits can solve ECDLP over 384 bit prime field as shown in Table 5.6, therefore, the resilient prime field size of safe elliptic curves can be decided as 512 bit that requires 4636 qubits which is far from attacker's reach. Accordingly, computational resources to generate elliptic curve with 521 bit can be estimated from (5.8) and

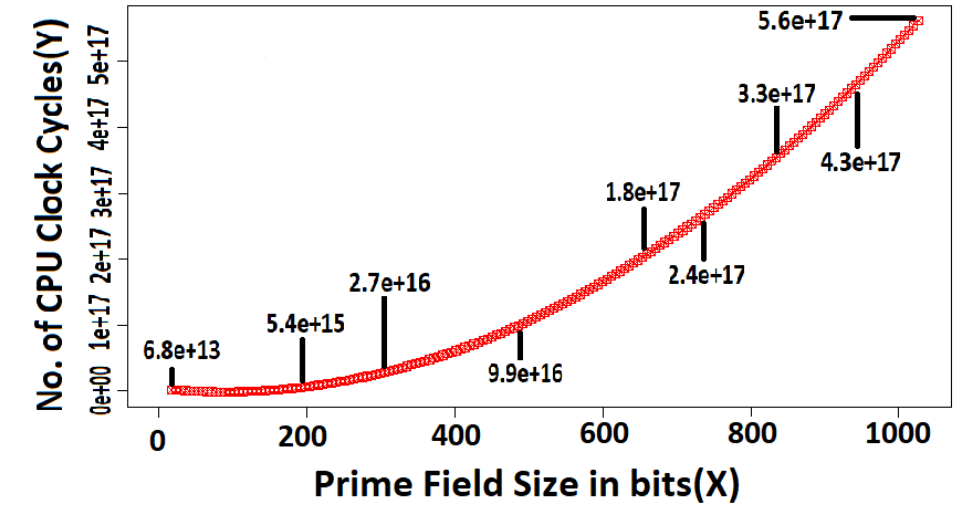
(5.9) as shown in Table 5.6.

Table 5.6: Estimate of computational investment for elliptic curves whose discrete logarithm problem (ECDLP) is intractable against quantum attacks

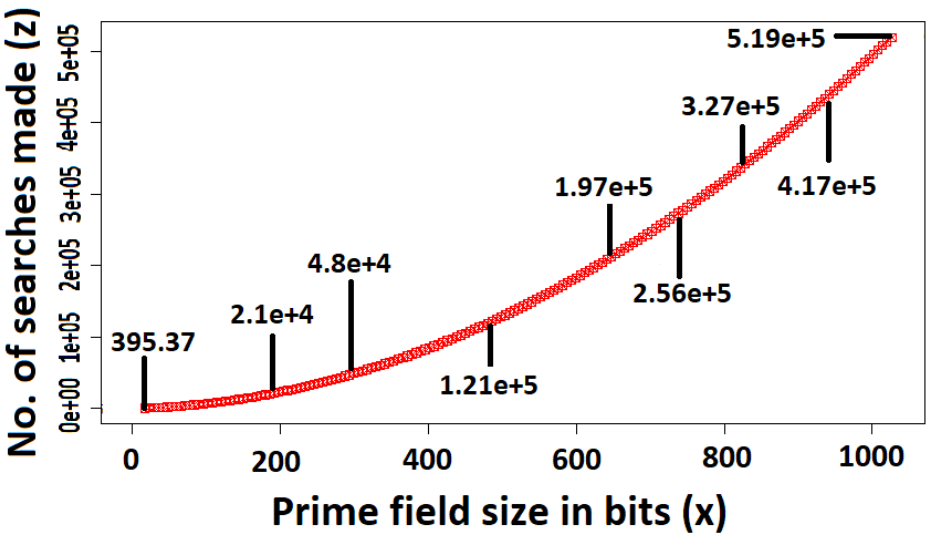
Prime Field Size p **(**n**-bit) of** $\mathbb{E}$	**Qubits to solve ECDLP [39, 40]**	**Proposed Computational Resources under 99% prediction interval (upper bound)**	
		Number of CPU clock cycles	**Number of Searches**
192	1754	9.69644E+15	27058.2
224	2042	1.46215E+16	34131.7
256	2330	2.10894E+16	42472.7
384	3484	6.43080E+16	91060.4
512	4636	1.34915E+17	163520
521	4719	1.40878E+17	169470
1024	9246	6.79685E+17	676720

5.5 Limitation of the Proposed Resource Estimate

It is noted that the predicted computational resources lie on a growing parabolic path as shown in Figure 5.5 which asserts that prediction of both η and ω in extrapolation cases up to 1024 bit prime field size would be precise for consideration. Moreover, as the proposed approach is to randomly choose elliptic curves from its security parameter space, there is an undeniable possibility that the predicted resource estimates in both interpolation and extrapolation cases may not be appropriate. For example, one may get an ideal elliptic curve over a given prime field size within the first or within few attempts itself in the best case scenario causing the predicted estimates appeared to be entirely inappropriate. But such events are extremely unlikely and rare to happen. However, it is always a good idea to consider not less than an average case (keeping in view the large prime field



(a)



(b)

Figure 5.5: Extrapolation Plots for: (a) number of CPU Clock Cycles (η); (b) number of Searches (ω)

size) to decide such computational resources needed for generation of elliptic curve randomly.

5.6 Determination of CPU Processor from Computational Resources Estimates

Let us assume c be the number of CPU cores, P be the CPU processor's capacity in Giga Hertz (GHz), x be the number of CPU clock cycles and t be the time in seconds, then processor's capacity is determined as the rate of CPU clock cycles i.e.,

$$P = \frac{x}{t} \implies P \times c = \frac{x}{t} \implies t = \frac{x}{P \times c} \tag{5.10}$$

Now, from Table 5.6, for 512 bit prime field size, the value of $x = 1.34915E + 17$. As there is a very slow upgradation in the CPU processor capabilities since 2006 which is normally in the range of 3 GHz to 3.6 GHz, let us consider that we have single CPU processor of 3 GHz processing power only. From Equation (5.6), it can be calculated that

$$t = \frac{1.34915E + 17 \times 10^{-9}}{3 \times c} \tag{5.11}$$

Here, Equation (5.6) suggests that number of CPU cores will decide the stipulated time for a successful derivation of an elliptic curve over a prime field size (in this particular case, it is 512 bit) using a single core processor of 3 GHz processing power. Now, to set up a trade-off between the appropriate requirement of CPU cores and time to successfully derive the elliptic curve, it is needed to consider the number of searches or attempts made for the same. It is obvious from Table 5.6 that 512 bit prime field size will require around 163520 searches or attempts in the security parameter of the elliptic curve. Let us assume that 1000 cores machine with 3 GHz processing power with each core is available and they will be assigned with

$\frac{163520}{1000} \approx 164$ searches to each one of them, then, the approximate time requried for successful generation of a suitable elliptic curve will be

$$t = \frac{1.34915E + 17 \times 10^{-9}}{3 \times 1000} \implies t \approx 44971.67\ Seconds \approx 12.5\ Hours \quad (5.12)$$

Here, it should be noted that searching the suitable elliptic curve requires a prime order elliptic curve that are validated through factorization method which is time intensive task. Therefore, each core should be loaded with minimum possible number of searches while implementing more number of CPU processing cores to the machine.

5.7 Summary

Computing new elliptic curves randomly over large prime fields for cryptographic purposes has been one of the biggest challenges observed in strategic interest because of huge computational resource requirements in terms of processor and targeted time line for curve computation. The thesis in this chapter approached such problem statistically using realistic data and practical assumptions and controls in the experiments as well as in analysis. Two large data sets with 2400 and 1170 elliptic curves were created for modeling and testing purposes respectively. Subsequently, computational resources i.e. the number of CPU clock cycles (Y) and number of searches (Z) are proposed from two novel equations $Y = 3.99E + 15 - 1.16E + 14 \times X + 6.43E + 11 \times X^2$ and $Z = -151 + 21.86 \times X + 0.4719 \times X^2$ respectively where X is the prime field size in bits. These equations were derived under 99% confidence interval along with certain statistical assumptions. Convincing statistical results about these equations were reported with the “goodness of fit" measured as R^2 value (called the coefficient of determination) which equals 91.71% and 96.70% in each case respectively, the

correlation coefficient (r) values between X, Y and between X, Z equals 0.789 and 0.965 respectively, $p - value$ for both equations were found to be < 0.005 with 39 degree of freedom for each equation respectively. The predicted resource estimates from the proposed equations were verified with real time test data both in interpolation and some extrapolation cases which confirms accuracy of the proposed regression models. In the light of Koblitz's work [**12**], novel statistical resource estimates for computing the number of searches as well as the number of CPU clock cycles (processor's estimate) to obtain a prime order elliptic curve defined over prime field are proposed. We further presented computational resource estimates of elliptic curves over prime fields sizes of 384, 512, 521 and 1024 bits against the quantum estimates suggested by Roetteler *et. al.* [**39**, **40**] to break the ECDLP. The advantage of the proposed work is that it suggests computational investments to compute new elliptic curves over chosen large prime field size which will be resilient to practical number of qubits under attack and therefore, the existing elliptic curve based cryptosystems may be kept alive. The contributions of this chapter in the thesis will help organizations to decide and allocate appropriate computational resources to randomly compute new elliptic curves over large prime fields within stipulated time. Moreover, the training data set can further be enlarged with more number of elliptic curves over different large prime fields to enhance the accuracy of the proposed models. This work can also be extended to find new computational resources to randomly obtain elliptic curves over the binary fields using appropriate procedure.

The next chapter deals with the construction of a novel kernel CSPRNG which uses cryptographically secure and trusted elliptic curves over large prime field whose requirements of computational resources are discussed in this chapter and whose method of computation is already discussed in Chapter 4 of the thesis.

Part II

Construction of a Novel CSPRNG Using Elliptic Curves For Kernel Applications

Chapter 6

Design and Implementation of The Proposed KCS-PRNG

"Random Numbers should not be generated with a method chosen at random"

- Donald Knuth

In this part of the thesis, a novel CSPRNG for kernel applications is designed and implemented whose generated bitstreams are statistically random looking and unpredictable with non-reproducibility property. This chapter in the second part of the thesis solves the Problem 7[1] as mentioned in Chapter 3.

6.1 Publications from this chapter

The thesis contributes the following publications from this chapter.

[1] A new competitive candidate CSPRNG for kernel or cryptographic usage is highly desirable which could exhibit statistical properties of randomness and unpredictability along with the non-reproducibility property of randomness.

1. **Kunal Abhishek** and E. George Dharma Prakash Raj, *On Random Number Generation for Kernel Applications*, Fundamenta Informaticae, IOS Press (2022). (Status: Accepted - In press)

2. **Kunal Abhishek** and E. George Dharma Prakash Raj, *Operating System Security: A Short Note*, IEEE India Info. Vol. 14 No. 2 Apr - Jun 2019.

6.2 Introduction

The design goals of RNG heavily depend on its target applications. A simple application like stochastic simulations or Monte Carlo integrations may require RNG to generate nothing more than a random looking bitstream [**22**]. However, a sensitive application of RNG like an operating system kernel on top of which entire critical systems run, certainly requires RNG to generate high quality pseudo random bitstreams which are also provably secure, unpredictable and **must be non-reproducible** which only a True Random Number Generator (TRNG) can provide in principal.

Moreover, a kernel uses a RNG to create ASLR offsets [**25**], generate salts to securely store users passwords [**88**] and generate random keys to implement various cryptographic primitives like encryption, authentication etc. The ASLR is one of the most important techniques used by the kernel (in special cases termed as Kernel-ASLR or KASLR) which randomizes the process layout to protect the locations of the targeted functions such as stack, heap, executable, dynamic linker/loader etc. [**25**]. The ASLR not only demands statistically qualified high quality pseudorandom number generator but also requires the output bitstream to be provably secure and unpredictable. Hence, a CSPRNG (or simply a PRNG with regular entropy inputs for unpredictability) is a preferred type of RNG for kernel applications. There are many good CSPRNGs which

are implemented in various operating systems and are used by their kernels. Fortuna, Yarrow and /dev/(u)random are the popular CSPRNGs which are currently implemented by Windows, MacOs/iOS/FreeBSD and Linux/Android operating systems respectively [**67**, **89**]. In this thesis, a new CSPRNG which exhibits 'non-reproducibility' property of a TRNG is proposed taking security of the above kernel applications into consideration.

In particular, the key contributions of this chapter are:

- A novel CSPRNG design comprises of two non-standard and verified secure elliptic curves and nine LFSRs uniquely configured in a clock-controlled fashion to attain exponential linear complexity is used to construct the proposed KCS-PRNG.
- A novel architecture of the KCS-PRNG is proposed to mitigate the gap of 'non-reproducibility' property.
- Two new non-standard and verified elliptic curves are used in this chapter (as described in Chapter 5) to mitigate the gap of 'non-reproducibility' property of the generated pseudorandom bitstreams by the proposed KCS-PRNG. Both elliptic curves are generated randomly over 256-bit prime fields to ensure cryptographic and implementation security and randomly retrieved from a newly created database of such elliptic curves.
- Extensive security analysis of the proposed KCS-PRNG carried out to ensure theoretical security.
- Experimental validation and demonstration of statistical qualities of randomness using NIST, Diehard, TestU01 test suites.
- Experimental validation and demonstration of 'non-reproducibility' property of the proposed KCS-PRNG.

- The proposed KCS-PRNG is compared with present kernel CSPRNGs like Fortuna, Yarrow and dev/random and an existing PRNG [**90**]. The proposed KCS-PRNG is also compared with an existing TRNG [**91**] in context of non-reproducibility of the generated random bitstreams.

6.3 The Proposed Design of KCS-PRNG

Generation of high quality cryptographically secure pseudorandom bitstreams is an intricate task which needs efficient design of the generator taking statistical properties of randomness ($R1$), unpredictability ($R2$, $R3$) and non-reproducibility ($R4$) of the output bitstreams into consideration (refer Section 1.4.6). For this reason, the proposed KCS-PRNG binds two modules in its design: first, a combination of two cryptographically safe elliptic curves and a nonlinear Sequence Generator consisting of nine clock-controlled LFSRs in alternating step configuration.

Following are the design decisions and assumptions of the proposed KCS-PRNG:

6.3.1 Selection of Elliptic curves

The main motivation of using elliptic curves in the proposed KCS-PRNG instead of stream ciphers/block ciphers like ChaCha20 and Triple DES or AES respectively as used by /dev/(u)random [**62**], Yarrow [**64**] and Fortuna [**66**] respectively is that one can choose different points on the selected elliptic curve to generate completely unrelated bitstreams under identical start conditions using the novel configuration used in the proposed design of KCS-PRNG. Moreover, the combination of elliptic curve and LFSR has been proven to exhibit enhanced randomness properties [**12**]. Two elliptic curves are used in KCS-PRNG for added complexity where each

elliptic curve provides nearly 2^{128} key space. Moreover, the advantages of keeping elliptic curves with the clock-controlled LFSRs is two-fold: first, the elliptic curves are used for mitigating the gap of 'non-reproducibility' property ($R4$). Second, elliptic curves are used to generate bitstreams which are non-invertible due to underlying hard ECDLP and hence, they make the proposed KCS-PRNG provably secure as well as forward secure to resist backtracking attacks. However, the choice of elliptic curves is considered to be a randomly generated one rather than the standard elliptic curves with fixed coefficients as being recommended by agencies like NIST [**32**], Brainpool [**34**] etc. The random derivation of elliptic curve parameters ensures trust and transparency in the implementation of elliptic curves [**51**]. Two elliptic curves selected for use in the KCS-PRNG are presented in Section 6.6 of this chapter whose computational details are described in Chapter 5 of this thesis.

6.3.2 Selection of a Clock-controlled LFSRs

The proposed KCS-PRNG is targeted for integration in the operating system kernel and therefore, it is implemented in software. However, implementation of LFSR in software is slower than its hardware implementation [**24**, **93**]. To address this performance issue, the Galois scheme is selected for optimal performance gain by the LFSRs in software without compromising the LFSR period and its cryptographic properties [**24**]. The chosen Galois configuration also saves excess operations as all the XOR operations are performed as a single operation [**24**]. A nonlinear Sequence Generator consisting of nine LFSRs $L_1, L_2, L_3, L_4, L_5, L_6, L_7, L_8$ and L_9 with corresponding primitive polynomial degrees 29, 31, 37, 41, 43, 47, 53, 59 and 61 respectively is selected. The primitive polynomials for these LFSRs feedback functions are

$$L_1 = x^{29} + x^{25} + x^{21} + x^{17} + x^{14} + x^{10} + x^6 + x^3 + 1,$$

$$L_2 = x^{31} + x^{27} + x^{23} + x^{19} + x^{15} + x^{11} + x^7 + x^3 + 1,$$
$$L_3 = x^{37} + x^{32} + x^{27} + x^{23} + x^{18} + x^{13} + x^9 + x^5 + 1,$$
$$L_4 = x^{41} + x^{36} + x^{31} + x^{26} + x^{20} + x^{15} + x^{10} + x^5 + 1,$$
$$L_5 = x^{43} + x^{37} + x^{31} + x^{25} + x^{20} + x^{15} + x^{10} + x^5 + 1,$$
$$L_6 = x^{47} + x^{41} + x^{35} + x^{29} + x^{23} + x^{17} + x^{11} + x^5 + 1,$$
$$L_7 = x^{53} + x^{46} + x^{40} + x^{33} + x^{26} + x^{19} + x^{13} + x^7 + 1,$$
$$L_8 = x^{59} + x^{52} + x^{44} + x^{36} + x^{29} + x^{22} + x^{14} + x^7 + 1,$$
$$L_9 = x^{61} + x^{53} + x^{45} + x^{38} + x^{30} + x^{23} + x^{15} + x^7 + 1.$$

These primitive polynomials used by the nine LFSRs have uniformly distributed feedback coefficients selected from Rajski *et. al.* [**94**]. These nine LFSRs $L_1, L_2, \cdots, L_9$ are further divided into three groups called Sequence Generator 1 (SG_1), Sequence Generator 2 (SG_2) and Sequence Generator 3 (SG_3). SG_1 has three LFSRs L_1, L_2 and L_3 whose output streams x_1, x_2 and x_3 are combined nonlinearly using nonlinear function

$$y_1 : f(x_1, x_2, x_3) = x_1x_2 \oplus x_2x_3 \oplus x_3x_1 \tag{6.1}$$

The resulting sequence y_1 has period $(2^{L_1} - 1)(2^{L_2} - 1)(2^{L_3} - 1)$ and linear complexity $(L_1L_2 + L_2L_3 + L_1L_3)$. Similarly, from equation (6.1), the linear complexities of the sequences y_2 and y_3 generated from SG_2 and SG_3 are $(L_4L_5 + L_5L_6 + L_6L_4)$ and $(L_7L_8 + L_8L_9 + L_9L_7)$ respectively. It may be noted that the initial state bits of all LFSRs together are $\sum_{i=1}^{9} L_i$ = 401 bits.

SG_1, SG_2 and SG_3 are configured in alternating step scheme to provide high linear complexity and large period to the Sequence Generator [**95**]. SG_1 is considered as the Controller of the Sequence Generator in the alternating step mode. It is known that the linear complexity $LC(x)$ of the overall alternating step

generator is bounded as follows [**95**]:

$$(LC_2 + LC_3)^{2LC_1-1} < LC(x) \leq (LC_2 + LC_3)^{2LC_1} \tag{6.2}$$

where LC_1, LC_2 and LC_3 are the linear complexities of SG_1, SG_2 and SG_3 respectively. The Alternating Step Sequence Generator used in the proposed KCS-PRNG is depicted in Figure 6.1 and described in Algorithm 8 [**95**].

6.3.3 The Proposed Novel KCS-PRNG Architecture

The proposed KCS-PRNG architecture is shown in Figure 6.2. The KCS-PRNG uses a Field Converter, Elliptic curve Point Multiplication and a Selector in addition to the Sequence Generator and two elliptic curves in its design.

Algorithm 8 Alternating Step Sequence Generator using Clock-controlled LFSRs

Require: Sequence Generators SG_1, SG_2 and SG_3, output bit length n
Ensure: n-bit sequence

1. **Loop** n-times:
2. SG_1 is clocked
2.1. **if** output of SG_1 is 1
2.1.1 SG_2 is clocked ▷ SG_3 is not clocked but its previous output bit is repeated. In case of the first clock cycle, previous output bit of SG_3 is taken as 0.
2.2. **else**
2.2.1. SG_3 is clocked ▷ SG_2 is not clocked but its previous output bit is repeated. In case of the first clock cycle, previous output bit of SG_2 is taken as 0.
2.3. **end if**
3. **return** $SG_2 \oplus SG_3$ ▷ Output of Sequence Generator in alternating step
4. **end Loop**

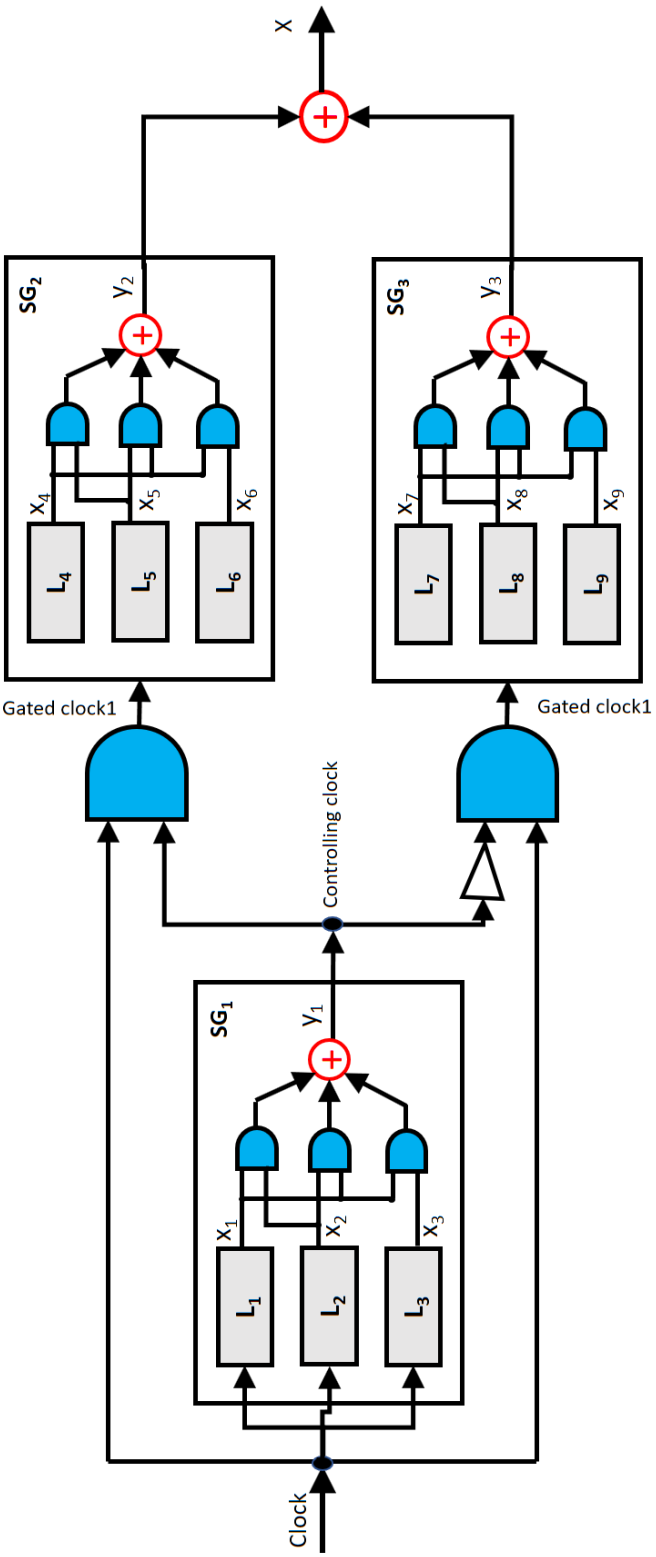


Figure 6.1: Sequence Generator in Alternating Step Configuration

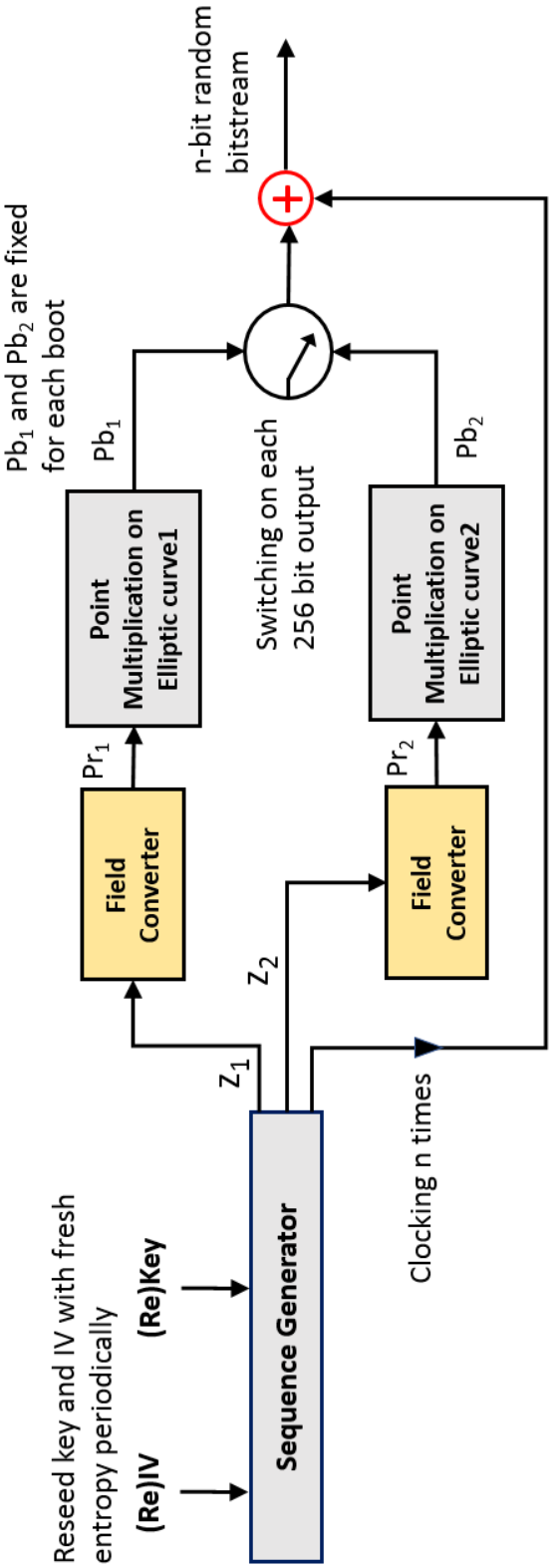


Figure 6.2: The proposed KCS-PRNG Architecture

Algorithm 9 Selection of 2 Elliptic curves

Require: Look-up Table $\mathcal{T}(EC_n, EC_n\_ID\_Status)$ where n is number of elliptic curves

Ensure: 2 Elliptic curves EC_r, EC_s from $\mathcal{T}$ where $r, s \in [1, n]$ and $r \neq s$

1. Count n ▷ Elliptic curves with $EC_n\_ID\_Status = 0 \; \forall \; n$ in $\mathcal{T}$
2. **if** $n \geq 2$

2.1. Fetch EC_r, EC_s from $\mathcal{T}$ where $EC_r\_ID\_Status = 0$ and $EC_s\_ID\_Status = 0$

2.2. Set $EC_r\_ID\_Status \longleftarrow 1$, $EC_s\_ID\_Status \longleftarrow 1$

2.3. Update $\mathcal{T}$

2.4. **return** EC_r, EC_s

3. **else**

3.1. Set $EC_n\_ID\_Status = 0 \; \forall \; n$ in $\mathcal{T}$ ▷ n is the number of elliptic curves in $\mathcal{T}$

3.2. Go to step 1

4. **end if**

The two elliptic curves are selected using the procedure as shown in Algorithm 9. A look-up table $\mathcal{T}$ with tuples (EC, EC_ID_Status) is created by retrieving elliptic curves from the database as discussed in Section 4.7.3 where EC is the elliptic curve and EC_ID_Status is the flag value to mark 0 for 'un-used curve' and 1 for the 'used curve'. $\mathcal{T}$ consists of 500 elliptic curves initially which are randomly generated and are cryptographically secure non-standard curves. All elliptic curves in $\mathcal{T}$ are initially marked with $EC\_ID\_Status = 0$. On each reboot of the proposed KCS-PRNG, it picks up two elliptic curves from $\mathcal{T}$ using Algorithm 9 and sets the corresponding $EC\_ID\_Status = 1$ of both the used elliptic curves in $\mathcal{T}$. The advantage of $\mathcal{T}$ is that even if the same seed (entropy) is supplied to the proposed KCS-PRNG on reboot of the generator, two new elliptic curves with $EC\_ID\_Status = 0$ will be selected from $\mathcal{T}$. The change of elliptic curves on each reboot of the KCS-PRNG changes the final output

by altering masking value between the output bits of the elliptic curves and the Sequence Generator. Hence, entirely unrelated bitstream are obtained as the output of the proposed generator even using exactly the same seed as input. When all elliptic curves in $\mathcal{T}$ are used then EC_ID_Status flags are reset to 0 for all elliptic curves in $\mathcal{T}$ in order to maintain unblocked supply of elliptic curves to the KCS-PRNG. More elliptic curves can be inserted into $\mathcal{T}$ to consistently mitigate the requirement of 'non-reproducibility' property $R4$ of the KCS-PRNG. Here, the mitigating factor of the the RNG requirement $R4$ is directly proportional to the number of un-used elliptic curves available in $\mathcal{T}$. This idea makes the proposed KCS-PRNG to mitigate the RNG requirement $R4$ to a practical extent.

6.3.4 Initialization of KCS-PRNG

The proposed KCS-PRNG uses two phases of pseudorandom bitstreams generation. In the first phase, the Sequence Generator is initialized whereas in the second phase, the desired length of pseudorandom bitstreams are generated using the Sequence Generator and the elliptic curves. The initialization phase involves two stages which includes, first, loading the key and initialization vector (IV) in to the generator and second, diffusing the key-IV pair across the entire states of the Sequence Generator [**96**] as described in the Algorithm 10 and as shown in Figure 6.3 and Figure 6.4 .

Algorithm 10 takes 574-bit of entropy bits which are harvested from various physical non-deterministic noise sources and generates 401-bit of key and 173-bit of Initialization Vector (IV). The key is first parallelly loaded in to SG_1, SG_2 and SG_3 of the Sequence Generator as shown in step 1. It is ensured that all the most significant bits (MSBs) of L_1, L_2 and L_3 will be set to 1 in step 2 and step 3. The Sequence Generator is then clocked 128 times so that the key is diffused across the entire states of all the nine LFSRs $L_1, L_2, \cdots, L_9$ and a new state of the Sequence

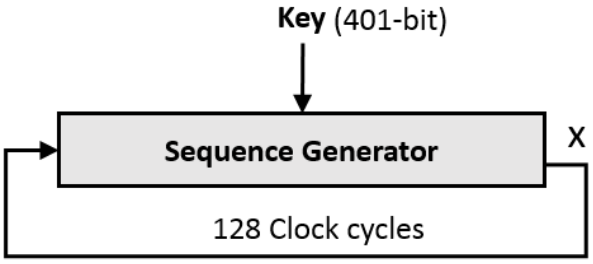


Figure 6.3: Initialization Stage 1: Loading and diffusion of the key

Generator is obtained in step 4 as shown in Figure 6.3. In steps 5, 6 and 7, a 173-bit IV is loaded in to L_1, L_2 and L_3 of SG_1 in bitwise fashion by XORing with the corresponding usual feedback bit of the LFSR and the output bit of the Sequence Generator to feedback the LFSRs through MSBs as shown in Figure 6.4. In step 8, the Sequence Generator is once again clocked 128 times to diffuse the IV completely among the LFSRs in SG_1 and gets entirely new states of all the nine LFSRs. It is ensured that the MSBs of all the nine LFSRs $L_1, L_2, \cdots, L_9$ are set to 1 as shown in steps 9 and 10. Step 11 returns the initialized Sequence Generator.

<This space is intentionally left blank.>

Algorithm 10 Initialization of Sequence Generator

Require: 401-bit entropy for Key and 173-bit entropy for Initialization Vector (IV)
Ensure: Initialized Sequence Generator

1. Initialize SG_1, SG_2 and SG_3 with 401-bit Key ▷ Stage 1: Loading LFSRs from the input Key
2. **if** MSB of any LFSR is 0

2.1. Ensure MSB of LFSR as 1

3. **end if**
4. Clock Sequence Generator 128 times ▷ Stage 2: Diffusion of key into all LFSRs states in the Sequence Generator
5. **Loop** 173 times:
6. Clock SG_1 with feedback = Feedback bit $\oplus$ IV bit $\oplus$ output bit of Sequence Generator ▷ Stage 1: Loading 173-bit IV to SG_1
7. **end Loop**
8. Clock Sequence Generator 128 times ▷ Stage 2: Diffusion of IV into all LFSRs states in SG_1
9. **if** MSB of the Sequence Generator is 0

9.1. Ensure MSB of the Sequence Generator as 1

10. **end if**
11. **return** Initialized Sequence Generator

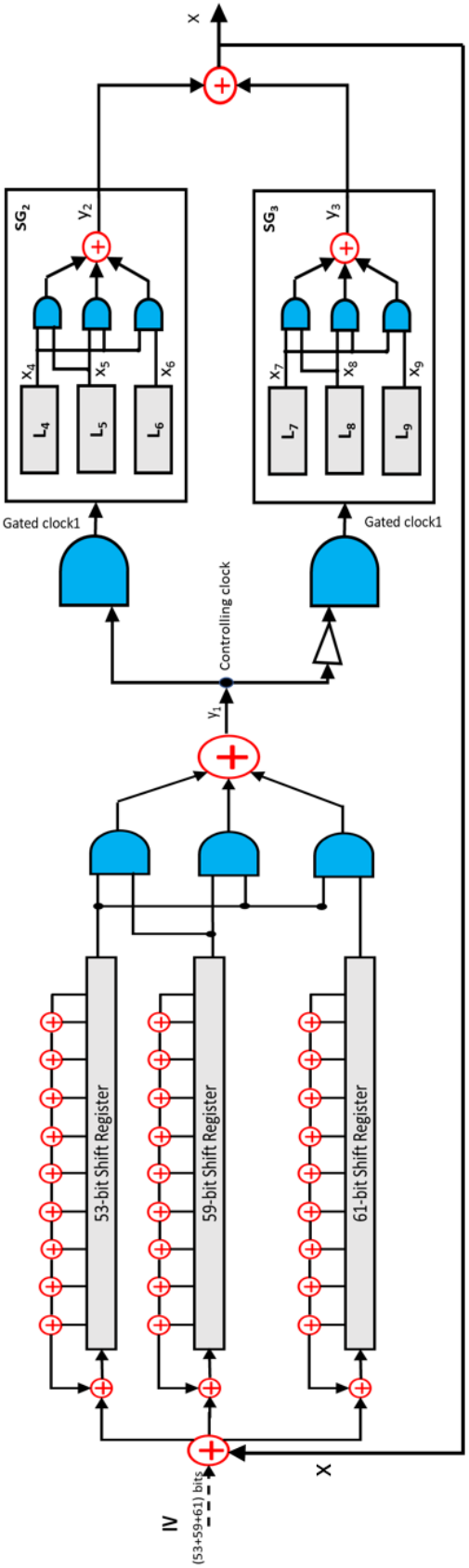


Figure 6.4: Initialization Stage 2: Loading of IV

6.3.5 KCS-PRNG Bitstream Generation

The Sequence Generator generates two sequences z_1 and z_2 of 256-bit length each of which is used by the field converter as the inputs. The field converter transforms z_1 and z_2 into integers and then transforms them into the field elements P_{r_1} and P_{r_2} of the two elliptic curves. These field elements or the secrets P_{r_1} and P_{r_2} are given as inputs to the two elliptic curve point multiplication functions as described in Algorithm 11. The secrets P_{r_1} and P_{r_2} are multiplied with their corresponding base points G_1 and G_2 which yields a new point on their respective elliptic curves. The x-coordinates of the two points obtained are only assigned to as the two integers P_{b_1} and P_{b_2} after transformation from the field elements. A selector is used to switch between the outputs of the two elliptic curves point multiplication functions to double the size of key space offered by the proposed KCS-PRNG.

Algorithm 11 Elliptic curve point multiplication

Require: Secrets P_{r_1} and P_{r_2} for 2 elliptic curves
Ensure: Points P_{b_1} and P_{b_2} of 2 elliptic curves in integer form

1. $P_{b_1} \longleftarrow G_1 \times P_{r_1}$ ▷ G_1 is the base point selected on first elliptic curve and P_{b_1} is the x-coordinate of P_{b_1}
2. $P_{b_1} \longleftarrow Integer(P_{b_1})$ ▷ $Integer()$ is transformation function from field to integer
3. $P_{b_2} \longleftarrow G_2 \times P_{r_2}$ ▷ G_2 is the base point selected on second elliptic curve and P_{b_2} is the x-coordinate of P_{b_2}
4. $P_{b_2} \longleftarrow Integer(P_{b_2})$ ▷ $Integer()$ is transformation function from field to integer
5. **return** P_{b_1}, P_{b_2}

Algorithm 12 describes the cryptographically secure pseudorandom bitstream generation scheme of the proposed KCS-PRNG. In step 1, two elliptic curves with hard ECDLP are selected from $\mathcal{T}$. In step 2, the Sequence Generator is initialized with 401-bit key and 173-bit IV as discussed in Algorithm 10. The Sequence Generator is used to generate 256-bit sequence z_1 by clocking 256 times in step

3. In step 4, z_1 is converted into the field element of the first elliptic curve and considered as the secret P_{r_1}. Step 5 involves generation of P_{b_1} by using elliptic curve point multiplication function taking the secret P_{r_1} as input. Similarly, steps 6, 7 and 8 are used to generate the integer P_{b_2} from the second elliptic curve point multiplication function. The Sequence Generator continuously generates n-bit length sequences as bounded by $\lceil \frac{n}{256} \rceil$ times loop in step 12. The proposed KCS-PRNG uses a selector in step 13 to iteratively select among P_{b_1} and P_{b_2}. In step 15, the Sequence Generator is clocked 256 times to generate 256-bit sequence s. The integers P_{b_1} or P_{b_2} is masked with s to produce 256-bit output by the KCS-PRNG. If $n < 256$, then 1-bit output of the Sequence Generator is masked with 1-bit of P_{b_1} or P_{b_2} (as decided by the selector in step 13) traversing from its Least Significant Bit (LSB) to MSB and result is returned. Once MSB of the P_{b_1} or P_{b_2} is used, the masking of the output of the Sequence Generator starts from the LSB of the P_{b_1} or P_{b_2} once again in rotating fashion. The KCS-PRNG is reseeded on every 100000 bit of output to maintain backward secrecy as shown in step 22.

<This space is intentionally left blank.>

Algorithm 12 The proposed KCS-PRNG bitstream generation

Require: Desired length of bitstream n, $(574 \times r)$-bit entropy for key and IV where $r = \lceil \frac{n}{100000} \rceil$ = number of (re)seeding required for KCS-PRNG

Ensure: n-bit cryptographically secure pseudorandom bitstream

1: Run Algorithm 9 to select two elliptic curves from $\mathcal{T}$

2: Run Algorithm 10 with input of 401-bit key and 173-bit IV to initialize the Sequence Generator

3: Run Algorithm 8 to generate 256-bit sequence z_1

4: Transform z_1 into field element P_{r_1} of first elliptic curve using field converter

5: Run Algorithm 11 with P_{r_1} as input to generate the integer P_{b_1}

6: Run Algorithm 8 to generate 256-bit sequence z_2

7: Transform z_2 in to field element P_{r_2} of second elliptic curve using field converter

8: Run Algorithm 11 with P_{r_2} as input to generate the integer P_{b_2}

9: Set $countSel = 1$

10: Set $bitCount = 1$

11: Set $t = 1$ where $t = 1$ to $\lceil \frac{n}{256} \rceil$

12: **Loop** $\lceil \frac{n}{256} \rceil$ times:

continued to next page..

Algorithm 12 The proposed KCS-PRNG.. (continued from previous page)

13: **if** $countSel == t \times 256$ ▷ Use Selector to select between the two elliptic curves

13.1. **if** t is even

13.1.1. Set $el = P_{b_2}$

13.1.2. **else**

13.1.3. Set $el = P_{b_1}$

13.2. **end if**

13.3. $countSel = 0$

13.4. t^{++}

14: Clock Sequence Generator 256 times to generate 256-bit sequence s

15.1 **if** $n < 256$

15.2 **return** $X \oplus i^{th}$ position of el from LSB ($i = 0$) to MSB ($i = 255$) where X is 1-bit output from Sequence Generator and i = 0 to 255 ▷ Output of KCS-PRNG

15.3 **else**

15.4 **return** $el \oplus s$ ▷ Output of KCS-PRNG

15: **end if**

16: **end if**

17: i^{++}

continued to next page..

Algorithm 12 The proposed KCS-PRNG.. (continued from previous page)

18: **if** $i == 255$

18.1. $i = 0$

19: **end if**

20: $countSel^{++}$

21: $bitCount^{++}$

22: **if** $bitCount == j \times 100000$ where $j = 1$ to r

22.1. $n = n - (j \times 100000)$

22.2. j^{++}

22.3. Go to step 2 ▷ Reseed the KCS-PRNG on every 100000 bits of output

23: **end if**

24: **end Loop**

6.3.6 Assumptions

Following assumptions are made in the proposed design of KCS-PRNG:

- KCS-PRNG always maintains 574-bit initial entropy.
- KCS-PRNG expects high per-bit entropy $\approxeq$ 1 for initialization. The generation details of entropy used in KCS-PRNG is outside the scope of this thesis and planned as a topic of future research.
- The key and IV are parts of the seed and hence, they are immediately shredded after use and is non-recoverable.
- The (Re)keying and (Re)IVing are done using different TRNGs or entropy harvesters using various different physical noise sources.

- Elliptic curves used in KCS-PRNG are randomly generated, cryptographically safe and trustworthy.
- Look-up Table $\mathcal{T}$ has authorized access only.

6.4 Security Analysis of the proposed KCS-PRNG

6.4.1 Linear complexity analysis

Let linear complexities of the Sequence Generators SG_1, SG_2 and SG_3 be LC_1, LC_2 and LC_3 respectively and following equation (6.1), are given by

$$\begin{aligned} LC_1 &= L_1L_2 + L_2L_3 + L_1L_3 = 3119 \\ LC_2 &= L_4L_5 + L_5L_6 + L_4L_6 = 5711 \\ LC_3 &= L_7L_8 + L_8L_9 + L_7L_9 = 9959 \end{aligned} \tag{6.3}$$

where $L_1, L_2, \cdots, L_9$ are the lengths of the LFSRs.

Moreover, while SG_1 is clocked regularly, SG_2 and SG_3 are connected in alternating step configuration. Thus, following equation (6.2), the overall linear complexity (LC) of the scheme is given by

$$\begin{aligned} &(5711 + 9959)^{2\times 3119-1} < LC(x) \leq (5711 + 9959)^{2\times 3119} \\ \implies &\ 15670^{6237} < LC(x) \leq 15670^{6238} \end{aligned} \tag{6.4}$$

It is imperative to note that the Sequence Generator of the proposed KCS-PRNG exhibits exponentially large linear complexity as demonstrated in equation (6.4) and therefore, the proposed generator is resistant to the Berlekamp-Massey attack [**95**].

6.4.2 Correlations test

Two correlation tests of random bitstreams generated by the proposed KCS-PRNG are conducted to verify non-correlation in the bitstream. The first test conducted was Serial or Autocorrelation test ($sstring - AutoCor$ test) which measures the correlation between the bits with the lag d [**97**]. In this test, a n-bit string is generated by the KCS-PRNG at the first level and the test statistic is computed such that it has the binomial distribution with the parameters being approximately standard normal for large $n - d$. The restriction imposed were $r + s \leq 32$ and $1 \leq d \leq \lfloor \frac{n}{2} \rfloor$ where r be the number of MSBs which are eliminated from the output before applying the test, s be the MSBs chosen from each generated random number and N be second-level number of replications [**97**, **98**]. The second test conducted was the Hamming Correlation test ($sstring - HammingCorr$) [**98**] in which the bit sequences generated by the proposed KCS-PRNG were verified for exhibiting uniform bits distribution without correlation. Both the Autocorrelation test and the Hamming Correlation test were conducted during TestU01 testing of the proposed generator. The proposed KCS-PRNG passed both the tests. Further, ENT tool [**99**] was used to measure bitwise correlation in the random bitstream file of 1GB size generated by the proposed KCS-PRNG which was estimated to be 0.000034. The obtained correlation is very close to the ideal correlation value of 0.0 and thus, concludes that the proposed design of the KCS-PRNG has no correlation issues and their results are shown in Table 6.1.

<This space is intentionally left blank.>

Table 6.1: Correlation test of the proposed KCS-PRNG.

sstring-AutoCor test	**N=1, n=1048513, r=0, s=32, d=1**
Normal statistic	0.41
p-value of test	0.34
Number of bits used	1048544
Result	Passed the test
sstring-AutoCor test	**N=1, n=1048514, r=0, s=32, d=2**
Normal statistic	0.80
p-value of test	0.21
Number of bits used	1048544
Result	Passed the test
sstring-HammingCorr test	**N=1, n=32768, r=0, s=32, L=32**
Normal statistic	-0.56
p-value of test	0.71
Number of bits used	1048576
Result	Passed the test
sstring-HammingCorr test	**N=1, n=16384, r=0, s=32, L=64**
Normal statistic	0.45
p-value of test	0.33
Number of bits used	1048576
Result	Passed the test
sstring-HammingCorr test	**N=1, n=8192, r=0, s=32, L=128**
Normal statistic	1.57
p-value of test	0.06
Number of bits used	1048576
Result	Passed the test

6.4.3 Period analysis (Validation of Requirement R1)

The Sequence Generator used in the KCS-PRNG comprises of nine LFSRs whose lengths $L_1, L_2, \cdots, L_9$ are coprime to each other. Hence, the period (P) of the Sequence Generator is given by

$$P = \prod_{i=1}^{9} (2^{L_i} - 1) \tag{6.5}$$

which is approximately 2^{401}.

6.4.4 Key space analysis

It is evident from equation (6.5) that the Sequence Generator in KCS-PRNG has a period of 2^{401} and thus, provides 2^{401} key space in case the generator gets seeded once and no reseeding happens. Moreover, the KCS-PRNG also uses two elliptic curves which provides 2^{128} and 2^{256} key space for $n \leq 256$ and $n > 256$ bits of output respectively to impose a successful Pollard's rho attack to solve the ECDLP. Hence the key space offered by the proposed KCS-PRNG is given by

$$\mathcal{K} = \begin{cases} (2^{401} \times 2^{128})^r = 2^{529r} & if(n \leq 256) \\ (2^{401} \times 2^{256})^r = 2^{657r} & if(n > 256) \end{cases} \tag{6.6}$$

where r be the number of (re)seeding the KCS-PRNG and n be the number of output bits of the proposed KCS-PRNG.

It is imperative to note that the key space offered by the proposed KCS-PRNG depends on the number of times the KCS-PRNG (re)seeds itself in single boot and therefore, exhibits virtually infinite key space in the range $\mathcal{K} \in [2^{529}, \infty)$ which is quite higher than the safe key space threshold of 2^{128} as recommended by [**90**, **100**]. Therefore, the proposed KCS-PRNG comfortably resists brute force attacks.

6.5 Experimental Validation of the Proposed KCS-PRNG

6.5.1 Experimental Validation of Requirement $R1$

i. NIST statistical test results

NIST test suite consists of 15 statistical tests to certify statistical strength of randomness of the RNG. An output bitstream of 1GB file size is generated by the proposed KCS-PRNG and subjected to the NIST tests using NIST statistical test suite SP 800-22 version 2.1.2 [**101**]. The input block size was set to be 1000000 bits and 1000 bitstreams. The significance level α was selected as 99% to conduct the test. The proposed KCS-PRNG passed all the NIST statistical tests and the details of test results obtained are depicted in Table 6.2.

The p-value measures randomness and supposed to be greater than 0.01 i.e., the confidence level to conclude that the sequence is uniformly distributed whereas the proportion i.e., the minimum pass rate for the test should fall in the range [0.98056, 0.99943] having the confidence interval α=0.01 and 1000 bitstreams [**91**]. As indicated in Table 6.2, the proposed KCS-PRNG not only qualifies the pass rate threshold of 0.98056 but also reports better pass rate of 0.9896 as compared to the pass rates of 0.987 and 0.9887 reported by the TRNG [**91**] and the PRNG [**90**] respectively.

<This space is intentionally left blank.>

Table 6.2: NIST test results of the proposed KCS-PRNG output bitstreams of 1GB file size with the input of 1000000-bit block size and 1000 bitstreams.

Statistical Test	**p-value**	**Proportion**	**Result**
Frequency	0.737915	0.991	Pass
Block Frequency	0.591409	0.988	Pass
CumulativeSums*	0.680755	0.993	Pass
Runs	0.281232	0.992	Pass
Longest Run	0.526105	0.996	Pass
Rank	0.036113	0.996	Pass
FFT	0.103138	0.990	Pass
NonOverlappingTemplate*	0.794391	0.990	Pass
Overlapping	0.779188	0.987	Pass
Universal	0.773405	0.991	Pass
Approx Entropy	0.653773	0.989	Pass
RandomExcursions*	0.489508	0.983	Pass
RandomExcursionsVariant*	0.163362	0.985	Pass
Serial*	0.680755	0.988	Pass
Linear Complexity	0.682823	0.985	Pass

*Only the result of first test instance is indicated here from the original results due to limitation of space.

ii. Diehard test results [**102**]

Diehard version 3.31.1 tests conduct a series of statistical tests and determine the p-values of the output bitstreams. The p-values indicate deviation of bit prediction from ideally expected probability of half. The expected p-value of a test should be in the range [0.025, 0.975] [**103**]. The proposed KCS-PRNG passed all the diehard tests as shown in Table 6.3.

Table 6.3: Diehard test results of the proposed KCS-PRNG output bitstreams of 1GB file size.

test-name	ntup	tsamples	psamples	p-value	Assessment
diehard-birthdays	0	100	100	0.27561288	Passed
diehard-operm5	0	1000000	100	0.13184067	Passed
diehard-rank-32x32	0	40000	100	0.44295780	Passed
diehard-rank-6x8	0	100000	100	0.88076181	Passed
diehard-bitstream	0	2097152	100	0.42947798	Passed
diehard-opso	0	2097152	100	0.12604767	Passed
diehard-oqso	0	2097152	100	0.94641900	Passed
diehard-dna	0	2097152	100	0.24390543	Passed
diehard-count-1s-str	0	256000	100	0.62287409	Passed
diehard-count-1s-byt	0	256000	100	0.91047395	Passed
diehard-parking-lot	0	12000	100	0.79390338	Passed
diehard-2dsphere	2	8000	100	0.17731451	Passed
diehard-3dsphere	3	4000	100	0.45129204	Passed
diehard-squeeze	0	100000	100	0.53561994	Passed
diehard-sums	0	100	100	0.94209561	Passed
diehard-runs*	0	100000	100	0.14811353	Passed
diehard-craps*	0	200000	100	0.92115680	Passed
marsaglia-tsang-gcd*	0	10000000	100	0.53120802	Passed
sts-monobit	1	100000	100	0.64501072	Passed
sts-runs	2	100000	100	0.94961272	Passed
sts-serial*	1	100000	100	0.62077367	Passed
rgb-bitdist*	1	100000	100	0.95378266	Passed
rgb-minimum-distance*	2	10000	1000	0.87517368	Passed
rgb-permutations*	2	100000	100	0.75286377	Passed
rgb-lagged-sum*	0	1000000	100	0.00308570	Passed
rgb-kstest-test	0	10000	1000	0.03414230	Passed
dab-bytedistrib	0	51200000	1	0.17158919	Passed
dab-dct	256	50000	1	0.07312246	Passed
dab-filltree*	32	15000000	1	0.61801753	Passed
dab-filltree2*	0	5000000	1	0.69361846	Passed
dab-monobit2	12	65000000	1	0.42742922	Passed

*Only the result of first test instance is indicated here from the original results due to limitation of space.

iii. TestU01 test results [**97**]

TestU01 is believed to impose the toughest tests to evaluate the statistical quality of random bitstreams [**90**]. The binary bitstream of 1GB file size generated by the proposed KCS-PRNG is subjected to the Rabbit and Alphabit test batteries of TestU01. The Rabbit and the Alphabit, by default, selected 1048576 bits (2^{20} bits) for SmallCrush (a fast statistical test battery) evaluation and applied 38 and 17 statistical tests respectively to the proposed KCS-PRNG output bitstream. The output bitstreams of KCS-PRNG are found to have p-values within the acceptable range of [0.001, 0.999] [**103**] which proved that the proposed KCS-PRNG exhibits long period, good structure and non-linearity.

6.5.2 Validation of Requirements R2 and R3

i. Next bit test

This test states that if a sequence of m-bits is generated by a generator, there should not be any feasible method which can predict the $(m+1)$th bit with the probability significantly higher than half [**104**, **105**]. This test is associated with predictability of the successive bits generated by the KCS-PRNG.

Since the KCS-PRNG is reseeded with fresh additional entropy of 574 bits (401 bits of key and 173 bits of IV), therefore, it maintains backward security [**65**].

ii. Test for state compromise extension attacks

This test states that if some state of a generator is leaked at a given time to an attacker, it would not be possible to recover unknown PRNG outputs from that known state [**106**]. Fundamentally, the state compromise extension

imposes two kinds of attack: first, a backtracking attack to learn previous outputs of the generator knowing some internal state of the generator at a particular time and second, the permanent compromise attack which enables all the future and past states of the generator vulnerable with the knowledge of some state at a given time [**106**].

Since the proposed KCS-PRNG is forward secure and provably secure due to underlying ECDLP intractability, therefore, it is resistant to the backtracking attack. Furthermore, as discussed in the next bit test, the proposed KCS-PRNG is (re)seeded on every 100000 bits of output generation, therefore, it exhibits backward secrecy and thus, resists the permanent compromise attack as well.

iii. Entropy Estimation (Experimental Validation of Requirement R2, R3) Entropy is the measurement of unpredictability or uncertainty. For an ideal TRNG, the expected entropy is 1 per bit which means that each bit i.e., '0' or '1' have equal proportion 0.5 in the file containing random bitstream [**91**]. The proposed KCS-PRNG is subjected to ENT tool [**99**] for estimation of the entropy of the KCS-PRNG generated 1GB file of random bitstream. The observed value of the entropy of output bitstream generated by the proposed KCS-PRNG is found to be 0.99999975 per bit which asserts that the design of KCS-PRNG maintains nearly an ideal unpredictability.

6.5.3 Experimental Validation of Requirement R4

Non-reproducibility test

The non-reproducibility test is conducted to validate if the RNG requirement $R4$ is met by the proposed KCS-PRNG. This test is conducted by running the generator twice with exactly the same input and verifying if the output sequences

are completely unrelated. Authors [**91**] have referred the non-reproducibility test as the restart test and they validated the first 20 bit output sequences of the generator six times under identical start conditions. Table 6.4 shows that the proposed KCS-PRNG has passed the non-reproducibility test six times by producing six completely unrelated 32 bits using the same inputs to the proposed generator.

Table 6.4: Non-reproducibility test of the proposed KCS-PRNG under identical start conditions.

Key Input (401-bit entropy)	1905119BCDC809077DB45D 1B3921DB5C06D11 C56C7FE B4F8EE935A2FB16B055281816 DFC551AC73C3BBF76EE26B13 0B8F5E68
IV Input (173-bit entropy)	190B6B491CDD9E97E6AB 26552990F5481183DEF9AE55
Check	First run of KCS-PRNG
32-bit Output	01010100111011111110001110100100
Check	Second run of KCS-PRNG
32-bit Output	00010010000100001111001111111110
Check	Third run of KCS-PRNG
32-bit Output	11000101110001101011100101111101
Check	Fourth run of KCS-PRNG
32-bit Output	01101010010110101011000010110101
Check	Fifth run of KCS-PRNG
32-bit Output	10110001000111011001101100011011
Check	Sixth run of KCS-PRNG
32-bit Output	01001100110010111100010011100110

Moreover, the KCS-PRNG uses two different elliptic curves on each boot and therefore, the output bitstream would be entirely unrelated even generated under identical start conditions. Hence, it is inferred that the proposed

KCS-PRNG generates non-reproducible pseudorandom bitstreams, provided it maintains minimum number of un-used elliptic curves (i.e., $t+1$ where $t \geq 1$ is the number of (re)boots made by the KCS-PRNG such that the generator gets at least two un-used elliptic curve on each (re)boot) in the look-up table consisting of elliptic curves.

6.6 Details of Two Elliptic Curves used in the Proposed KCS-PRNG

Elliptic curves over 256-bit prime fields whose ECDLPs are found to be hard and secure from ECC and trusted security perspectives, are selected for use in the proposed KCS-PRNG. The elliptic curves are generated randomly over the 256-bit prime field size in order to build the trust as indicated in Chapter 4 and Chapter 5 of this thesis. The verification details against the criteria as suggested in [**36**] of the two elliptic curves selected for experimentation purposes in this work are summarized in Chapter 4 of this thesis. There are 256 elliptic curves randomly retrieved from the database of elliptic curves which was created as discussed in Section 4.7.3 to construct the look-up table $\mathcal{T}$ consisting of elliptic curves defined over 256 bit prime field only. Two elliptic curves are then randomly picked up from $\mathcal{T}$ which we name as KG256r2 and KG256r3 for demonstration purpose with the proposed KCS-PRNG in this chapter. The look-up table $\mathcal{T}$ is already discussed in Section 6.3.3.

<This space is intentionally left blank.>

Table 6.5: First elliptic curve (KG256r2) used in the proposed KCS-PRNG

Elliptic curve parameter/Validation	**Value**
Equation Model	Short Weierstrass
Prime field p	107950916448798859175726520127695641701905650240413266754029756463013406356611
Coefficient a	372887183393790501733839885870938677488017443500794637387049768833559724285251
Coefficient b	916504671845195285271956698224008976232881639421184099288854863928706840034 90
Co-factor h	1
Base Point $G_{x,y}$	(288316309299981640447519481483047945182095511258995077033882817311082819373 85, 894203087546989715773049283931158794868926234430116943488818230945595444620 40)

Table 6.6: Second elliptic curve (KG256r3) used in the proposed KCS-PRNG

Elliptic curve parameter/Validation	**Value**
Equation Model	Short Weierstrass
Prime field p	109746855844275773548954457536426580651436369726592499974288558315649948115511
Coefficient a	807875372876919341096326926174458375424612948238742890489089825866611037460 54
Coefficient b	883913684151338226385849966027249087174735330816376473569065796316339345837 32
Co-factor h	1
Base Point $G_{x,y}$	(818187913518001917867298577272592592457045739861832411072535706506900681235 5, 726739824138688932221586625748634204798379752280001956725309493473753995600 02)

6.7 Performance Analysis of the Proposed KCS-PRNG

The proposed KCS-PRNG was run on Intel® Core™ i7-7700 CPU @ 3.60GHz processor. The source code of the KCS-PRNG is developed in C++ and extensively used CryptoPP version 8.2.1 library. The KCS-PRNG software program was run on Ubuntu version 16.04.1 with kernel version 4.15.0-96-generic. The KCS-PRNG program was (re)seeded on every 100000 bits output in generation of 1GB file of cryptographically secure pseudorandom bitstream. It gave an impressive throughput of 2.5 Mbps in software which asserts its high throughput-oriented design. The proposed KCS-PRNG for kernel applications offers a better security by meeting all the RNG requirements from $R1$ to $R4$ as compared to the existing PRNG [**90**] and kernel CSPRNGs like/dev/random [**62**, **63**], Yarrow [**64**], and Fortuna [**65**, **66**].

6.8 Comparison of proposed KCS-PRNG with recent Kernel CSPRNGs and TRNG

The proposed KCS-PRNG is designed to meet all the requirements of a RNG as discussed in Section 1.4.6. The features of the proposed KCS-PRNG are compared with the popular CSPRNGs used by the current operating system kernels and a recently well acknowledged TRNG [**91**] in Table 6.7. The reason behind the comparison of KCS-PRNG with TRNG is that, it meets the RNG requirement $R4$ which a TRNG only meets. Table 6.7 also consolidates interesting comparison results of KCS-PRNG with an existing TRNG based on Oscillator-Rings [**91**].

Table 6.7: Comparison of the proposed KCS-PRNG with recent Kernel CSPRNGs and TRNG

Criterian	**/dev/(u)random**	**Yarrow**	**Fortuna**	**KCS-PRNG**	**TRNG**
Hard problem used	ChaCha20 Stream cipher	3DES	AES128 in counter mode	ECDLP	Physical property of Oscillator-Rings
Hash function	SHA160, MD5 [**107**]	SHA160	SHA256	SHA256	Not applicable
RNG requirements met	*R*1, *R*2, *R*3	*R*1, *R*2, *R*3	*R*1, *R*2, *R*3	*R*1, *R*2, *R*3, *R*4 (Mitigated)	*R*1, *R*2, *R*3, *R*4
Unblocked supply of random bits	No	No	Yes	Yes	Yes
Correlation Test	*	*	*	Passed (serial correlation of 0.000034)	Passed
Per bit entropy rate	*	*	*	0.99999975	0.9993
Linear complexity $LC(x)$	*	*	*	$8830^{19917} < LC(x) \leq 8830^{19918}$	Not applicable
Period	*	*	2^{128} in single call [**66**]	2^{401}	Infinite
Key space	*	*	*	$[2^{529}, \infty)$	Infinite
Throughput	8-12Kbps [**107**]	No results [**107**]	7.2 Mbps	2.5 Mbps	6 Mbps on Xilinx Spartan-3A FPGA
Statistical tests passed	Diehard [**107**]	Not available [**107**]	Diehard [**66**]	NIST, Diehard, TestU01	NIST
NIST proportion obtained	*	*	*	0.9896	0.987
Restart/Non-reproducibility Test	*	*	*	Passed	Passed

*No reference available

The KCS-PRNG is compared with popular kernel CSPRNGs namely /dev/(u)random used by Linux and Android kernels, Yarrow used by MacOS/iOS/FreeBSD kernel and Fortuna used by Windows kernel respectively on the basis of various criteria related to cryptographic security, randomness tests and throughput to conclude their suitability for strategic applicatons such as kernel applications.

The kernel CSPRNGs use design which are based on non-invertible functions which are supposed to be cryptographically hard. Such CSPRNGs are considered to be provably secure for use. /dev/(u)random, Yarrow and Fortuna are the kernel CSPRNGs which uses ChaCha 20 (a secure stream cipher), 3DES and AES128 in counter mode (secure block ciphers) respectively which are the non-invertible functions supposed to be cryptographically hard. However, the TRNG compared in the thesis uses oscillator rings to extract randomness. In contrast, the proposed KCS-PRNG uses ECDLP to provide provable security to its generated bitstreams. The benefits of using ECDLP as hard problem are discussed in Section 6.3.1. Further, the KCS-PRNG uses stronger SHA256 hash function as compared to SHA160/MD5 hash functions used by /dev/(u)random and Yarrow respectively. KCS-PRNG also competes with TRNG in light of non-reproducibility of the generated bitstreams to meet the $R4$ requirement of RNG as discussed in Section 6.3 and Section 6.5.3 for the first time in the literature. Furthermore, it is observed that among all the kernel CSPRNGs, the KCS-PRNG only provides details on most of the important results such as correlation value, per bit entropy rate, linear complexity, period and key space of its generated bitstreams obtained during the security analysis of the generator as discussed in Section 6.4. These results are considered as crucial criteria to measure the theoretical security of the random bitstream generators which are unfortunately not available in public domain to the best of the literature survey conducted in the thesis. The

KCS-PRNG has impressive throughput of 2.5 Mbps, however, it is slower than that of Fortuna as shown in Table 6.7. As discussed in Section 2.8.3, Fortuna uses its pools which contain percomputed hard coded values, at different rate for output bitstreams generation. However, KCS-PRNG does not maintain such pools having precomputed hard coded values for security reasons. Finally, Figure 6.5 pictorically represents the metrics comparison chart of the proposed KCS-PRNG with the popular CSPRNGs and a recent TRNG.

Figure 6.5: Metrics comparison of proposed KCS-PRNG with recent Kernel CSPRNGs and TRNG

6.9 Recent PRNG based Attacks

Klein [**108**], Vice President of security research at SafeBreach and a security researcher at Israel's Bar-Ilan University, discovered a weakness in the Linux Kernel PRNG which allowed the hackers to use cross-layer attacks against the Linux kernel. The Linux kernel PRNG allowed the hackers to get inference about the internal state of the PRNG from one Open Systems Interconnection (OSI) (network) layer and use this internal state to predict the random number value in another OSI layer. This weakness in the PRNG also allowed hackers to identify and track both the Linux and the Android devices. The attack is aimed to downgrade E-mail security, hijack E-mails, hijack HTTP traffic, circumvent E-mail anti-spam and blacklisting mechanisms, mount a local Denial of Service (DoS) attack (blackhole hosts), poison reverse DNS resolutions and attack the machine's Network Time Protocol (NTP) client, responsible for the machine's clock.

It is worth to note that both the Linux and Android based operating systems use the /dev/(u)random random number generator which were exploited due to predictability of their internal states in the above mentioned attack as the attackers were able to predict entire random sequences generated by it. However, the proposed KCS-PRNG does not allow such leakage of its internal states due to non-reproducible random bit sequences generated by it. Hence KCS-PRNG can encounter such attacks completely to serve the kernel applications unhindered.

6.10 Summary

A novel CSPRNG called KCS-PRNG is presented in this chapter which exhibits qualities of a CSPRNG and TRNG for use in cryptography such as securing kernel applications. The combination of clock-controlled LFSRs as a nonlinear sequence generator and two non-standard and trusted elliptic curves is proven

to be an excellent choice to design such a CSPRNG. The KCS-PRNG has successfully validated through all the tests of NIST, Diehard and TestU01 test suites. The NIST test also proved that KCS-PRNG exhibits impressive and the highest proportion i.e., the pass rate of 0.9896 as compared to the existing PRNG [**6**] with 0.9887 and TRNG [**7**] with 0.987 proportion values respectively. The KCS-PRNG demonstrated to exhibit nearly an ideal 0.99999975 per bit entropy and minimal serial correlation of 0.000034 in its generated bitstreams. The KCS-PRNG also showed an impressive throughput of 2.5 Megabits per second. An extensive security analysis of the KCS-PRNG proved that the proposed generator is resistant to important attacks like Berlekamp-Massey attacks, brute force attacks, next-bit tests, state compromise extension attacks and correlation attacks on the proposed generator. In summary, the KCS-PRNG has been proven to exhibit: higher security property (from RNG requirements $R1$ to $R4$), provably secure, very high per bit entropy rate, minimal bitwise correlation, highly nonlinear with linear complexity $LC(x)$ bounded as $15670^{6237} < LC(x) \leq 15670^{6238}$, very large period in the range of $[N_1 \times 2^{401}, (N_1 + N_2) \times 2^{401}]$ per boot where $N_1 < N_2$ being the order of two elliptic curves used, huge key space in the range of $[2^{529}, \infty)$ and impressive throughput to generate uninterrupted cryptographically secure bitstreams. The proposed design of the KCS-PRNG allows periodic change of elliptic curves in the look-up table maintained by the generator to mitigate the gap of the security property $R4$ i.e., 'non-reproducibility' requirement to a practical extent. The use of elliptic curves from its look-up table makes the KCS-PRNG customizable than the current kernel CSPRNGs like /dev/random, Yarrow and Fortuna whose designs are based on stream cipher like ChaCha20 and block ciphers like Triple DES and AES respectively. Hence, it is inferred that the proposed KCS-PRNG qualifies as a competent CSPRNG for adoption in the kernel applications.

Chapter 7

Conclusion and Future Research

The present thesis covered seven important research problems in the applied cryptography domain with respect to Short Weierstrass form of elliptic curves and their implementations in various computer applications of strategic nature such as operating system kernels, in particular. Strategic applications accept only those elliptic curves in its cryptosystem implementations which are transparently computed for trust building as well as whose parameters i.e., the coefficients and prime are rigorously verified for their cryptographic suitability. The thesis first thoroughly evaluated the computational approaches of Short Weierstrass elliptic curves from computation, security and trust persperctives and concluded that the strategic or mission critical applications requires preferably the random approach to compute elliptic curves for cryptosystem design in order to avoid any special structures or pre-studied values which may be vulnerable to unknown (intentionally non-disclosed) attacks. Subsequently, three new *trusted security acceptance criteria* were proposed to derive the curve parameters which can be trusted by its users. Two cryptographically secure Short Weierstrass elliptic curves over 256 bit (called as KG256r1) and 384 bit (called as KG384r1) prime field sizes were proposed which were randomly generated using explicit and well-documented procedures and verified against their cryptographic security. A

database of 500 such similar elliptic curves ove 256 bit prime field size having nearly very high ρ-complexity of 127.8 bit and 191.6 bit each on an ideal 128 bit and 192 bit of symmetric security scales respectively is created for future usage. Further, an important requirement of large prime field order elliptic curves is felt for cryptographic purposes in order to keep existing elliptic curve cryptographic systems alive in presence of quantum capable adversaries where the thesis proposed desired computing resource estimates to compute such elliptic curves. These computing resource estimates was measured in terms of the quantum of CPU clock cycles and searches made in the security parameter space of the elliptic curves. A range of computing estimates of elliptic curve over certain bit of prime field size in terms of number of CPU clock cycels for processing power and number of attempts or searches made is proposed in the thesis against the number of qubits required to solve the ECDLP offered by the elliptic curves defined over that particular field size. The proposed results help user to select a suitable prime field size of the desired elliptic curve which can co-exist with appropriate number of available qubits with quantum computers for a reasonably long period.

Furthermore, these elliptic curves are used in the design of the proposed KCS-PRNG to enable non-reproducibility property of its generated pseudorandom bitstreams for the first time in the literature. The existing CSPRNGs used by operating system kernels do not exhibit the non-reproducibility property of their generated pseudorandom bitstreams till date. Given the properties of the proposed KCS-PRNG, it is inferred that the KCS-PRNG qualifies as a competent CSPRNG for adoption in the kernel applications and can also be used in the implementation of various cryptosystems in deriving their keys and supporting their encryption schemes etc.

7.1 Research Contribution to the Society

The major contributions of the research carried out in this thesis towards the society are as follows:

- Indian Defence Agencies (IDAs) like triforces will be able to compute cryptographically secure and trusted elliptic curves over large prime fields for development of strategic cryptosystems discarding so claimed secure elliptic curve recommended by the international agencies through their (possibly sabotaged) standards.

- The proposed research provides a cost effective solution to counterfeit the technology fallout of the ECC technology in presence of the quantum adversaries to a reasonable extent. The proposed research enables the existing ECC based cryptosystems safe to co-exist in presence of the quantum adversaries by simply replacing the old elliptic curves with the new ones which are defined over a reasonably larger prime field sizes.

- Apart from the Windows and Linux based personal computers and servers, one of the most important and widely used applications for using the proposed KCS-PRNG is the Android/Windows/iOS/MacOS based mobile devices. The KCS-PRNG is proven to counter advanced attacks on operating system kernel used by the mobile devices to prevent them from their data compromise. The KCS-PRNG does not give any scope for leakage of its internal states due to non-reproducibility of its generated pseudo random bitstreams and hence ensures non-predictability of its generated bitstreams.

- Internet of Things (IoT) devices can use proposed trusted elliptic curves KG256r1 and KG384r1 for digital signing and strong authentication purposes.

- The thesis contributions can also be used for strong and trusted authentication services in automobile vehicles such as driverless cars using proposed elliptic curves as well as in their control systems by updating new KCS-PRNG in their operating system kernel for trusted security services.

- The thesis contributions will lead to stronger authentication mechanisms using proposed elliptic curves for sophisticated medical robotic equipments.

- The thesis contributions will be highly useful in ubiquitous computing.

7.2 Future Directions

7.2.1 Future Directions in ECC in Quantum Presence

Elliptic curve cryptosystems are safe in forthcoming few years until the quantum hardware with reasonable qubits is built which may use Shor's algorithm [**29**] to break ECDLP in polynomial time. Chen et. al. [**38**] recently observed that isogenies of elliptic curves of supersingular class which have non-abelian structures seem to be more challenging problem than ECDLP of Short Weierstrass elliptic curves against quantum attacks. Though supersingular elliptic curves are not standardized so far to the best of the authors' knowledge, the next generation may prefer to select supersingular elliptic curves instead of Short Weierstrass elliptic curves to resist quantum attacks in particular.

<This space is intentionally left blank.>

7.2.2 Open Problems for Future Work

The thesis presents the following three open problems for future research:

Problem 1

> Estimation of computational resources in terms of the number of CPU clock cycles and the number of searches to be made in the security parameter space of elliptic curves over binary fields.

Problem 2

> Standardization of the elliptic curves over 256 bit and 384 bit or even higher prime field sizes using the proposed trusted security acceptance criterion in addition to ECDLP security and ECC security criteria.

Problem 3

> Research and Development of concrete mechanism to obtain initial entropy for the proposed KCS-PRNG.

●●●●●

Bibliography

Bibliography

[1] Koblitz, Neal, Alfred Menezes, and Scott Vanstone. "The state of elliptic curve cryptography." Designs, codes and cryptography 19, no. 2 (2000): 173-193.

[2] Hankerson, Darrel, Alfred J. Menezes, and Scott Vanstone. "Guide to elliptic curve cryptography". Springer Science & Business Media, 2006.

[3] Bos, Joppe W., Craig Costello, Patrick Longa, and Michael Naehrig. "Selecting elliptic curves for cryptography: An efficiency and security analysis." Journal of Cryptographic Engineering 6, no. 4 (2016): 259-286.

[4] Menezes, Alfred J., Tatsuaki Okamoto, and Scott A. Vanstone. "Reducing elliptic curve logarithms to logarithms in a finite field." IEEE Transactions on information Theory 39, no. 5 (1993): 1639-1646.

[5] Frey, Gerhard, and Hans-Georg Ruck. "A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves." Mathematics of computation 62, no. 206 (1994): 865-874.

[6] Koblitz, Ann Hibner, Neal Koblitz, and Alfred Menezes. "Elliptic curve cryptography: The serpentine course of a paradigm shift." Journal of Number theory 131, no. 5 (2011): 781-814.

[7] Smart, Nigel P. "The discrete logarithm problem on elliptic curves of trace one." Journal of Cryptology 12, no. 3 (1999): 193-196.

[8] Boneh, Dan, Ben Lynn, and Hovav Shacham. "Short signatures from the Weil pairing." In International conference on the theory and application of cryptology and information security, pp. 514-532. Springer, Berlin, Heidelberg, 2001.

[9] Menezes, Alfred. "Evaluation of security level of cryptography: the Elliptic Curve Discrete Logarithm Problem (ECDLP)." University of Waterloo (2001).

[10] Flori, Jean-Pierre, Jérôme Plût, Jean-Rene Reinhard, and Martin Ekerå. "Diversity and Transparency for ECC." IACR Cryptol. ePrint Arch. 2015 (2015): 659.

[11] Washington, Lawrence C. "Elliptic curves: number theory and cryptography". Second Edition, CRC press, 2008.

[12] Koblitz, Neal. "Constructing elliptic curve cryptosystems in characteristic 2." In Conference on the Theory and Application of Cryptography, pp. 156-167. Springer, Berlin, Heidelberg, 1990.

[13] Cohen, Henri, Gerhard Frey, Roberto Avanzi, Christophe Doche, Tanja Lange, Kim Nguyen, and Frederik Vercauteren, eds. "Handbook of elliptic and hyperelliptic curve cryptography". CRC press, 2005.

[14] Kraft, James S., and Lawrence C. Washington. "An introduction to number theory with cryptography". Chapman and Hall/CRC, 2018.

[15] Schoof, René. "Counting points on elliptic curves over finite fields." Journal de théorie des nombres de Bordeaux 7, No. 1 (1995): 219-254.

[16] Blake, Ian, Gerald Seroussi, Gadiel Seroussi, and Nigel Smart. "Elliptic curves in cryptography". Vol. 265. Cambridge university press, 1999.

[17] Cohen, Henri, Henry Cohen, and Henri Cohen. "A course in computational algebraic number theory". Vol. 8. Berlin: Springer-Verlag, 1993.

[18] Blake, Ian F., Gadiel Seroussi, and Nigel P. Smart, eds. "Advances in elliptic curve cryptography". Vol. 317. Cambridge University Press, 2005.

[19] Schoof, René. "Elliptic curves over finite fields and the computation of square roots mod p." Mathematics of computation 44, no. 170 (1985): 483-494.

[20] Fouquet, Mireille, Pierrick Gaudry, and Robert Harley. "An extension of Satoh's algorithm and its implementation." Journal of the Ramanujan Mathematical Society 15 (2000): 281-318.

[21] Satoh, Takakazu. "On p-adic point counting algorithms for elliptic curves over finite fields." In International Algorithmic Number Theory Symposium, pp. 43-66. Springer, Berlin, Heidelberg, 2002.

[22] Koç, Çetin Kaya. "About cryptographic engineering." In Cryptographic engineering, pp. 1-4. Springer, Boston, MA, 2009.

[23] Forouzan, Behrouz A., and Debdeep Mukhopadhyay. "Cryptography and network security" (Sie). McGraw-Hill Education, 2011.

[24] Schneier, Bruce. "Applied cryptography: protocols, algorithms, and source code in C". John Wiley & Sons, 2007.

[25] Marco-Gisbert, Hector, and Ismael Ripoll Ripoll. "Address space layout randomization next generation." Applied Sciences 9, no. 14 (2019): 2928.

[26] Koblitz, Neal. "A course in number theory and cryptography". Vol. 114. Springer Science & Business Media, 1994.

[27] Caelli, William J., Edward P. Dawson, and Scott A. Rea. "PKI, elliptic curve cryptography, and digital signatures." Computers & Security 18, no. 1 (1999): 47-66.

[28] Valenta, Luke, Nick Sullivan, Antonio Sanso, and Nadia Heninger. "In search of CurveSwap: Measuring elliptic curve implementations in the wild." In 2018 IEEE European Symposium on Security and Privacy (EuroS&P), pp. 384-398. IEEE, 2018.

[29] Shor, Peter W. "Algorithms for quantum computation: discrete logarithms and factoring." In Proceedings 35th annual symposium on foundations of computer science, pp. 124-134. Ieee, 1994.

[30] Balasubramanian, Ramachandran, and Neal Koblitz. "The improbability that an elliptic curve has subexponential discrete log problem under the Menezes—Okamoto—Vanstone algorithm." Journal of cryptology 11, No. 2 (1998): 141-145.

[31] Gaudry, Pierrick, Florian Hess, and Nigel P. Smart. "Constructive and destructive facets of Weil descent on elliptic curves." Journal of Cryptology 15, No. 1 (2002): 19-46.

[32] Kerry, Cameron F., and Patrick D. Gallagher. "Digital signature standard (DSS)." FIPS PUB (2013): 186-4.

[33] SEC, SECG. "2: Recommended elliptic curve domain parameters." Standards for Efficient Cryptography Group, Certicom Corp (2000).

[34] Brainpool, E. C. C. "ECC Brainpool standard curves and curve generation." (2005): 106-109.

[35] Lochter, M., and J. Mekle. "RFC 5639: ECC Brainpool Standard Curves & Curve Generation." Internet Engineering Task Force (2010).

[36] Bernstein, Daniel J., and Tanja Lange. "SafeCurves: choosing safe curves for elliptic curve cryptography, 2015." URL: https://safecurves.cr.yp.to. (2014).

[37] Liu, Zhe, and Hwajeong Seo. "IoT-NUMS: evaluating NUMS elliptic curve cryptography for IoT platforms." IEEE Transactions on Information Forensics and Security 14, no. 3 (2018): 720-729.

[38] Chen, Lily, Lily Chen, Stephen Jordan, Yi-Kai Liu, Dustin Moody, Rene Peralta, Ray Perlner, and Daniel Smith-Tone. "Report on post-quantum cryptography". Vol. 12. US Department of Commerce, National Institute of Standards and Technology, 2016.

[39] Roetteler, Martin, Michael Naehrig, Krysta M. Svore, and Kristin Lauter. "Quantum resource estimates for computing elliptic curve discrete logarithms." In International Conference on the Theory and Application of Cryptology and Information Security, pp. 241-270. Springer, Cham, 2017.

[40] Roetteler, Martin, Kristin Lauter, and Krysta Svore. "Quantum resource estimates for computing elliptic curve discrete logarithms." U.S. Patent 10,430,162, issued October 1, 2019.

[41] Pohlig, Stephen, and Martin Hellman. "An improved algorithm for computing logarithms over GF (p) and its cryptographic significance (corresp.)." IEEE Transactions on information Theory 24, no. 1 (1978): 106-110.

[42] Van Oorschot, Paul C., and Michael J. Wiener. "Parallel collision search with cryptanalytic applications." Journal of cryptology 12, no. 1 (1999): 1-28.

[43] Semaev, Igor. "Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p." Mathematics of computation 67, no. 221 (1998): 353-356.

[44] Baier, Harald, and Johannes Buchmann. "Generation methods of elliptic curves." An evaluation report for theInformation-technology Promotion Agency, Japan (2002).

[45] Morain, François. "Building cyclic elliptic curves modulo large primes." In Workshop on the Theory and Application of of Cryptographic Techniques, pp. 328-336. Springer, Berlin, Heidelberg, 1991.

[46] Konstantinou, E., A. Kontogeorgis, Y. C. Stamatiou, et al. – J Cryptol., Vol. 23, 2010, 477. https://doi.org/10.1007/s00145-009-9037-2

[47] Konstantinou, Elisavet, Aristides Kontogeorgis, Yannis C. Stamatiou, and Christos Zaroliagis. "On the efficient generation of prime-order elliptic curves." Journal of cryptology 23, No. 3 (2010): 477-503.

[48] Savaş, Erkay, Thomas A. Schmidt, and Cetin K. Koç. "Generating elliptic curves of prime order." In International Workshop on Cryptographic Hardware and Embedded Systems, pp. 142-158. Springer, Berlin, Heidelberg, 2001.

[49] Vo, San C. "A Survey of Elliptic Curve Cryptosystems, Part I: Introductory." NASA Advanced Supercomputing Division; NAS Technical Report—NAS-03-012 (2003).

[50] Costello, Craig, Patrick Longa, and Michael Naehrig. "A brief discussion on selecting new elliptic curves." Microsoft Research. Microsoft 8 (2015).

[51] Bernstein, Daniel J., Tung Chou, Chitchanok Chuengsatiansup, Andreas Hülsing, Eran Lambooij, Tanja Lange, Ruben Niederhagen, and Christine Van Vredendaal. "How to Manipulate Curve Standards: A White Paper for the Black Hat http://bada55. cr. yp. to." In International Conference on Research in Security Standardisation, pp. 109-139. Springer, Cham, 2015.

[52] https://www.schneier.com/blog/archives/2013/09/the_nsa_is_brea.html

[53] https://www.nist.gov/system/files/documents/2017/05/09/2014-VCAT-Annual-Report_final.pdf

[54] https://cryptosith.org/michael/data/talks/2015-04-28-UWNumberTheorySeminar.pdf

[55] Scott, M. Re: NIST Announces Set of Elliptic Curves. Posting to the sci.crypt Mailing List, 1999.

[56] Hamburg, Mike. "Ed448-Goldilocks, a new elliptic curve." IACR Cryptol. ePrint Arch. 2015 (2015): 625.

[57] Shamir, A. "Method and apparatus for protecting public key schemes from timing and fault attacks." In EUROCRYPT'97. 1997.

[58] Dąbrowski, Przemysław, Rafał Gliwa, Janusz Szmidt, and Robert Wicik. "Generation and Implementation of Cryptographically Strong Elliptic Curves." In International Conference on Number-Theoretic Methods in Cryptology, pp. 25-36. Springer, Cham, 2017.

[59] Alekseev, Evgeny Konstantinovich, V. D. Nikolaev, and Stanislav Vital'evich Smyshlyaev. "On the security properties of Russian standardized elliptic curves." 2018.

[60] Proos, John, and Christof Zalka. "Shor's discrete logarithm quantum algorithm for elliptic curves." arXiv preprint quant-ph/0301141 (2003).

[61] Wohlwend, Jeremy. "Elliptic curve cryptography: Pre and post quantum". Technical report, 2016.

[62] Viega, John. "Practical random number generation in software." In 19th Annual Computer Security Applications Conference, 2003. Proceedings., pp. 129-140. IEEE, 2003.

[63] Dodis, Yevgeniy, David Pointcheval, Sylvain Ruhault, Damien Vergniaud, and Daniel Wichs. "Security analysis of pseudo-random number generators with input: /dev/random is not robust." In Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security, pp. 647-658. 2013.

[64] Kelsey, John, Bruce Schneier, and Niels Ferguson. "Yarrow-160: Notes on the design and analysis of the yarrow cryptographic pseudorandom number generator." In International Workshop on Selected Areas in Cryptography, pp. 13-33. Springer, Berlin, Heidelberg, 1999.

[65] Ferguson, Niels, Bruce Schneier, and Tadayoshi Kohno. "Cryptography engineering: design principles and practical applications". John Wiley & Sons, 2011.

[66] McEvoy, Robert, James Curran, Paul Cotter, and Colin Murphy. "Fortuna: cryptographically secure pseudo-random number generation in software and

hardware." In 2006 IET Irish Signals and Systems Conference, pp. 457-462. IET, 2006.

[67] Dodis, Yevgeniy, Adi Shamir, Noah Stephens-Davidowitz, and Daniel Wichs. "How to eat your entropy and have it too: Optimal recovery strategies for compromised RNGs." Algorithmica 79, no. 4 (2017): 1196-1232.

[68] Jablon, David. "IEEE P1363 standard specifications for public-key cryptography." In CTO Phoenix Technologies Treasurer, IEEE P1363 NIST Key Management Workshop. 2001.

[69] Kerckhoffs, Auguste. "La cryptographic militaire." Journal des sciences militaires (1883): 5-38.

[70] Yin, Edward. "Curve selection in elliptic curve cryptography." San Jose State University, Project, Spring (2005).

[71] https://rump2007.cr.yp.to/15-shumow.pdf

[72] Hales, T. C. "The NSA Back Door to NIST". Notices of the AMS, Vol. 61, 2013, No 2, pp. 190-192.

[73] Bernstein, D. J., T. Lange. "Security Dangers of the NIST Curves". In: Invited Talk, International State of the Art Cryptography Workshop, Athens, Greece, 2013.

[74] Agence nationale de la s´ecurit´e des syst'emes d'information. "Publication d'un param´etrage de courbe elliptique visant des applications de passeport ´electronique et de l'administration ´electronique fran¸caise", 2011. https://tinyurl.com/nhog26h

[75] Cheng, Qi. "Hard problems of algebraic geometry codes." IEEE Transactions on Information Theory 54, no. 1 (2008): 402-406.

[76] McIvor, Ciaran J., Maire McLoone, and John V. McCanny. "Hardware Elliptic Curve Cryptographic Processor Over $rmGF(p)$." IEEE Transactions on Circuits and Systems I: Regular Papers 53, No. 9 (2006): 1946-1957.

[77] Gutterman, Zvi, Benny Pinkas, and Tzachy Reinman. "Analysis of the linux random number generator." In 2006 IEEE Symposium on Security and Privacy (S&P'06), pp. 15-pp. IEEE, 2006.

[78] https://www.2uo.de/myths-about-urandom/

[79] Aumasson, Jean-Philippe. "Serious cryptography: a practical introduction to modern encryption". No Starch Press, 2017.

[80] Abd-El-Barr, Mostafa. "Fundamentals of computer organization and architecture". John Wiley and Sons, 2005.

[81] R. P. Brent. "Recent progress and prospects for integer factorisation algorithms". In International Computing and Combinatorics Conference, Springer, Berlin, Heidelberg, pages 3-22, 2000.
https://doi.org/10.1007/3-540-44968-X_2

[82] T. Satoh, and K. Araki. "Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves". Rikkyo Daigaku sugaku zasshi, 47(1): 81-92, 1998.

[83] V. G.Martinez, L. Gonzalez-Manzano, and A. M.Munoz. "Secure Elliptic Curves in Cryptography". In Computer andNetwork Security Essentials, Springer, Cham, pages 283-298, 2018.
https://doi.org/10.1007/978-3-319-58424-9_16

[84] T. C. Urdan. "Statistics in plain English". Routledge, 2011.
https://doi.org/10.4324/9780203851173

[85] J. Gareth. "An introduction to statistical learning: with applications in R". Springer Verlag, 2010

[86] S.Weisberg. "Applied linear regression" (Vol. 528). Third Edition. John Wiley & Sons, 2005. https://doi.org/10.1002/0471704091

[87] S. C. Gupta and V. K. Kapoor. "Fundamentals of Mathematical Statistics (A Modern Approach)". S. Chand & Sons, New Delhi, 2000.

[88] Tanenbaum AS, Woodhull AS. "Operating systems: design and implementation". Third Edition, Englewood Cliffs: Prentice Hall;ISBN 978-93-325-5051-3; 2018.

[89] D¨orre F, Klebanov V. "Pseudo-random number generator verification: A case study". InVSSTE 2015 Jul 18 (pp. 61-72). Springer, Cham.

[90] Alhadawi HS, Zolkipli MF, Ismail SM, Lambi´c D. "Designing a pseudorandom bit generator based on LFSRs and a discrete chaotic map". Cryptologia. 2019 May 4;43(3):190-211.

[91] Anandakumar NN, Sanadhya SK, Hashmi MS. "FPGA-based true random number generation using programmable delays in oscillator-rings". IEEE Transactions on Circuits and Systems II: Express Briefs. 2019 May 30;67(3):570-4.

[92] Gong G, Lam CC. "Linear recursive sequences over elliptic curves". In Sequences and their Applications 2002 (pp. 182-196). Springer, London.

[93] Mukhopadhyay S, Sarkar P. "Application of LFSRs for parallel sequence generation in cryptologic algorithms". In International Conference on Computational Science and Its Applications 2006 May 8 (pp. 436-445). Springer, Berlin, Heidelberg.

[94] Rajski J, Tyszer J. "Primitive polynomials over GF (2) of degree up to 660 with uniformly distributed coefficients". Journal of Electronic testing. 2003 Dec 1;19(6):645-57.

[95] Menezes AJ. PC v. Oorschot, and SA Vanstone, "Handbook of Applied Cryptography", fifth printing ed.

[96] Teo SG. "Analysis of nonlinear sequences and stream ciphers" (Doctoral dissertation, Queensland University of Technology).

[97] L'Ecuyer P, Simard R. "TestU01: AC library for empirical testing of random number generators". ACM Transactions on Mathematical Software (TOMS). 2007 Aug 15;33(4):1-40.

[98] L'Ecuyer P, Simard R. "TestU01: A Software Library in ANSI C for Empirical Testing of Random Number Generators–User's Guide", Compact Version. 2013.

[99] Walker J. "ENT: a pseudorandom number sequence test program. Software and documentation". Available at/www.fourmilab.ch/random/S. 2008 Jan.

[100] https://www.ecrypt.eu.org/ecrypt2/documents/D.SPA.13.pdf

[101] Bassham III LE, Rukhin AL, Soto J, Nechvatal JR, Smid ME, Barker EB, Leigh SD, Levenson M, Vangel M, Banks DL, Heckert NA. Sp 800-22 rev. 1a. "A statistical test suite for random and pseudorandom number generators for cryptographic applications". National Institute of Standards & Technology; 2010.

[102] Marsaglia G. DIEHARD Test suite.
Online: http://www.stat.fsu.edu/pub/diehard. 1998;8(01):2014.

[103] Bhattacharjee K, Maity K, Das S. "A Search for Good Pseudo-random Number Generators: Survey and Empirical Studies". arXiv preprint arXiv:1811.04035. 2018 Nov 3.

[104] Lavasani A, Eghlidos T. "Practical next bit test for evaluating pseudorandom sequences". 2009 (pp 19-33).

[105] Rose GG, Gantman A, Xiao L, inventors; Qualcomm Inc, assignee. "Cryptographically secure pseudo-random number generator". United States patent US 8,019,802. 2011 Sep 13.

[106] Kelsey J, Schneier B, Wagner D, Hall C. "Cryptanalytic attacks on pseudorandom number generators". InInternational workshop on fast software encryption 1998 Mar 23 (pp. 168-188).Springer, Berlin, Heidelberg.

[107] R¨ock A. "Pseudorandom number generators for cryptographic applications". na; 2005.

[108] Klein, Amit. "Cross layer attacks and how to use them (for dns cache poisoning, device tracking and more)". In 2021 IEEE Symposium on Security and Privacy (SP), pp. 1179-1196. IEEE, 2021.

Appendix: Published Articles

BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **21**, No 4
Sofia • 2021 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2021-0045

Evaluation of Computational Approaches of Short Weierstrass Elliptic Curves for Cryptography

Kunal Abhishek[1], *E. George Dharma Prakash Raj*[2]

[1]*Society for Electronic Transactions and Security, Chennai, India*
[2]*Bharathidasan University, Tiruchirappalli, India*
E-mails: *kunalabh@gmail.com georgeprakashraj@yahoo.com*

Abstract**:** *The survey presents the evolution of Short Weierstrass elliptic curves after their introduction in cryptography. Subsequently, this evolution resulted in the establishment of present elliptic curve computational standards. We discuss the chronology of attacks on Elliptic Curve Discrete Logarithm Problem* (*ECDLP*) *and investigate their countermeasures to highlight the evolved selection criteria of cryptographically safe elliptic curves. Further, two popular deterministic and random approaches for selection of Short Weierstrass elliptic curve for cryptography are evaluated from computational, security and trust perspectives and a trend in existent computational standards is demonstrated. Finally, standard and non-standard elliptic curves are analysed to add a new insight into their usability. There is no such survey conducted in past to the best of our knowledge.*

Keywords**:** *Computational approaches, evaluation, cryptography, elliptic curve, ECDLP, security.*

1. Introduction

Computation of elliptic curve requires extensive mathematical research to compute curve's parameters over large prime field for its use in cryptography [1]. There are several agencies like National Institute of Standards and Technology (NIST), Standards for Efficient Cryptography Group (SECG), Brainpool, etc., who have recommended standard elliptic curves over various prime field orders. However, it is important to note the rationale behind the approaches adopted for selection of elliptic curve parameters from computational, security and trust perspectives. The scope of this article is limited to the Short Weierstrass form of elliptic curves which are used for constructing most of the present cryptosystems such as Public Key Infrastructure (PKI) [2], Secure SHell (SSH), Transport Layer Security (TLS), IPSec, JSON Web Encryption (JWE) [3], etc.

The key contributions of this paper enlist:

1. A comprehensive survey for evaluation of the computational approaches of cryptographically secure elliptic curves is presented.

2. Evolution of Elliptic Curve Cryptography (ECC) with theoretical advancements in cryptographic mathematics and their significant impact on standardization of computational methods is presented.

3. Chronology of attacks on Elliptic Curve Discrete Logarithm Problem (ECDLP) and their countermeasures is presented.

4. Selection criteria of cryptographically secure elliptic curves are discussed.

5. A trend in computational approaches of elliptic curves in standards recommended by various agencies is demonstrated.

6. Standard and non-standard elliptic curves are compared from computational, trust and security perspectives to add a new insight into their usability.

Rest of the paper is organized as follows: Section 2 gives preliminaries on elliptic curves in Short Weierstrass form and ECDLP. Section 3 describes evolution of ECC with time and theoretical advancements in applied mathematics to establish present computational standards and selection criteria of elliptic curve. Section 4 focuses on evaluation of two popular approaches to compute cryptographically secure elliptic curves. Section 5 demonstrates the trend of approaches for computation of elliptic curve parameters adopted by various agencies in their proposed standards. Section 6 differentiates between standard and non-standard elliptic curves in various contexts. Finally, Section 7 concludes the paper with future directions.

2. Preliminaries

2.1. Elliptic curve in short weierstrass form

Let the finite field $\mathbb{F}_q$ has characteristic greater than 3. An elliptic curve $\mathbb{E}$ over $\mathbb{F}_q$ is the set of all solutions (x, y) to an equation

(1) $$\mathbb{E}: y^2 = x^3 + ax + b,$$

where the coefficients $a, b \in \mathbb{F}_q$ and $4a^3 + 27b^2 \neq 0$, together with a special point ∞ called the point at infinity which serves as the identity element of $\mathbb{E}$ which is known to be an abelian group [4].

2.2. The elliptic curve discrete logarithm problem

Definition 1 (ECDLP). Given an elliptic curve $\mathbb{E}$ defined over a finite field $\mathbb{F}_q$, a point $P \in \mathbb{E}(\mathbb{F}_q)$ of order n, and a point $Q \in \langle P \rangle$, determine the integer $l \in [0, n-1]$ such that

(2) $$Q = lP.$$

The integer l is called the discrete logarithm of Q to the base P, denoted as $l = \log_p Q$ [5].

The definitions: Definition 2 [6], Definition 3 [7], Definition 4 [8] and Definition 5 [9] define supersingular curve, embedding degree, prime field anomalous curve and class number of elliptic curves respectively which need to be carefully considered for selection of elliptic curves with intractable ECDLP for cryptography.

Definition 2 (Supersingular Elliptic Curves). If $\#\mathbb{E}(\mathbb{F}_q) = q + 1 - t$ denote the order of elliptic curve then $\mathbb{E}(\mathbb{F}_q)$ is said to be supersingular if p divides t where p be the characteristic of $\mathbb{F}_q$ and t be the trace of $\mathbb{E}$.

$\mathbb{E}(\mathbb{F}_q)$ is supersingular provided the trace (t) of the curve, $t^2 = 0; q; 2q; 3q$ or $4q$ [6]. Supersingular elliptic curves are vulnerable to attack due to Menezes, Okamoto and Vanstone (MOV) which solves Discrete Logarithm Problem (DLP) of

supersingular curves to the DLP in a finite field with sub-exponential complexity [6, 10].

Definition 3 (Embedding Degree of Elliptic Curve). If $\mathbb{E}(\mathbb{F}_q)$ be the elliptic curve over $\mathbb{F}_q$ then $\mathbb{E}$ is said to have embedding degree k, a smallest positive integer, such that $n \mid (q^k - 1)$ where n be the base point order.

It is also observed that ECC standards do not allow elliptic curves with low embedding degrees.

Definition 4 (Prime Field Anomalous Curves). An elliptic curve $\mathbb{E}$ defined over a prime field $\mathbb{F}_p$ is said to be prime field anomalous if $\#\mathbb{E}(\mathbb{F}_p)=p$, i.e., the curve has trace 1.

Prime field anomalous curves are trace one curves for which the ECDLP can be solved in linear time [10]. The prime field anomalous attack does not extend to any other classes of elliptic curves but the one having trace one [8].

Definition 5 (Class Number). Let $h(N)$ denotes the class number of the order N of elliptic curve $\mathbb{E}$. Then $h(N)$ is the minimum degree of a number field over which the elliptic curve $\mathbb{E}$ admits a faithful lift.

3. Evolution of elliptic curves for cryptography

Table 1. Evolution of Short Weierstrass elliptic curves for cryptography

Year	Event	Impact on ECC Standardization
1985	Elliptic curves were proposed for use in cryptography	ECC were extensively studied to develop cryptosystems
1987	Efficient point counting algorithm on elliptic curves by Schoof, Elkies and Atkin called SEA Algorithm was developed [17-18]	Uses complexity $O(\ln^5 p)$ for point counting
1992	Elliptic Curve based Digital Signature Algorithm (ECDSA) was developed [19]	Considered as a mature signature scheme in NIST standard
1993	Reduction of ECDLP of supersingular elliptic curves having trace zero to logarithm in a finite field [6]	Became selection criteria for safe elliptic curve in all standards
1994	Proposal of Shor algorithm [20] generalizes to solve ECDLP Random Quantum Polynomial (RQP) time using quantum computers	Led to realization that elliptic curves will be unsafe once sufficient quantum capability is built. So, new computational standard required for quantum resistance
1996	It was proved that the condition $N \mid (q^k - 1)$ is sufficient to realize the MOV algorithm under mild condition. Further, it was proved that randomly generated curves have $k > \log^2 q$ [21]	Became selection criteria for safe elliptic curve in all standards
1997	Proposal of a linear algorithm to solve ECDLP of trace one [10, 22]	Became selection criteria for safe elliptic curve in all standards
1999	NIST recommendation of 15 elliptic curves [23]	Widely accepted standard later
2000	SECG recommendation of elliptic curves [24]	Widely accepted standard later
2005	Recommendation of Brainpool first set of elliptic curves for standardization [25]	International effort for elliptic curve standardization
2010	Brainpool revised their specifications and published Request for Comment (RFC) 5639 [26]	Standard established
2014	Review of existing elliptic curves generation mechanisms by Bernstein and Lange [27] who coined two terms: ECDLP security and ECC security. They observed that Short Weierstrass form of elliptic curves are dominant in both the software and hardware implementations	Two new terms: ECDLP security and ECC security became important verification criteria for curve selection with side channel attack resistance
2014	NUMS-curve (Nothing Upon My Sleeves) were proposed under IETF standard [28]	Curves with better performance proposed under IETF Standard
2015	NIST Call for next generation elliptic curves with new models and optimized parameters resistant to side channel analysis was placed [28]	NIST wanted to replace its standard elliptic curves
2016	NIST report [29] on Post Quantum Cryptography (PQC). Resistance of elliptic curve cryptosystems was looked for quantum computing	Isogenies of supersingular elliptic curves were discussed as resistant to PQC instead of ECDLP
2017-2020	Proposal of Quantum resources required to run Shor algorithm to solve ECDLP in polynomial time [30]	Roeteller et. al. suggested quantum resource estimates to break ECDLP

Note: N=Order of elliptic curve, q=prime power, k=embedding degree.

Table 2. Chronology of attacks on ECDLP and their countermeasures

Attack	Description type	Countermeasure type
Pohlig-Hellman, DLP attack	Private key can be recovered using Chinese Remainder Theorem [31]	N must be a prime or near prime with small cofactor, $N \geq 2^{160}$ [5]
Pollard-rho, DLP attack	A parallelized Pollard-rho on r processors can solve ECDLP in $\sqrt{(\pi n)}/\sqrt{(2r)}$ steps [5, 32]	$n \geq 2^{160}$ [13, 32]
Pollard's Lambda, DLP attack	Faster method than Pollard-rho when ECDLP lies in subinterval $[1, b]$ of $[1, n-1]$, where $b < 0.39n$ [13]	Private key should be selected uniformly at random within interval $[1, n-1]$ [30]
Index-Calculus, DLP attack	ECDLP can be solved using multiplicative group $\mathbb{F}_q^*$ of the finite field $\mathbb{F}_q$ [13]	Small prime fields should be avoided, i.e., $n \geq 2^{160}$ [13]
Exhaustive Search, DLP attack	Computes successive multiples of base point till public key is achieved	n should be sufficiently large [8]
Shanks' Baby step Giant step, DLP attack	Fully exponential deterministic algorithm to determine n on $\mathbb{E}(\mathbb{F}_q)$ which requires approximately $\sqrt{N}$ steps and around $\sqrt{N}$ storage	$n \geq 2^{160}$ [13]
Weil pairing and Tate pairing attacks, Pairing based attack	ECDLP of $\mathbb{E}(\mathbb{F}_q)$ can be reduced to ordinary DLP on extension field $\mathbb{F}^*_{q^k}$ for some $k \geq 1$ where the number field sieve algorithm can be used to solve ECDLP [4, 6]. MOV reduction attack [6]	$n \nmid (q^k - 1)\ \forall k \geq 20$ [7, 18] and $\forall k \geq (q-1)/100$ [5] $p \nmid t$ and $t^2 \neq 0, 2q, 3q$ or $4q$ [6] (Non-supersingularity)
Multiple logarithm, DLP attack	Multiple instances of ECDLP for the same elliptic curve parameters	$n \geq 2^{160}$
Prime field anomalous curve, Pairing based attack	Trace of $\mathbb{E}(\mathbb{F}_p)=1$, i.e., $\#\,\mathbb{E}(\mathbb{F}_p)=p$ [8, 12]	$N \neq p$ [5]

Note: q=size of underlying field, p=prime characteristic, n=order of a point on $\mathbb{E}$, N=order of $\mathbb{E}$, r=number of processors, k=embedding degree, t=trace of curve.

Table 3. Elliptic curve parameters selection criteria

Elliptic curve parameter	Criteria	Benefit(s)
Prime p	1. Crandall prime $2^\alpha - \gamma$ where $\gamma < 2^{10}$ [33, 34] 2.Montgomery-friendly prime $2^\alpha(2^\beta - \gamma) - 1$ where $\alpha, \beta, \gamma \geq 0$ 3. $p \equiv 3 \bmod 4$ 4. Mersenne prime $p = 2^k - 1$ 5. p= random value 6. Length of $p \geq 221$ bits [27]	1. For best possible performance by limiting carry propagation during multiply-reduce and γ is small [34] 2. Accelerates Montgomery arithmetic [33] 3. Such primes can compute modular square root in constant time countering constant time attack using Side channels [33]. The point compression method allows representing one point (x, y) of $\mathbb{E}$ only its abscissa x and one bit discriminating between the two possible values $\pm y$. However, recovering y requires computing a square root in $\mathbb{F}_p$. This is easier when $p \equiv 3 \bmod 4$ since in this case, $c^{(p+1)/2}$ is a square root of c if c is a square [9] 4. Mersenne primes are special primes of unique form which enables fast arithmetic [33] Minimizes time for modular multiplication [35] 5. No pre-studied value or special structure vulnerable to cryptanalysis 6. To counter brute-force attack
Coefficient a	1. $a = -3$ 2. a= random value	1. For efficiency reasons. Practically all curves have low-degree isogenies to curves with $a = -3$, so this choice does not affect security. P1363 allows $y^2 = x^3 + ax + b$ without the requirement $a = -3$ [9] 2. No pre-studied value or special structure
Coefficient b	1. Should not be square in $\mathbb{F}_p$ [9] 2. b=random value	1. To avoid compressed representations of elliptic curve points as $(0, 0)$ and $(0, x)$ would be identical as $x = \sqrt{b}$ with least significant bit as 0 [26] 2. No pre-studied value or special structure
Elliptic curve order N	1. N should be prime [13, 18] 2. N should be composite	1. Prime order curve selected to resist Pohlig-Hellman and Pollard's Rho attacks [5, 9]. Small subgroup attacks are avoided [9, 13] 2. Prime group order curves do not have points with $y = 0$ [36]. Special points of the form $(x, 0)$ exist if the curve has an even order [9]
Base point order n	n should be prime to avoid Weil and Tete pairing attacks [5, 9]	$n \geq 2^{160}$ and $n \nmid (q^k - 1)$ where k is the embedding degree of elliptic curve
Cofactor h	Preferably 1	For optimal bit security, $h = 1$ though $1 \leq h \leq 4$ for performance gain [5, 9, 36]
Base point $G_{x,y}$	Randomly chosen base point [4]	Prime order of base point gives maximum elliptic curve group size

Elliptic curves have been extensively studied and reviewed for cryptography soon after the proposals of Neal Koblitz and Victor Miller during 1985-1987. ECC has evolved with time and theoretical advancements in cryptographic mathematics, which subsequently has significant impacts on evolution of elliptic curve computational standards, which is discussed in Table 1. Moreover, elliptic curves are expected to be resistant to cryptographic attacks that can be ensured through the implementation of appropriate countermeasures. Table 2 [8] briefly depicts such countermeasures for important discrete logarithm (DLP) based attacks and pairing based attacks which resulted in the evolution of cryptographically safe elliptic curve selection criteria. Table 3 shows important selection criteria for elliptic curve parameters and their benefits to select elliptic curves with desired properties.

4. Evaluation of computational approaches

Elliptic curves need to qualify certain mathematical validations in order to certify that the elliptic curve has the claimed order, resists all known attacks on ECDLP and base point order has also the claimed order [5]. There are usually two approaches either of which can be used to compute an elliptic curve over prime field: first, the deterministic approach and second, the random approach. However, in both – the deterministic and random approaches, following conditions are critical for the elliptic curve to meet cryptographic requirements [4, 5, 11]:

C1: Resistance to Pohlig-Hellman and Pollard's Rho attack, i.e., $n > 2^L$ where n is sufficiently large prime that divides order of the elliptic curve group $\#\mathbb{E}(\mathbb{F}_q)$. Here, $L \geq 160$, the length in bits.

C2: Resistance to Semaev-Smart-Satoh-Araki attack (Smart-ASS) [10, 12], i.e., $L \leq \lfloor \log_2 q \rfloor$ ensures $2^L \leq q$ or $\#\mathbb{E}(\mathbb{F}_q) \neq q$. It avoids the attack on prime field anomalous curves.

C3: $n > 4\sqrt{q}$ guarantees that $\mathbb{E}(\mathbb{F}_q)$ has a unique subgroup of order n as $\#\mathbb{E}(\mathbb{F}_q) \leq (\sqrt{q} + 1)^2$ by Hasse's theorem [5, 13] and so, $n^2 \nmid \#\mathbb{E}(\mathbb{F}_q)$.

4.1. Evaluation of deterministic approach

In this section, we evaluate the deterministic approach of computation of elliptic curves with respect to computational method, computational complexity, security, trust and specific gains for cryptography.

4.1.1. Computational method

Complex Multiplication (CM) is a widely accepted deterministic computational approach for standardization of elliptic curves. The CM method proceeds with fixing the prime field order p first and then constructs an elliptic curve over the field $\mathbb{F}_p$ [11]. It gives a choice for selecting primes of special forms, accepts the order of the elliptic curve field p as input, and determines the CM discriminant D. The field order p is selected such that it meets the conditions C1, C2 and C3. The CM method is efficient when the finite field size p and the field order $\#\mathbb{E}(\mathbb{F}_q) = p + 1 - t$ are chosen such that CM-field of $\mathbb{E}$, i.e., $\mathbb{Q}\left(\sqrt{(t^2 - 4p)}\right)$ has small class number [4, 5]. A crucial step of CM method is to compute the roots of a special type of class field polynomials

called the Hilbert and Weber polynomials [14]. These polynomials are uniquely determined by D. Equations (3) and (4) [15], and (5) [16] constitute the basis of computation of Short Weierstrass elliptic curves using CM method.

Definition 6 (Twist). Given $\mathbb{E}: y^2=x^3+ax+b$ with $a, b \in \mathbb{F}_p$ the twist of $\mathbb{E}$ by c is the elliptic curve given by

(3) $$\mathbb{E}_c: y^2=x^3+ax+b,$$

where $c \in \mathbb{F}_p$.

Theorem 1. If the order of an elliptic curve is $\#\mathbb{E}(\mathbb{F}_p)=p+1-t$, then the order of its twist is given as

(4) $$\begin{cases} \mathbb{E}_c(\mathbb{F}_p{}^*) = (p+1-t) \text{ if } c \text{ is square in } \mathbb{F}_p, \\ (p+1+t) \text{ if } c \text{ is non-square in } \mathbb{F}_p. \end{cases}$$

Theorem 2 (Atkin-Morain). Let p be an odd prime such that

(5) $$4p=t^2+Ds^2,$$

for some $t, s \in \mathbb{Z}$. Then, there is $\mathbb{E}(\mathbb{F}_p)$ such that $\#\mathbb{E}(\mathbb{F}_p)=p+1-t$ [16].

The CM method is called the Atkin-Morain method when the elliptic curve is derived over prime field [37]. Equation (5) observes that D is the integer which can be determined from a given prime p called the CM discriminant of p. Algorithm 1 describes a general CM method [38] for constructing an elliptic curve over a given prime field.

Algorithm 1. Elliptic curve generation over prime field using CM approach

Input: Nil

Output: Elliptic curve over a prime field $\mathbb{E}(\mathbb{F}_p)$

Step 1. Choose elliptic curve field order p, a prime

Step 2. Find smallest CM discriminant D from equation (5) along with trace t

Step 3. Construct the orders of the two elliptic curve $\mathbb{E}(\mathbb{F}_q)=p+1\pm t$

Step 4. if one of the curve orders is a prime or nearly a prime

Step 5. Fix elliptic curve order

Step 6. else Repeat Step 1 to determine D and t

Step 7. end if

Step 8. Construct the class polynomial $H_D(x)$ //Class polynomial is independent of p

Step 9. Find a root j_0 of $H_D(x) \pmod p$ // j_0 is the j-invariant of the desired elliptic curve

Step 10. Set $k=j_0/(1728 - j_0) \pmod p$ // such that $\mathbb{E}: y^2=x^3+3kx+2k$

Step 11. if $\#\mathbb{E} \neq p+1-t$

Step 12. Construct the twist $\mathbb{E}_c$ //using a randomly selected non-square $c \in \mathbb{F}_p$ following equations (3) and (4)

Step 13. **return** $\mathbb{E}_c$

Step 14. else

Step 15. return $\mathbb{E}$

Step 16. end if

4.1.2. Computational complexity

The bit complexity (β) of CM method depends on b and h where b= length of field order p, h= class number, h_c= cross over class number for which the random approach

and CM approach have the same runtime. When $h(D) < h_c(b)$ where D is the CM discriminant, then CM method is faster than random approach [11]. CM method can generate a prime order elliptic curve in time $\tilde{O}((\log N)^4)$ [38].

4.1.3. Security

Deterministic approach is vulnerable to non-disclosed attacks. Bernstein et. al. [39] showed that standards can be sometimes purposely designed in such a way that it can be manipulated by the agency who recommended those standards. Also, sufficient information about the computational mechanisms of curve parameters has not been made publicly available [7]. It is always a concern for researchers that the ECDLP of deterministically computed elliptic curves can be solvable by using very efficient sub-exponential or polynomial time algorithm using non-guessable very high computing power unknown to outside world.

4.1.4. Trust

The elliptic curve parameters which are selected deterministically are sometimes distrusted due to lack of sufficient proofs of their computational mechanisms [40]. Moreover, trust in the curve parameters is doubtful due to possibility of intentional non-disclosed properties of the curve parameters. There are some serious statements of distrust expressed by many reputed scientists and researchers on NIST recommended elliptic curves which was generated through deterministic approach. Some of such statements of distrust are given as below:

- "I no longer trust the constants. I believe the National Security Agency (NSA) has manipulated them through their relationships with industry." – Bruce Schneier (see [41]).
- "NIST should generate a new set of elliptic curves for use with ECDSA in FIPS 186... The set of high-quality curves should be described precisely in the standard, and should incorporate the latest knowledge about elliptic curves." – Edward Felten (see [42, 43]).
- "NIST should ensure that there are no secret or undocumented components or constants in its cryptographic standards whose origin and effectiveness cannot be explained." – Steve Lipner (see [42, 43]).
- "However, in practice the NSA has had the resources and expertise to dominate NIST, and NIST has rarely played a significant independent role." – Koblitz, Koblitz and Menezes [7].
- "We don't know how $Q = [d]P$ was chosen, so we don't know if the algorithm designer [NIST] knows [the backdoor] d." – Shumow and Ferguson (see [44]).
- "Consider now the possibility that one in a million of all curves have an exploitable structure that "they" know about, but we don't. Then "they" simply generate a million random seeds until they find one that generates one of "their" curves." – Scott [45].
- Many more.

4.1.5. Specific gains of deterministic approach

CM method adheres to "Performance over slightly sacrificed security" principle for computation of elliptic curves. Fast elliptic curve computation is possible in CM method due to elimination of the need for a point counting algorithm and fixing of certain parameters like prime p with special structures [40]. CM method allows much faster arithmetic with elliptic curves as compared to random approach to achieve higher performance of elliptic curve cryptosystems [5]. It provides smaller, faster and easily implementable software code due to offline precalculations while adopting deterministic computational approach [46]. Prime order elliptic curves generated using CM method with $a = -3$ are backward compatible with implementation supporting most of the standardized elliptic curves [42]. CM method can only be adopted to construct ordinary elliptic curves with low embedded degree $k > 6$ [7]. CM method is not efficient if there is no restriction on the class number of the elliptic curve [8]. This method is useful in deriving elliptic curves with small class numbers for which ECDLP is hard and gives the same security level as given by the elliptic curves which are generated randomly [5, 8].

4.2. Evaluation of random approach

Random approach allows obtaining elliptic curves, which are ordinary, and avoids any special form or structure. This approach uses 'early-abort strategy' to obtain desired elliptic curve [5]. A general observation is that elliptic curves generated using random approach have not been given preference for standardization. We evaluate random approach from computational method, computational complexity, security, trust and specific gains perspectives in this section.

4.2.1. Computational method

In random approach, the elliptic curve generation algorithm computes curve parameters keeping ECDLP security and procedural transparency in consideration. Algorithm 2 describes a general random approach as preferred in [3-6, 11, 17, 18, 27, 33, 38] to derive cryptographically safe elliptic curve over prime field.

Algorithm 2. Elliptic curve generation over prime field using random approach
Input: Randomness
Output: Elliptic curve $\mathbb{E}(\mathbb{F}_p)$, base point $G_{x,y}$, curve order N
Step 1. Select randomly a prime p of desired size
Step 2. Fix $K = \mathrm{GF}(p)$ // Generate Field K of order p
Step 3. Choose randomly coefficient a
Step 4. Choose randomly coefficient b
Step 5. Generate $\mathbb{E}(K)$ // Elliptic curve over $\mathbb{F}_p$
Step 6. if $4a^3 + 27b^2 \neq 0$ // Non-singularity check
Step 7. else go to Step 3
Step 8. end if
Step 9. Compute order N of $\mathbb{E}$
Step 10. if N is prime // To resist Pohlig-Hellman attack
Step 11. else go to Step 3
Step 12. end if

Step 13. **if** $\mathbb{E}$ is supersingular // To resist MOV attack
Step 14. else go to Step 3
Step 15. end if
Step 16. if $N \neq p$ // Non-anomalous check
Step 17. else go to Step 3
Step 18. end if
Step 19. Select randomly a base point $G_{x,y}$ on $\mathbb{E}$
Step 20. Compute base point order n // $n \geq 160$ bits and $n > 4\sqrt{p}$
Step 21. if $n \neq N$ // Check for cofactor as 1
Step 22. else go to Step 19
Step 23. end if
Step 24. Compute Twist $\mathbb{E}_c$ // For twist security of elliptic curve
Step 25. Compute order N' of $\mathbb{E}_c$
Step 26. if $\mathbb{E}_c$ is non-singular & N' is prime & $\mathbb{E}_c$ is non-supersingular // All criteria to be met for $\mathbb{E}_c$
Step 27. else go to Step 3
Step 28. end if
Step 29. return $\mathbb{E}(\mathbb{F}_p)$, $G_{x,y}$, N // Return elliptic curve parameters

Here, the prime field p is fixed and coefficients a and b are kept varying until a suitable elliptic curve $\mathbb{E}$ with prime order N is obtained. Some validations to meet the cryptographic requirements C1, C2 and C3 are also kept. We observe that all the elliptic curve parameters such as p, a, b and $G_{x,y}$ are randomly generated in order to avoid any special structure or known values whose choices are ambiguous.

4.2.2. Computational complexity

For random approach, the bit complexity (β) only depends on length of prime (r_0) and falls in the range $O(\log^{5+\epsilon} k_0 r_0)$ to $O(\log^7 k_0 r_0)$ where $\epsilon > 0$ and k_0 is the cofactor [11].

4.2.3. Security

Random approach does not allow any special structure of curve parameters in order to eliminate doubts on intentional non-disclosure of backdoors [5]. Elliptic curves, which are randomly computed, have no hidden goals that can be proved in determination of the curve parameters. This ensures that the elliptic curve parameters are trusted and not suspected to belong to a (not publicly known to be) vulnerable class. This approach is favourable when long-term security is desired with an ignorable sacrifice of efficiency [7]. Elliptic curves can be frequently changed for security reasons when computed randomly [40]. The only way to compromise elliptic curve security in such case is to solve ECDLP rather than just attacking particular classes of weak elliptic curves. Hence, random approach is specifically preferred to obtain elliptic curves for implementation in strategic or military grade cryptosystems.

4.2.4. Trust

Random approach ensures that no intentional construction with hidden weakness in the elliptic curve parameters is present in order to prevent future exploitation to

recover user's private key [5]. The trust in derivation of the elliptic curve parameters is maintained due to the use of absolutely new values drawn randomly each time. Moreover, there are no patent issues with randomly selected new curve parameters. Random approach protects against attacks in special classes of elliptic curves, which may be vulnerable in future [5]. However, random values of elliptic curve parameters are always arguable by others for their emanation and random number generation, in case they are not explained adequately.

4.2.5. Specific gains of random approach

Random approach adheres to the principle of "security over performance" for computation of elliptic curve parameters. Computing order of the elliptic curve is a time-intensive task and hence, selecting elliptic curve using random approach is a slower process as compared to the deterministic approach where one starts with fixing the order of the elliptic curve. Point compression and decompression also require more computation in randomly generated elliptic curves [40]. Elliptic curves are computed with nearly the same probability to ensure that curves are not special in any sense when they are computed randomly [5, 11].

5. Approaches adopted by agencies for elliptic curve computation

Many agencies have recommended elliptic curves over various security levels for standardization. Table 4 depicts the popular standard elliptic curves in Short Weierstrass form with their computational approaches. Here, randomly generated elliptic curves means those elliptic curves whose parameters like field order p, field coefficients a, b and basepoint $G_{x,y}$ are randomly or pseudo-randomly (a secure hash function is used to generate curve parameters from random value given as input to the hash function to confirm that parameters are indeed computed pseudo randomly) generated or otherwise, they are considered to be obtained from the deterministic approach. Clearly, from Table 4, the trend demonstrates that the CM method, i.e., the deterministic approach is the preferred computational approach for standardization of elliptic curves.

Table 4. Computational approach adopted for Short Weierstrass elliptic curve computation

Name of elliptic curve	Agency	Year	Security level in bits	Approach
NIST [23]	National Security Agency (NSA)	2001	112, 128, 192, 256	Deterministic
Brainpool [25, 26]	European Consortium of Companies and Government	2005	128, 192, 256	Pseudo-random
ANSSI FRP256v1 [39]	ANSSI	2011	128	Random
SECG [24]	Certicom	2000	112, 128, 192, 256	Deterministic
NUMS-Curves [28, 42]	Microsoft Research	2014	128, 192, 256	Deterministic
Russian Standardized Curves [47] GOST R 34.10-2001 GOST R 34.10-2012 GOST R 34.11-2012	Russian National Cryptographic Standards	2001, 2012	128, 256	Deterministic

6. Standard and non-standard elliptic curves

Elliptic curves are standardized to enable compatibility and interoperability across diverse applications. Moreover, non-standard elliptic curves are mostly used by strategic applications such as military applications or non-military but other critical infrastructure applications such as nuclear reactors' command and control systems etc. These applications do not really believe in Kerckhoffs's principle [48] of security, which says "A cryptographic system should be secure even if everything about the system, except the key, is public knowledge". Unlike Kerckhoff's principle, the strategic applications do believe that not only the keys but the algorithm too should also be kept private to protect critical information infrastructure better. In such cases, they compute elliptic curves preferably using random approach instead of deterministic approach. Table 5 compares between the standard and non-standard elliptic curves from computation, trust and security perspectives to help the readers about their usability concerns.

Table 5. Standard elliptic curves versus non-standard elliptic curves

Standard elliptic curve	Non-standard elliptic curve
Prefers deterministic approach of computation to get performance benefits in elliptic curve arithmetic. This helps in standardization of elliptic curves by global acceptance	Prefers random approach of computation for long term security so that any special kind of curve is avoided which may lead to vulnerability to an unanticipated attack
Adheres to Kerckhoffs's principle of security and fixes elliptic curves for compatibility and interoperability among diverse applications across the globe	Adheres mostly to strategic principle of security which says that keys and algorithm both needs to be kept secret
Standard elliptic curves are subject to public exposure and often attract cryptanalysis as more people use it. Hence, there is always a high chance of collision with the secret key [49]	Negligible chance of collision with the secret key that's why random approach is preferred
Distrust comes with presence of special structures of the curve parameters	Trusted new values of curve parameters known to designer only. Prefers random approach to compute elliptic curve parameters
Standard elliptic curves are globally accepted and trusted	Not published and mostly not supported by the standards. Hence, trusted by their proposers or/and in closed group only
Compatible across applications and interoperable due to standardization	Not compatible. Applications need to be made interoperable explicitly
Better approach in case where elliptic curve needs to be computed over large prime fields	Better approach in case where elliptic curve needs to be transparently computed without any special structures known to others [50]
Curve parameters and compression techniques have patent issues	No patent issues
Already published and analysed thoroughly. Non deniable chances of hiding backdoors	Derivation procedure of curve parameters are known to the proposers only and hence, negligible chances of backdoors. High degree of trust observed by the proposers of non-standard elliptic curves
Standard elliptic curves are fixed to maintain compatibility among applications	Non-standard elliptic curves have edge over the standard ones as they can be replaced frequently for added security
More prone to get attacked by sophisticated advancements in mathematics and discoveries	In case of randomly selected curve parameters, curve is safe until sub-exponential algorithm is known to break it in particular [33]

7. Conclusion and future directions

Short Weierstrass elliptic curves are widely used for cryptographic purposes. An evolution chart of events is presented which has significant impact on introducing

elliptic curves for use in cryptography. We discuss about important attacks on ECDLP and their countermeasures, which became the basic selection criteria of elliptic curves for their consideration in cryptography. This paper also discuss rationale behind the selection criteria used to compute cryptographically suitable elliptic curve parameters. Two popular approaches, i.e., deterministic and random approaches to compute cryptographically secure Short Weierstrass elliptic curves and rationale behind them are evaluated in detail. A trend of approaches for computation of elliptic curve parameters for cryptographic purposes is also demonstrated which favours deterministic approach in standardization so far. We also differentiate between standard and non-standard elliptic curves with respect to their computational approaches, trust and security and bring out the desirable facts to choose either of them on need basis. Hence, it is inferred that this comprehensive evaluation and analysis of computational approaches of cryptographically safe elliptic curves will be helpful to those who wish to compute Short Weierstrass elliptic curves for design of cryptosystems with desired properties of the elliptic curves.

Standardization of elliptic curves, which are computed using random approach will be, preferred in future citing the trust requirements of strategic applications.

***Acknowledgements*:** The authors would like to thank SETS for giving the opportunity to extensively work on elliptic curves and to write this article. Authors would also like to show their gratitude to Dr. Ananda Mohan P. V. and Dr. Reshmi T. R. for their valuable suggestions.

References

1. Koblitz, N. A Course in Number Theory and Cryptography. Springer Science & Business Media, 2 September 1994.
2. Caelli, W. J., E. P. Dawson, S. A. Rea. PKI, Elliptic Curve Cryptography, and Digital Signatures. – Computers & Security, Vol. **18**, 1 January 1999, No 1, pp. 47-66.
3. Valenta, L., N. Sullivan, A. Sanso, N. Heninger. In Search of CurveSwap: Measuring Elliptic Curve Implementations in the Wild. – In: Proc. of 2018 IEEE European Symposium on Security and Privacy (EuroS&P'18), 24 April 2018, IEEE, pp. 384-398.
4. Koblitz, N., A. Menezes, S. Vanstone. The State of Elliptic Curve Cryptography. Designs, Codes and Cryptography. – Vol. **19**, March 2000, No 2, pp. 173-193.
5. Hankerson, D., A. Menezes, S. Vanstone. Guide to Elliptic Curve Cryptography. Springer, 2003. ISBN: 0-387-95273-X.
6. Menezes, A. J., T. Okamoto, S. A. Vanstone. Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field. – IEEE Transactions on Information Theory, Vol. **39**, September 1993, No 5, pp. 1639-1646.
7. Koblitz, A. H., N. Koblitz, A. Menezes. Elliptic Curve Cryptography: The Serpentine Course of a Paradigm Shift. – Journal of Number Theory, Vol. **131**, 1 May 2011, No 5, pp. 781-814. DOI:10.1016/j.jnt.2009.01.006.
8. Smart, N. P. The Discrete Logarithm Problem on Elliptic Curves of Trace One. – Journal of Cryptology, Vol. **12**, 1 Jun 1999, No 3, pp. 193-196.
9. Flori, J. P., J. Plût, J. R. Reinhard, M. Ekerå. Diversity and Transparency for ECC. – IACR Cryptol. ePrint Arch., 11 Jun 2015, 2015/659.
10. Menezes, A. Evaluation of Security Level of Cryptography: The Elliptic Curve Discrete Logarithm Problem (ECDLP). University of Waterloo, 14 December 2001.
11. Baier, H., J. Buchmann. Generation Methods of Elliptic Curves. An Evaluation Report for the Information-Technology Promotion Agency, Japan, 27 August 2002.
12. Semaev, I. Evaluation of Discrete Logarithms in a Group of p-Torsion Points of an Elliptic Curve in Characteristic p. – Mathematics of Computation, Vol. **67**, 1998, No 221, pp. 353-356.

13. W a s h i n g t o n, L. C. Elliptic Curves: Number Theory and Cryptography. CRC Press, 3 April 2008.
14. K o n s t a n t i n o u, E., A. K o n t o g e o r g i s, Y. C. S t a m a t i o u, et al. – J Cryptol., Vol. **23**, 2010, 477.
https://doi.org/10.1007/s00145-009-9037-2
15. K o n s t a n t i n o u, E., A. K o n t o g e o r g i s, Y. C. S t a m a t i o u, C. Z a r o l i a g i s. On the Efficient Generation of Prime-Order Elliptic Curves. – Journal of Cryptology, Vol. **23**, 1 July 2010, No 3, pp. 477-503.
16. S a v a ş, E., T. A. S c h m i d t, C. K. K o ç. Generating Elliptic Curves of Prime Order. – In: Proc. of International Workshop on Cryptographic Hardware and Embedded Systems, 14 May 2001. Berlin, Heidelberg, Springer, pp. 142-158.
17. S c h o o f, R. Elliptic Curves over Finite Fields and the Computation of Square Roots mod p. – Mathematics of Computation, Vol. **44**, 1985, No 170, pp. 483-494.
18. I. F. Blake, G. Seroussi, N. P. Smart, Eds. Advances in Elliptic Curve Cryptography. – Cambridge University Press, Vol. **317**, 25 April 2005.
19. M e n e z e s, A. J., T. O k a m o t o, S. A. V a n s t o n e. Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field. – IEEE Transactions on Information Theory, Vol. **39**, September 1993, No 5, pp. 1639-1646.
20. S h o r, P. W. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. – In: Proc. of 35th Annual Symposium on Foundations of Computer Science, IEEE, 20 November 1994, pp. 124-134.
21. B a l a s u b r a m a n i a n, R., N. K o b l i t z. The Improbability that an Elliptic Curve has Subexponential Discrete Log Problem under the Menezes-Okamoto-Vanstone Algorithm. – Journal of Cryptology, Vol. **11**, 1 March 1998, No 2, pp. 141-145.
22. G a u d r y, P., F. H e s s, N. P. S m a r t. Constructive and Destructive Facets of Weil Descent on Elliptic Curves. – Journal of Cryptology, Vol. **15**, March 2002, No 1, pp. 19-46.
23. PUB, F. Digital Signature Standard (DSS). – FIPS PUB. 27 January 2000, pp. 186-192.
24. SEC, S. 2: Recommended Elliptic Curve Domain Parameters. Standards for Efficient Cryptography Group, Certicom Corp., September 2000.
25. B r a i n p o o l, E. C. ECC Brainpool Standard Curves and Curve Generation. October, 2005.
http://www. ecc-brainpool. org.
26. L o c h t e r, M., J. M e k l e. RFC 5639: ECC Brainpool Standard Curves & Curve Generation. Internet Engineering Task Force, March 2010.
27. B e r n s t e i n, D. J., T a n j a L a n g e. SafeCurves: Choosing Safe Curves for Elliptic Curve Cryptography. 2015. Citations in this document. September 2014.
https://safecurves.cr.yp.to
28. L i u, Z., H. S e o. IoT-NUMS: Evaluating NUMS Elliptic Curve Cryptography for IoT Platforms. – IEEE Transactions on Information Forensics and Security, Vol. **14**, 13 July 2018, No 3, pp. 720-729.
29. C h e n, L., et. al. Report on Post-Quantum Cryptography. US Department of Commerce, National Institute of Standards and Technology, 28 April 2016.
30. R o e t t e l e r, M., M. N a e h r i g, K. M. S v o r e, K. L a u t e r. Quantum Resource Estimates for Computing Elliptic Curve Discrete Logarithms. – In: Proc. of International Conference on the Theory and Application of Cryptology and Information Security, 3 December 2017, Cham, Springer, pp. 241-270.
31. P o h l i g, S., M. H e l l m a n. An Improved Algorithm for Computing Logarithms over GF(p) and Its Cryptographic Significance (Corresp.). – IEEE Transactions on Information Theory, Vol. **24**, January 1978, No 1, pp. 106-110.
32. V a n O o r s c h o t, P. C., M. J. W i e n e r. Parallel Collision Search with Cryptanalytic Applications. – Journal of Cryptology, Vol. **12**, January 1999, No 1, pp. 1-28.
33. B o s, J. W., C. C o s t e l l o, P. L o n g a, M. N a e h r i g. Selecting Elliptic Curves for Cryptography: An Efficiency and Security Analysis. – Journal of Cryptographic Engineering, Vol. **6**, November 2016, No 4, pp. 259-286.
34. H a m b u r g, M. Ed448-Goldilocks, a New Elliptic Curve. IACR Cryptol. ePrint Arch. Jun 2015, 2015/625.

35. C r a n d a l l, R. E. Method and Apparatus for Public Key Exchange in a Cryptographic System. October 1992. US Patent. (5,159,632).
36. D ą b r o w s k i, P., R. G l i w a, J. S z m i d t, R. W i c i k. Generation and Implementation of Cryptographically Strong Elliptic Curves. – In: Proc. of International Conference on Number-Theoretic Methods in Cryptology, 11 September 2017. Cham, Springer, pp. 25-36.
37. M o r a i n, F. Building Cyclic Elliptic Curves Modulo Large Primes. – In: Proc. of Workshop on the Theory and Application of of Cryptographic Techniques, 8 April 1991. Berlin, Heidelberg, Springer, pp. 328-336.
38. C o h e n, H. A Course in Computational Algebraic Number Theory. Berlin, Springer-Verlag, August 1993.
39. B e r n s t e i n, D. J., et.al. How to Manipulate Curve Standards: A White Paper for the Black Hat. – In: Proc. of International Conference on Research in Security Standardisation, 15 December 2015, Cham, Springer, pp. 109-139.
http://bada55.cr.yp.to
40. S a n, C. V. A Survey of Elliptic Curve Cryptosystems. Part I. Introductory. Technical Report, NAS Technical Report-NAS-03-012, 2003.
41. **https://www.schneier.com/blog/archives/2013/09/the_nsa_is_brea.html**
42. C o s t e l l o, C., P. L o n g a, M. N a e h r i g. A Brief Discussion on Selecting New Elliptic Curves. Microsoft Research. Microsoft, 8 June 2015.
43. **https://www.nist.gov/system/files/documents/2017/05/09/2014-VCAT-Annual-Report_final.pdf**
44. **https://cryptosith.org/michael/data/talks/2015-04-28-UWNumberTheorySeminar.pdf**
45. S c o t t, M. Re: NIST Announces Set of Elliptic Curves. Posting to the sci.crypt Mailing List, 1999.
46. S a v a ş, E., T. A. S c h m i d t, C. K. K o ç. Generating Elliptic Curves of Prime Order. – In: Proc. of InInternational Workshop on Cryptographic Hardware and Embedded Systems, 14 May 2001. Berlin, Heidelberg, Springer, pp. 142-158.
47. A l e k s e e v, E. K., V. D. N i k o l a e v, S. V. S m y s h l y a e v. On the Security Properties of Russian Standardized Elliptic Curves. Vol. **9**, 2018, No 3, pp. 5-32.
48. P e t i t c o l a s, F. La cryptographie Militaire. 1883.
49. Y i n, E. Curve Selection in Elliptic Curve Cryptography. San Jose State University, Project, Spring, 2005.
50. A b h i s h e k, K., E. G. R a j. Computation of Trusted Short Weierstrass Elliptic Curves for Cryptography. – Cybernetics and Information Technologies, Vol. **21**, 2021, No 2, pp. 70-88.

Received: 06.07.2021; Second Version: 26.10.2021; Accepted: 08.11.2021

BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **21**, No 2
Sofia • 2021 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2021-0020

Computation of Trusted Short Weierstrass Elliptic Curves for Cryptography

Kunal Abhishek[1], *E. George Dharma Prakash Raj*[2]

[1]*Society for Electronic Transactions and Security, Chennai, India*
[2]*Bharathidasan University, Tiruchirappalli, India*
E-mails: *kunalabh@gmail.com georgeprakashraj@yahoo.com*

***Abstract*:** *Short Weierstrass elliptic curves with underlying hard Elliptic Curve Discrete Logarithm Problem* (*ECDLP*) *are widely used in cryptographic applications. A notion of security called Elliptic Curve Cryptography* (*ECC*) *security is also suggested in literature to safeguard the elliptic curve cryptosystems from their implementation flaws. In this paper, a new security notion called the "trusted security" is introduced for computational method of elliptic curves for cryptography. We propose three additional "trusted security acceptance criteria" which need to be met by the elliptic curves aimed for cryptography. Further, two cryptographically secure elliptic curves over 256 bit and 384 bit prime fields are demonstrated which are secure from ECDLP, ECC as well as trust perspectives. The proposed elliptic curves are successfully subjected to thorough security analysis and performance evaluation with respect to key generation and signing/verification and hence, proven for their cryptographic suitability and great feasibility for acceptance by the community.*

***Keywords*:** *Short Weierstrass elliptic curves, prime field, cryptography, ECDLP Security, ECC Security, Trusted Security.*

1. Introduction

Short Weierstrass elliptic curves are considered to be as secure for cryptography as the underlying hardness of their Elliptic Curve Discrete Logarithm Problem, i.e., (ECDLP) which is defined as finding a scalar k knowing any two points P and Q on elliptic curve $\mathbb{E}$ holding the relation $Q = kP$. This is known as the ECDLP security of the selected elliptic curve when used for cryptography [1]. The most efficient publicly known method to solve ECDLP or break the ECDLP security is the Pollard's rho algorithm which takes approximately $0.886\sqrt{n}$ point additions where n is the base point order [1-2]. One must select an elliptic curve which is ECDLP secure for cryptographic applications. Another notion of security for selecting suitable elliptic curves for cryptography is known as elliptic curve cryptography security, i.e., ECC

security in short, the term coined by Bernstein and Lange [1] which ensures prevention from any information leakage from the implementation flaws of the elliptic curve.

Most of the popular standards today such as National Institute of Standards and Technology (NIST) [3], Brainpool [4], Standards for Efficient Cryptography 2 (SEC2) [5], IEEE P1363 [6], etc., recommended those elliptic curves which are ECDLP secure and attain some sort of ECC security (for only some standard curves [1]). It is worthwhile to note that an ECC based cryptosystem can be compromised by either compromising the ECDLP security or the ECC security. All the present day standards have recommended Short Weierstrass elliptic curves keeping either or both of these security notions into consideration. This paper introduces a critical security notion which we call as "trusted security" of elliptic curves which ensures that the selected elliptic curve is free from any manipulation from its computation perspective and can be trusted for use in cryptographic applications. The trusted security notion of computation of elliptic curves minimizes the risks involved in generation of safe curve parameters deterministically where they are vulnerable to (intentionally) non-disclosed attacks with (intentionally) non-disclosed properties of the curve parameters. In such cases, the ECDLP can be solvable by using very efficient sub-exponential or polynomial time algorithm using non-guessable high computing power.

The key contributions of this paper are as follows:

1. Introduction of a new security notion called as "trusted security acceptance criteria" as an important security evaluation criterion along with the ECDLP security and ECC security criteria for computation of Short Weierstrass elliptic curves aimed for cryptography.

2. Evaluation of standard Short Weierstrass elliptic curves from trust perspective.

3. Argument that trust in generation method of elliptic curves can be achieved only through computation of the curve parameters randomly without considering any of their pre-studied values such as $a = -3$ or p as Mersenne primes, etc. The randomly selected elliptic curve parameters can be derived using any good quality user trusted Random Number Generator (RNG) along with competitive curve performance.

4. Demonstration of two new elliptic curves called as Kunal-George 256 bit first random elliptic curve (KG256r1) and Kunal-George 384 bit first random elliptic curve (KG384r1) defined over 256 bit and 384 bit prime field sizes respectively for cryptography which are secure from ECDLP security, ECC security as well as trusted security perspectives.

5. Evaluation of the proposed elliptic curves KG256r1 and KG384r1 with respect to cryptographic key pair generation, signing and verification from performance perspective.

Organization of the paper is as follows.

Section 2 deals with the background and problem statements of the presented work. Section 3 introduces the proposed "trusted security acceptance criteria" for cryptographically safe elliptic curve computation. Section 4 evaluates standard Short Weierstrass elliptic curves from trusted security acceptance criteria perspective.

Section 5 describes the generation procedure including the proposed trusted security acceptance criteria to derive new elliptic curves KG256r1 and KG384r1 for evaluation and demonstration. Section 5 also holds the discussion on importance of trusted security acceptance criteria of elliptic curves to minimize the risk of manipulating the curve parameters intended for cryptographic purposes. Section 6 presents demonstration of the proposed trusted Short Weierstrass elliptic curves for cryptography. Section 7 gives the security analysis of the proposed elliptic curves. Section 8 discusses results obtained in the presented work and demonstration of the performance metrics of the proposed elliptic curves. Finally, Section 9 concludes the paper and gives future directions.

2. Background and problem statements

An elliptic curve in Short Weierstrass form consists of three parameters: a prime number p which is the order of the underlying field over which the elliptic curve is defined and two field coefficients a and b. The formal definition of a Short Weierstrass elliptic curve and its twisted curve are as follows:

Definition 1 [7]. A Short Weierstrass elliptic curve $\mathbb{E}(\mathbb{F}_p)$ of prime field order p is the set of all solutions (x, y) to the equation

$$\mathbb{E}: y^2 = x^3 + ax + b, \tag{1}$$

where a, b are the coefficients in $\mathbb{F}_p$ with field characteristic greater than 3. The elliptic curve $\mathbb{E}$ also includes a special point $\mathbb{O}$ called the point at infinity. $\mathbb{E}$ has non-singularity condition, i.e., its discriminant $\triangle_{\mathbb{E}} = 4a^3 + 27b^2 \neq 0$.

The field order p determines the security level offered by the elliptic curve. Hence, it is important to select p as big as possible. Generally, $p \geq 256$ bits in size gives accepted security level while p of 256 bit length is considered as widely accepted prime field size of the elliptic curve for interoperability purposes.

Definition 2 [8]. If $\mathbb{E}: y^2 = x^3 + ax + b$ be an elliptic curve with $a, b \in \mathbb{F}_p$ the twist of $\mathbb{E}$ by $c \in \mathbb{F}_p$ is defined as

$$\mathbb{E}': y^2 = x^3 + ac^2x + bc^3. \tag{2}$$

It is important to select those elliptic curves which are cryptographically secure and trusted for constructing cryptographic systems. Transport Layer Security (TLS), Secure SHell (SSH) and Internet Protocol Security (IPSec) [9], Public Key Infrastructure (PKI) [10], etc., are some of the popular applications which require safe elliptic curves in their cryptosystem design. Most of such commercial applications use standard elliptic curves over prime field of 256 bit sizes for sufficient security and interoperability purposes. However, Bernstein et al. [2] have recently pointed out some mechanisms such that a new elliptic curve can be proposed to sabotage public standards. They demonstrated convincing methods by which they were able to implant vulnerability in the elliptic curves known as BADA55 curves by utilizing the gain of many bits of freedom [2] which satisfies the public standards and can be put forward for standardization to fool the users. This essentially proves that an attacker can exploit unknown (known to him) vulnerability to sabotage existing public standards and justify his selection of elliptic curve parameters citing performance gain and his own way of getting randomness, i.e., verifiably random,

etc., which is used in the generation of the vulnerable curve parameters. B e r n s t e i n et al. [2] comprehensively demonstrated how a wrong or non-trustable elliptic curve can be derived using the procedure led by the public standards and their recommended public criteria. They showed that plausible variations in the Brainpool curve generation procedure and Microsoft curve generation procedure respectively can be used to sabotage public standard. Further, the Agence Nationale de la Securite des Systemes d'Information (ANSSI) standard recommended FRP256V1 elliptic curve which has low twist security of order 2^{79} which means that there are 2^{79} elliptic curve additions required to mount the twist attack to get user's secret key [2]. Also, there is no reasonably sufficient documentation available for this curve. Furthermore, B e r n s t e i n et. al. [2] demonstrated computation of the BADA55-R-256 curve which meets the public security criteria for ECDLP security and ECC security but still is a manipulated curve. Finally, we understand that computation of an elliptic curve can be manipulated by any deterministic method of computation of the curve parameters and variety of reasons can be cited with selection of the curve parameters adhering to some public standard of proposer's convenience.

Summarizing, the problems pertained with the trust consists of one or more issue(s) from the following:

- No sufficient explanation on the RNG used for seed or randomness generation.
- Intentional variation in standard elliptic curve generation procedure recommended by the curve proposing agencies by themselves.
- Intentional hiding of information about the curve parameters even providing detailed documentations on curve generation process of standard elliptic curves.
- Sabotaged standards.
- Root problem of the lack of trust is the deterministic approach adopted by all the agencies in standardizing their proposed elliptic curves.

With the above prevalent issues, an obvious question arises that "because you can explain, does not mean that you will explain everything". We answer this question by introducing a set of three important security evaluation criteria called "trusted security acceptance criteria" for computation of suitable elliptic curves for cryptography which can be additionally invoked along with the ECDLP security and ECC security criteria to mitigate the trust issues in curve generation process to a great extent.

3. Trusted security acceptance criteria for elliptic curves for cryptography

Standard elliptic curves followed deterministic approach in computation of their coefficients and primes. Most of them used pre-studied values whose credibility and trustworthiness are doubted [2, 11-13] due to origination of the curve parameters and lack of proof for the randomness used in the curve generation process such as use of computationally convenient primes like powers of two, etc. Hence, there is a need to introduce additional security acceptability criteria to invoke trust in the computation of elliptic curve parameters for use and in standardization. In this paper, a set of three

new security evaluation criteria of cryptographically safe elliptic curve called the "trusted security acceptance criteria" for elliptic curve used for cryptography is introduced which is as follows:

a. T1: User trusted Random Number Generator (RNG) to provide (pseudo)randomness.

A RNG should be selected preferably by its user for assuring that user is fully aware of the technicality of the RNG and hence he/she trusts it completely. Apart from the trust aspect, the RNG should adhere to the following properties as indicated by Koc [14] and Schneier [15]:

- The bitstream generated by a PseudoRandom Number Generator (PRNG) or Cryptographically Secure PRNG (CSPRNG) should be statistically sound, i.e., it has a large period.
- The bitstream generated should be unpredictable, i.e., the RNG should be forward secure as well as backward secure.

The curve parameters should be chosen randomly in a trustworthy way to avoid any uneasy explanation about the generation of the curve constants and hence, the requirement of user trusted and strong RNG is critical in trust building.

b. T2: No pre-studied values of the curve coefficients and prime.

The well-known constants are accepted by everyone without hesitation but their non-exposed property may be used for construction of vulnerable elliptic curves. BADA55-VPR-224 is such an example which used $\cos(1)$ constant [2]. The elliptic curve coefficients a, b must not use any pre-studied values to avoid the scope of manipulation. Moreover, the prime field order p can only have special structure if it is randomly selected with suitable size (normally ≥ 224) bits for fast reduction on the elliptic curve.

c. T3: Reproducibility of new elliptic curves of nearly the same cryptographic strength and suitability using the same method and apparatus.

One must get new elliptic curves of nearly the same cryptographic strength using the same method and apparatus. We consider Pollard's rho values of the elliptic curves and their respective twisted curves as the measurement of their cryptographic strengths which is the number of elliptic curve point additions to solve the ECDLP. Generally, $0.886\sqrt{n}$ elliptic curve point additions are required to break the ECDLP where n is the order of the base point [1-2].

4. Evaluation of standard elliptic curves from trust perspective

Standard Short Weierstrass elliptic curves claimed to have followed rigorous ECDLP security validations and sometime ECC security validations together to arrive at the curve parameters for recommendation. They claimed that they used seeds which were randomly generated and some of them adhered to verifiably random way of obtaining the curve parameters. Table 1 evaluates standard elliptic curves from trust perspectives for use in cryptography.

Table 1. Evaluation of the standard Short Weierstrass elliptic curves from trust perspective

Elliptic curve	Trusted Security (T1, T2, T3)	Remarks
NIST P224r1	None	Deterministic approach with pre-studied coefficients and prime [3]
NIST P256r1	None	Deterministic approach with pre-studied coefficients and prime [3]
NIST P384r1	None	Deterministic approach with pre-studied coefficients and prime [3]
secp224r1	None	Special structure of prime p (Mersenne prime) and insufficient documentation [5]
secp256r1	None	Special structure of prime p (Mersenne prime) and insufficient documentation [5]
secp384r1	None	Special structure of prime p (Mersenne prime) and insufficient documentation [5]
secp521r1	None	Special structure of prime p (Mersenne prime) and insufficient documentation [5]
ANSSI FRP256v1 curve	None	Pre-studied value of coefficient a and insufficient documentation [2, 16]
Brainpool	T2	None of the Brainpool curves are generated by their own stipulated procedure [2, 4]
NUMS curves	None	Deterministic approach with pre-studied coefficients and prime [2, 17]

It is imperative to note from Table 1 that, there is an ardent need for new elliptic curves which are cryptographically secure as well as trusted. Following section will focus on the generation details of trusted Short Weierstrass elliptic curves to be used for cryptography.

5. Cryptographically secure elliptic curve generation using the proposed trusted security acceptance criteria

Short Weierstrass elliptic curves have a unique property that it can only exhibit prime order [18] in order to get maximum security of ECDLP without compromising any bit of security [19]. However, elliptic curves of cryptographic interest must get validated against their ECDLP security, ECC security as well as trusted security. It is now observed from previous sections that random approach of computing safe elliptic curves is the only way to achieve all of these three security notions. A standard procedure is shown as the flow chart in Fig. 1 for a bird's eye view of generation of the trusted Short Weierstrass elliptic curves intended for cryptography.

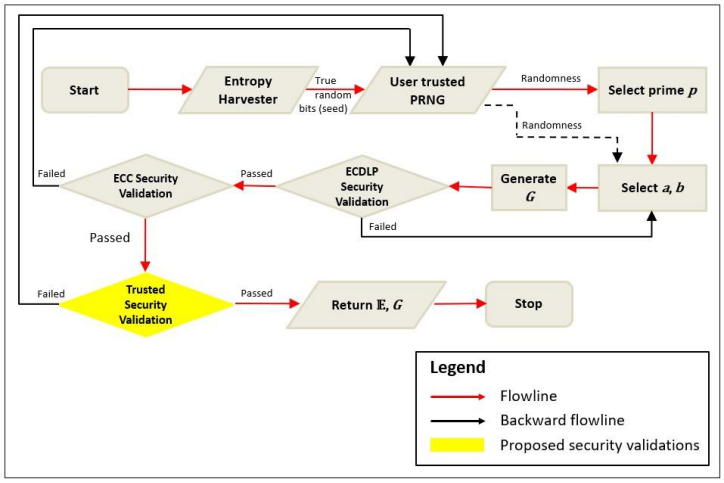


Fig. 1. Flow chart of generation of cryptographically secure and trusted Short Weierstrass elliptic curve

An entropy harvester which is used to obtain sufficient number of true random bits from various physical noise sources like device randomness, disk randomness, Human Interface Device (HID) (key board, mouse, etc.), interrupt randomness, etc., is used to seed a user trusted (means user is aware of the technicality of the RNG and associated security risks completely) PRNG/CSPRNG as depicted in Fig. 1. The user trusted PRNG supplies desired number of (pseudo)random bits to generate suitable p, a and b. An elliptic curve $\mathbb{E}$ is constructed over prime field p (which is fixed in our case, but one can choose other way also to generate suitable elliptic curves by fixing the curve order N randomly, etc.) with coefficients a and b. Now $\mathbb{E}$ is subjected to ECDLP security validation failing which it will regenerate the coefficients a and b until it gets suitable curve coefficients for $\mathbb{E}$ to be ECDLP secure. A base point G is also selected randomly over elliptic curve $\mathbb{E}$ and gets verified for its prime order for acceptability. Once $\mathbb{E}$ is validated for ECDLP security, it is subjected to security validation from ECC security perspective which expects $\mathbb{E}$ to have its twist $\mathbb{E}$' also to be as secure as $\mathbb{E}$ is. In case of the fact that ECC security validation does not pass, one needs to regenerate the prime p and subsequently coefficients a and b to get ECDLP security and ECC security successfully validated. Finally, the ECDLP secure and ECC secure $\mathbb{E}$ is verified with the proposed trusted security acceptance criteria (indicated in yellow decision box in Fig. 1) failing which the process is re-initiated with deriving prime p and coefficients a and b as fresh until one gets an acceptable $\mathbb{E}$. Lastly, $\mathbb{E}$ and G are returned as the output. The elliptic curve generation procedure is detailed in Algorithm 1.

5.1. Assumptions

Following assumptions were made considered while computing the curve parameters using Algorithm 1:

i. User trusted cryptographically strong RNG is available to provide randomness required in computation of secure elliptic curve.

ii. Sufficient entropy is available in the system. Generally, more than 2000 bits of entropy is expected to be available with the system to seed the RNG sufficiently to uninterruptedly generate elliptic curves up to over 384 bit prime field sizes. Also, the operating system is not used for the first time after installation as sufficient entropy will not be available with the machine.

iii. Compilers, CPU Processors, SAGE and other participating modules in the curve parameter generation are trusted.

5.2. Standard elliptic curve generation procedure including trusted security acceptance criteria

Algorithm 1 shows the standard procedure along with the proposed trusted security acceptance criteria as discussed in Fig. 1 with detailed security validations of elliptic curve from ECDLP security, ECC security and trusted security perspectives.

Algorithm 1. Generation of trusted cryptographically safe Short Weierstrass elliptic curve

Input: Prime field size (l) in bits and randomness from user trusted RNG

Output: Trusted cryptographically safe elliptic curve $\mathbb{E}$ over prime field p with base point $G_{x,y}$

Step 1. Input prime field size l in bits

Step 2. Obtain seed S as true random bits of desired length from entropy harvester

Step 3. Set seed S for user trusted RNG

Step 4. Select randomly prime p such that $p \equiv 3 \bmod 4$ // for fast arithmetic on $\mathbb{E}$

Step 5. Choose randomly the coefficient a of $\mathbb{E}$

Step 6. Choose randomly the coefficient b of $\mathbb{E}$

Step 7. Construct the elliptic curve $\mathbb{E}$ with curve parameters p, a and b

Step 8. Enforce ECDLP security validation:

Step 8.1. If discriminant $\triangle_{\mathbb{E}} = 4a^3 + 27b^2 \neq 0$ // $\mathbb{E}$ must be non-singular

Step 8.2. Else go to Step 5

Step 8.3. If curve order N is prime

Step 8.4. Else go to Step 5

Step 8.5. If $\mathbb{E}$ is non-anomalous case // $N \neq p$

Step 8.6. Else go to Step 5

Step 8.7. If $\mathbb{E}$ is not supersingular curve

Step 8.8. Else go to Step 5

Step 8.9. Generate randomly the base point $G_{x,y}$ on $\mathbb{E}$

Step 8.10. Validate if base point order n is prime

Step 8.11. Else go to Step 8.9

Step 8.12. If cofactor is 1

Step 8.13. Else go to Step 5

Step 8.14. If Pollard's rho value $< 2^{100}$

Step 8.15. Else go to Step 5

Step 8.16. If embedding degree $k \geq (N-1)/100$ // guarantees intractability of DLP

Step 8.17. Else go to Step 5

Step 9. Enforce ECC security validation: (If $\mathbb{E}$ is twist secure, i.e., all validations in Step 8 applied to $\mathbb{E}'$)

Step 9.1. If twist discriminant $\triangle_{\mathbb{E}'}$ of $\mathbb{E} = 4a^3 + 27b^2 \neq 0$

Step 9.2. Else go to Step 4

Step 9.3. If order of $\mathbb{E}'$, N is prime

Step 9.4. Else go to Step 4

Step 9.5. If $\mathbb{E}'$ is non-anomalous case

Step 9.6. Else go to Step 4

Step 9.7. If $\mathbb{E}'$ is not supersingular curve

Step 9.8. Else go to Step 4

Step 9.9. Generate randomly the base point $G'_{x,y}$ on $\mathbb{E}'$

Step 9.10. Validate if base point order n' is prime

Step 9.11. Else go to Step 9.9

Step 9.12. If cofactor of $\mathbb{E}'$ is 1

Step 9.13. Else go to Step 4

Step 9.14. If Pollard's rho value of $\mathbb{E}' < 2^{100}$

Step 9.15. Else go to Step 4

Step 9.16. If embedding degree $k' \geq (N' - 1)/100$

Step 9.17. Else go to Step 4

Step 10. Enforce trusted security validation

Step 10.1. Validate if RNG is trusted // Proposed validation criterion T1

Step 10.2. Else go to Step 2

Step 10.3. Validate if coefficients a and b have no pre-studied value // Proposed validation criterion T2

Step 10.4. Else go to Step 2

Step 10.5. Validate if elliptic curves with similar cryptographic strength can be generated with the same method and apparatus // Proposed validation criterion T3

Step 10.6. Else go to Step 2

Step 11. Return $\mathbb{E}$: $\{p, a, b\}$ and $G_{x,y}$

Algorithm 1 takes elliptic curve field size (l) in bits as the input in Step 1. A seed S is extracted from the entropy harvester in Step 2. In our case, we used /dev/random as the PRNG which takes true random bits through a Hardware based RNG (HRNG) that extracts entropy directly. We used /dev/random PRNG available with Linux Fedora kernel Version 4.13.9 for obtaining randomness in desired bit lengths. The HRNG uses various noise sources like input randomness, device randomness, disk randomness, HID (key board, mouse, etc.), interrupt randomness to provide random bits as the seed S to /dev/random in Step 3. S is used to initialize /dev/random to provide randomness to the curve generation process as and when required. As the curve generation program needs a user trusted secure RNG, we leave it to the user to select his/her trusted RNG for fulfilling the randomness requirements. Here our focus is to recommend users to use their own trusted RNGs to avoid any possible manipulation in curve computation and we demonstrate how a trusted Short Weierstrass elliptic curve can be generated for cryptography. In Step 4, the prime p of user desired l bit length is randomly selected and subsequently, checked that it

should hold the form of $p \equiv 3 \bmod 4$ for fast reduction, i.e., fast elliptic curve arithmetic on $\mathbb{E}$. It is noted that p is first chosen randomly and then verified for this form to avoid any pre-studied value. The curve coefficients a and b are then chosen randomly in Step 5 and Step 6 respectively using different seeds, i.e., a and b have independent initializations. Now, an elliptic curve $\mathbb{E}$ is constructed with p, a and b in Step 7.

The ECDLP security validations are enforced in Step 8 which includes non-singularity in Step 8.1, prime curve order in Step 8.3, non-anomalous property in Step 8.5, non-supersingularity in Step 8.7, random selection of base point in Step 8.8 with prime base point order in Step 8.9, small cofactor as 1 in Step 8.11, high Pollard's rho in Step 8.14 and high embedding degree in Step 8.16 respectively. Non-singularity of elliptic curve confirms that curve is smooth and indeed an elliptic curve [20-22]. Prime order elliptic curve with order N is resistant to Pohlig-Hellman attack when $N \geq 2^{160}$ [23]. Non-anomalous case of elliptic curve, i.e., when curve order $N \neq p$, confirms that curve is resistant to pairing based attacks [23]. Non-supersingularity of elliptic curve prevents the ECDLP from the Menezes, Okamoto and Vanstone (MOV) reduction attack with sub-exponential complexity which takes place when the conditions that p divides trace t or/and $t^2 = 0, p, 2p, 3p$ or $4p$ are met [24-25]. The cofactor value determines the cryptographic security and gives maximum security when selected as 1 [23, 25]. The Pollard's rho value of elliptic curve determines the number of elliptic curve point additions to find a collision. This check is very important as a parallelized Pollard-rho on r processors can solve ECDLP in $(\sqrt{\pi n})/\sqrt{2r}$ steps [23, 26]. The embedding degree of elliptic curve $k \geq 20$ is considered sufficient to guarantee intractability of the discrete logarithm problem in the extension field [7].

The ECC security validations are enforced in Step 9 of Algorithm 1 in which it looks for the twist of the selected elliptic curve to be secure against all the ECDLP security validations as described above. The twist security of elliptic curve prevents from any implementation flaws or information leakage about the user's secret key [1].

The trusted security validations are carried out in Step 10 to ensure the method of generation of elliptic curve is trusted in terms of the randomness used in the curve generation process and the curve parameters are drawn randomly. It also ensures that the procedure described in Algorithm 1 can be used to obtain Short Weierstrass elliptic curves of nearly the same cryptographic strength each time on its execution.

Finally, a trusted and secure elliptic curve $\mathbb{E}$: $\{p, a, b\}$ and base point $G_{x,y}$ is returned in Step 11.

6. Demonstration of trusted Short Weierstrass elliptic curves

We used Algorithm 1 to derive two trusted Short Weierstrass elliptic curves KG256r1 and KG384r1 defined over 256 bit and 384 bit respectively for demonstration. The details of the proposed KG256r1 and KG384r1 is shown in Table 2 and Table 3, respectively. These elliptic curves have undergone security analysis in Section 7 to ensure that the elliptic curves generated using Algorithm 1 have nearly the same

cryptographic strength in terms of Pollard's rho complexity and other criteria like big discriminant, embedding degree, trace, etc., while being compliant with the three security notions, i.e., ECDLP security, ECC security and trusted security.

Table 2. The proposed KG256r1 elliptic curve

KG256r1	
p	105659876450476807015340827963890761976980048986351025435035631207814085532543
a	577801306981151765834884991713447710888985073378732385904009553711296851388 26
b	102451950841073747949316796495896937960702115486975363798323596797327090813462
N	105659876450476807015340827963890761976544313325663770762399235394744121359871
G	(538516633311464649781099807461241598582198637115148595458601407868879196006 4, 884401665317899467231260835467506331798660390928837647840416110655479261590 80)
h	1 (smallest cofactor)

Table 3. The proposed KG384r1 elliptic curve

KG384r1	
p	308504936566801493400799664217561138887972017059009663818402880868888024111765 879720207350125234692675645054207640 51
a	268937684885793435941799884521325825414071666675195106719690165313905189264848 525778882798918582235919301325173556 2
b	282679914441081045194064979674986566053141057529253438397674572433074909758239 545163835466127028012727836567748393 9
N	308504936566801493400799664217561138887972017059009663818414387546839003900776 173235655548729960739791037659175227 31
G	(263821674697227290786867915392591910846306526222054061903021467945234141274511 834239141208114870550550647928753455 76, 202628051316606152195895866462280785015451818341996421511941020893449272958898 572935639891270202600201220024040452 04)
h	1 (smallest cofactor)

Resources used. The curve generation programme was written in Python language using Python Version 2 and Python Version 3.6 compilers and ran on a desktop server having 2*Intel® Xeon® E5-2620v4 processor with 32 CPU cores and 2.1 GHz clock frequency and 128 GB DDR4 memory. The desktop server was equipped with Linux Fedora operating system (kernel Version 4.13.9) and SAGE Version 8.1 was used for number theory arithmetic support for the curve generation program.

7. Security analysis of the proposed KG256r1 and KG384r1 elliptic curves

7.1. Analysis of the ECDLP and ECC security of the proposed KG256r1 and KG384r1 elliptic curves

We used SafeCurves verification script [1] to verify ECDLP security and ECC security of the elliptic curve parameters. Algorithm 2 describes the SafeCurves verification script which was used to verify the KG256r1 and KG384r1 elliptic curves against its ECDLP and ECC security.

Algorithm 2. Verification of the proposed elliptic curve parameters for cryptographic security

Input: Elliptic curve parameters $p, a, b, N, G_{x,y}$

Output: Safe/Weak Elliptic Curve

Step 1. Verify if shape of elliptic curve is Short Weierstrass

Step 2. Else return "Not Short Weierstrass elliptic curve"

Step 3. Verify if p is prime

Step 4. Else return "Weak elliptic curve"

Step 5. Verify if discriminant $< -2^{100}$

Step 6. Else return "Weak elliptic curve"

Step 7. Verify if base point order is prime

Step 8. Else return "Weak elliptic curve"

Step 9. Verify if GCD (Curve order, base point order)=1

Step 10. Else return "Weak elliptic curve"

Step 11. Verify if base point is on curve

Step 12. Else return "Incorrect base point"

Step 13. Verify if co-factor is 1 or 2 or 4

Step 14. Else return "Weak elliptic curve"

Step 15. Verify if $p+1-t$ is a multiple of base point order n

Step 16. Else return "Weak elliptic curve"

Step 17. Verify if embedding degree of curve $\geq (N-1)/100$

Step 18. Else return "Weak elliptic curve"

Step 19. Verify if elliptic curve is MOV safe

Step 20. Else return "Weak elliptic curve"

Step 21. Verify if base point order of twist != p

Step 22. Else return "Weak elliptic curve"

Step 23. Verify if twist equation is elliptic

Step 24. Else return "Weak elliptic curve"

Step 25. Verify if twist shape is Short Weierstrass

Step 26. Else return "Weak elliptic curve"

Step 27. Verify co-factor of twist is 1 or 2 or 4

Step 28. Else return "Weak elliptic curve"

Step 29. Verify if GCD (base point order of twist, p) $= 1$

Step 30. Else return "Weak elliptic curve"

Step 31. Verify if Pollard's rho value of elliptic curve $\geq 2^{100}$

Step 32. Else return "Weak elliptic curve"

Step 33. Verify if rigidity is True

Step 34. Else return "Weak elliptic curve"

Step 35. Verify if twist rho value $\geq 2^{100}$

Step 36. Else return "Weak elliptic curve"

Step 37. Verify if Joint Rho $\geq 2^{100}$

Step 38. Else return "Weak elliptic curve"

Step 39. Otherwise, return "Cryptographically safe elliptic curve"

It is obvious that ECDLP security is a critical security requirement for qualifying any elliptic curve for cryptography. However, SafeCurves [1] proposed ECC security as another security notion for evaluating elliptic curves to ensure that the ECC implementations do not reveal or leak information about user's secret key. For Short Weierstrass elliptic curves, a twist secure elliptic curve can prevent ECC implementation flaws [1]. The elliptic curve $\mathbb{E}$ is twist secure if its twist $\mathbb{E}'$ is secure which means that all the ECDLP security validations are also successfully compliant by $\mathbb{E}'$ [1].

Both KG256r1 and KG384r1 elliptic curves qualified all the ECDLP and ECC security verifications executed in Algorithm 2. The field orders p and curve orders N of both elliptic curves were verified deterministically for being a prime number using Pocklington's theorem [1]. We avoided any special structure of prime or pre-studied value to prevent from any vulnerability. For example, NIST P-224 prime, i.e., $p = 2^{224} - 2^{96} + 1$ was used by BADA55-VPR-224 and standard ANSSI prime

0xF1FD178C0B3AD58F10126DE8CE42435B3961ADBCABC8CA6DE8FCF353D86E9C03

was used by BADA55-R-256 curve, respectively, to demonstrate vulnerable curves to the community [2]. Moreover, the discriminants, embedding degrees, cofactor values and Pollard's rho values of both curves and their respective twist curves were verified successfully possessing more than their expected threshold values. These curves were also verified to confirm that they are not a case of anomalous and supersingular ones as discussed in Section 5.2 and thus, they are suitable for cryptography. Table 4 and Table 5 shows these values possessed by both KG256r1 and KG384r1 elliptic curves.

7.2. Analysis of trusted security of KG256r1 and KG384r1 elliptic curves

7.2.1. Validation of Trusted Security Criteria T1

We trust and used /dev/random PRNG for curve generation procedure due to the fact that it has faced a lot of successful cryptanalysis [27-29] and sustained long with the Linux kernel since 1994 [28]. Moreover, the latest versions (Version 4.8 or later) of /dev/random have overcome [30] the criticism of having possible entropy attacks [2]. We used Linux Fedora kernel Version 4.13.9 and selected /dev/random as the PRNG (sometimes /dev/random is referred as True Random Number Generator (TRNG) because it has the direct interface with the HRNG). We are actually making a point here that choose your trusted RNG and own the risk associated with your selection.

7.2.2. Validation of Trusted Security Criteria T2

To validate the T2 criterion, no pre-studied values of the curve coefficients a and b are used as they have been chosen randomly and independently. The prime numbers p in both proposed curves KG256r1 and KG384r1 are selected randomly first and then chosen with a form of $p \equiv 3 \bmod 4$ for performance tuning and there is no evidence of these primes p and coefficients a and b reported in past as the pre-studied ones.

Table 4. Verification result of the ECDLP security of the proposed elliptic curves

Elliptic curve $\mathbb{E}$	Offered security level	Rho complexity (ρ-value)	Embedding degree (k)	Trace (t)	Discriminant (D)	Curve order (N)	Co-factor (h)	Non-anomalous?	Non-supersingular?
KG256r1	128	127.8	1056598764504768070153408279638907619765443133256637070762399234741213598 71	435735660687254672636395813069964172673	–232773939807348908504365875364485420437302456081553631250351030754382165243	1056598764504768070153408279638907619765443133256637070762399234741213598 71 ($N > 2^{256}$)	1	Yes	Yes
KG384r1	192	191.6	3085049365668014934007996642175611388879720170590096638184143875468390039007764173232556555487299607397910376591752273	–1150667795097978901029351544819860472604711539260496758679	–12207793825204495300330233147726211104554029829927838928931279744644290302463031293456607066435943911501375652123116 3	3085049365668014934007996642175611388879720170590096638184143875468390039007764173232556555487299607397910376591752273 ($N > 2^{384}$)	1	Yes	Yes

Table 5. Verification result of the ECC security of the proposed elliptic curves

Twist of elliptic curve $\mathbb{E}'$	Offered security level in bits	Rho complexity (ρ'-value)	Embedding degree (k')	Curve order (N')	Co-factor (h')	Non-anomalous?	Non-supersingular?
KG256r1	128	127.8	44024948521032002923058678318287817490589910269599283378196677925360 35404384	1056598764504768070153408279638907619765464703828010767202702088404970 5217 ($N' > 2^{256}$)	1	Yes	Yes
KG384r1	192	191.6	30850493656680149340079966421756113888797201710590096638181839134190374190937043227555862047591515205086455602524492400 5373	30850493656680149340079966421756113888797201705900966381839137419093704432275586204759151520508645560252449240053 73 ($N' > 2^{384}$)	1	Yes	Yes

7.2.3. Validation of Trusted Security Criteria T3

To validate the T3 criterion, we conducted an experiment by taking three trials of executing Algorithm 1 under the same operational environment with same method and apparatus to retrieve three independent elliptic curves of the same prime lengths. Subsequently, we examined if they exhibit nearly the same cryptographic strength measured in terms of Pollard's rho value for the curves and their respective twists as discussed in Section 5.2. Table 6 shows the results obtained from this experiment which proves the successful validation of T3 criterion by the proposed KG256r1 and KG384r1 elliptic curves.

Table 6. Validation of Trusted Security criteria of three new elliptic curves: T3

Trial number	Elliptic curve parameters	Pollard's rho value/Twist rho value
1	*p*: 87052253706622316800662279631344302713612816742118516445715106163825624186987	Rho: $2^{127.6}$ Twist rho: $2^{127.6}$
	a: 17461513680488110202189680065467433355982187313809984308530183605390654503146	
	b: 47423645344793070876962443040716664351751669315369958110810670226406616322940	
	$G_{x,y}$: (345624448642634477922898816667823681998089127518316633864441350836419706701 03, 44973717098200324632781286735408077067884851416905001940895476727480258436423)	
2	*p*: 83857931886285555818472058950522827195247211639379970952195176566538052148959	Rho: $2^{127.6}$ Twist rho: $2^{127.6}$
	a: 15222031410359054028041793088708374885174581007053672026416069700422500171995	
	b: 75723663712830868158926603330488486312788754915163584116380630010872983931491	
	$G_{x,y}$: (79991145613299850861660922601873046504314421039422310330231620709939495217575, 74048930300595054686355764380599730714484651315014966555673263252180995491420)	
3	*p*: 11545517368364733676669519855538661606218595740007470090246539865076961715338 3	Rho: $2^{127.8}$ Twist rho: $2^{127.8}$
	a: 89247089594531861167221907824679361896477781827771349654639873760799894221702	
	b: 47456080838438598020722031163435824555796019933240946112077132887442648196 18	
	$G_{x,y}$: (87380972861908942926601892812209714038534482432156502027178728221855540030831, 10901022470361027580777699966258739901041560575689220765054078354933206914768 7)	

8. Results and discussion

The proposed elliptic curves KG256r1 and KG384r1 are compared with other similar standard Short Weierstrass elliptic curves like NIST, SEC2, Brainpool, FRP256v1 and NUMS curves from ECDLP security, ECC security and trusted security perspectives in this section.

8.1. Comparison of the proposed KG256r1 and KG384r1 elliptic curves with standard elliptic curves from ECDLP and ECC security perspectives

It is imperative to note from Table 7 that none of the standard elliptic curves have passed all the SafeCurves verification criteria [1] of ECDLP security and ECC security. However, Brainpool recommended elliptic curves have deviated in their own stipulated procedure of generation [2] and hence cannot be trusted easily. Also, their verifiably random generation method is under question as such thing can be

intentionally implanted to manipulate the standard as demonstrated by Bernstein et. al. through BADA55 curves [2].

Table 7. Comparison of ECDLP Security and ECC Security of the standard elliptic curves and the proposed elliptic curves [1]

Verification criterion	Details	Supported by the curve
SafeField	Prime of the forms 1 mod 4 and 3 mod 4	A, B, C, D1, KG256r1, KG384r1
safeEquation	Elliptic curve over prime field possessing either Short Weierstrass or Montgomery or Edward equation	A, B, C, D1, KG256r1, KG384r1
safeBase	Possessing prime order of base point	A, B, C, D1, KG256r1, KG384r1
safeRho	Rho value must be $\geq 2^{100}$	A, B, C, D1, KG256r1, KG384r1
safeTransfer	Resistant to Smart-ASS attack (additive transfer) and MOV attack (multiplicative transfer)	A, B, C, D1, KG256r1, KG384r1
safeDiscriminant	Absolute value of complex-multiplication field discriminant $\|D\| > 2^{100}$	A, B, D1, KG256r1, KG384r1
safeRigid	Allows only fully rigid and somewhat rigid curves	B, C, KG256r1, KG384r1
safeTwist	Above security requirements for twist of the curve as well	C, KG256r1, KG384r1
safeCurve	Elliptic curve is safe if all the above criteria are met	KG256r1, KG384r1

Note: A = NIST recommended elliptic curves, B = Brainpool recommended elliptic curves, C = SEC2 elliptic curves, D1 = ANSSI recommended elliptic curve FRP256v1.

8.2. Comparison of cryptographic security of the proposed KG256r1 and KG384r1 with standard elliptic curves

Table 8. Comparative security evaluation of the proposed elliptic curves with the standard elliptic curves

Elliptic curve	ECDLP security	ECC security	Trusted security (T1, T2, T3)	Remarks
NIST P224r1	Yes	No	No	No RNG description. Pre-studied value of coefficient a and special structure of prime p in Mersenne form. Weak twist security [3]
NIST P256r1	Yes	No	No	No RNG description. Pre-studied value of coefficient a and special structure of prime p in Mersenne form. Weak twist security [3]
NIST P384r1	Yes	No	No	No RNG description. Pre-studied value of coefficient a and special structure of prime p in Mersenne form. Weak twist security [3]
SEC2 prime curves	Yes	No	No	Special structure of prime p (Mersenne prime) and insufficient documentation [5]
Brainpool curves	Yes	No	No	None of the Brainpool curves are generated by their own stipulated procedure [2]
ANSSI FRP256v1 curve	Yes	No	No	Pre-studied value of coefficient a and insufficient documentation [2]
NUMS curve	Yes	No	No	Deterministic approach with pre-studied coefficients and prime [18]
KG256r1	Yes	Yes	Yes	Randomly generated curve parameters with no pre-studied value. User trusted RNG to minimize the risk of manipulation
KG384r1	Yes	Yes	Yes	Randomly generated curve parameters with no pre-studied value. User trusted RNG to minimize the risk of manipulation

The proposed elliptic curves KG256r1 and KG384r1 are compared with standard Short Weierstrass elliptic curves from overall security of ECDLP, ECC and trust perspectives in Table 8.

We observe from Table 8 that only the proposed KG256r1 and KG384r1 elliptic curves are secure from ECDLP, ECC and trust perspectives and standard elliptic curves have met the ECDLP security validations only.

8.3. Performance of the proposed elliptic curves

The proposed KG256r1 and KG384r1 elliptic curves demonstrated with cryptographic operations like key pair generation, signing and verification on desktop machine having x86_64 with Intel(R) Core(TM) i5-10400 CPU with 2.90GHz processor, 16GB DDR4 memory using GNU/Linux version 5.4.0-58-generic and Python Version 3.8.5 software library. Table 9 shows the performance metrics of the proposed elliptic curves in cryptographic implementations such as key pair generation, signing and verification. The values indicated are the average of 10,000 trials' outcomes.

Table 9. Performance of the proposed elliptic curves in cryptographic implementations

Proposed elliptic curve	Key pair generation		Signing		Verification	
	Time elapsed (in s)	Number of CPU clock cycles used	Time elapsed (in s)	Number of CPU clock cycles used	Time elapsed (in s)	Number of CPU clock cycles used
KG256r1	0.021468	62,260,026	0.0215207	62,410,198	0.0426380	123,650,476
KG384r1	0.035866	104,012,382	0.035838	103,931,139	0.106852	309,871,025

9. Conclusion and future directions

Three new trusted security acceptance criteria T1, T2, T3 are proposed to compute cryptographically safe elliptic curves over the prime fields. These trusted security acceptance criteria or simply, the trusted security criteria are invoked along with the ECDLP security and ECC security in order to minimize the scope of manipulation in the curve parameters due to some (intentionally) non-disclosed property or methods exhibited by their proposers and sabotaged standards. We also discussed in detail that only the randomly drawn curve parameters will have the trust factor where a user trusted strong RNG plays a critical role. The choice of selection of RNG is left with the users who will own the risks associated with his chosen RNG to generate the seed and randomness for curve parameters generation requirements. We also computed two new elliptic curves called KG256r1 and KG384r1 after validating them through the newly proposed trusted security acceptance criteria along with the ECDLP and ECC security validations. Furthermore, we experimentally proved that if elliptic curves are generated keeping these three security notions into consideration then they would have nearly the same cryptographic strength in terms of Pollard's rho

complexity and trustworthiness or suitability. Hence, it is inferred that one must verify trusted security acceptance criteria for randomly generated elliptic curves in addition to ECDLP and ECC security validations for secure implementation of elliptic curve based cryptosystems.

The proposed argument of trusted security and demonstrated KG256r1 and KG384r1 elliptic curves gives the feasibility of future standardization of such randomly generated elliptic curves for trusted cryptographic implementations.

***Acknowledgements*:** The authors would like to thank Society for Electronic Transactions and Security (SETS) and Dr. N. Sarat Chandra Babu, Executive Director, SETS for giving the opportunity to conduct the research and write this article. Authors would also like to sincerely thank Dr. Ananda Mohan P. V. and Dr. Reshmi T. R. for their valuable suggestions and Mr. Santhosh Kumar T. for his help in experimentation.

References

1. Bernstein, D. J., T. Lange. SafeCurves: Choosing Safe Curves for Elliptic-Curve Cryptography. Accessed 31 January 2021. **https://safecurves.cr.yp.to**
2. Bernstein, D. J., T. Chou, C. Chuengsatiansup, A. Hülsing, E. Lambooij, T. Lange, R. Niederhagen, C. van Vredendaal. How to Manipulate Curve Standards: A White Paper for the Black Hat. – In: International Conference on Research in Security Standardisation, Springer, Cham, 15 December 2015, pp. 109-139. **http://bada55.cr.yp.to**
3. National Institute for Standards and Technology. FIPS PUB 186-2: Digital Signature Standard, 2000. Accessed 31 January 2021. **http://csrc.nist.gov/publications/fips/archive/fips186-2/fips186-2.pdf.**
4. Lochter, M., J. Merkle. RFC 5639: Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation. 2010. Accessed 31 January 2021. **https://tools. ietf.org/html/rfc5639**
5. Certicom Research. SEC 2: Recommended Elliptic Curve Domain Parameters. Version 1.0. 2000. Accessed 31 January 2021. **http://www.secg.org/SEC2-Ver-1.0.pdf**
6. Institute of Electrical and Electronics Engineers. IEEE 1363-2000: Standard Specifications for Public Key Cryptography, 2000. Accessed 31 January 2021. **http: //grouper.ieee.org/groups/1363/P1363/draft.html**
7. Koblitz, A. H., N. Koblitz, A. Menezes. Elliptic Curve Cryptography: The Serpentine Course of a Paradigm Shift. – Journal of Number Theory, Vol. **131**, 2011, No 5, pp. 781-814.
8. Savaş, E., T. A. Schmidt, C. K. Koç. Generating Elliptic Curves of Prime Order. – In: International Workshop on Cryptographic Hardware and Embedded Systems, Berlin, Heidelberg, Springer, May 2001, pp. 142-158.
9. Valenta, L., N. Sullivan, A. Sanso, N. Heninger. In Search of CurveSwap: Measuring Elliptic Curve Implementations in the Wild. – In: 2018 IEEE European Symposium on Security and Privacy (EuroS&P), IEEE, April 2018, pp. 384-398.
10. Caelli, W. J., E. P. Dawson, S. A. Rea. PKI, Elliptic Curve Cryptography, and Digital Signatures. – Computers & Security, Vol. **18**, 1999, No 1, pp. 47-66.
11. Shumow, D., N. Ferguson. On the Possibility of a Back Door in the NIST sp800-90 Dual Ec Prng. – In: Proc. Crypto, Vol. 7, 2007.
12. Hales, T. C. The NSA Back Door to NIST. – Notices of the AMS, Vol. **61**, 2013, No 2, pp. 190-192.
13. Bernstein, D. J., T. Lange. Security Dangers of the NIST Curves. – In: Invited Talk, International State of the Art Cryptography Workshop, Athens, Greece, 2013.

14. K o c, C. K. About Cryptographic Engineering. – In: Cryptographic Engineering, Boston, MA, Springer, 2009, pp. 1-4.
15. S c h n e i e r, B. Applied Cryptography: Protocols, Algorithms, and Source Code in C. Second Edition. John Wiley & Sons, 2007.
16. Agence nationale de la s´ecurit´e des syst`emes d'information. Publication d'un param´etrage de courbe elliptique visant des applications de passeport ´electronique et de l'administration ´electronique fran¸caise, 2011.
https://tinyurl.com/nhog26h
17. B o s, J. W., C. C o s t e l l o, P. L o n g a, M. N a e h r i g. Selecting Elliptic Curves for Cryptography: An Efficiency and Security Analysis. – Journal of Cryptographic Engineering, 2015, pp. 1-28.
18. C o s t e l l o, C., P. L o n g a, M. N a e h r i g. A Brief Discussion on Selecting New Elliptic Curves. Microsoft Research. Microsoft. 8 Jun 2015.
19. B o s, J. W., C. C o s t e l l o, P. L o n g a, M. N a e h r i g. Selecting Elliptic Curves for Cryptography: An Efficiency and Security Analysis – Journal of Cryptographic Engineering, Vol. **6**, November 2016, No 4, pp. 259-286.
20. C h e n g, Q. Hard Problems of Algebraic Geometry Codes. – IEEE Transactions on Information Theory, Vol. **54**, 2008, No 1, pp. 402-406.
21. M c I v o r, C. J., M. M c L o o n e, J. V. M c C a n n y. Hardware Elliptic Curve Cryptographic Processor Over rmGF(p). – IEEE Transactions on Circuits and Systems, Vol. **53**, 2006, No 9, pp. 1946-1957.
22. S c h o o f, R. Elliptic Curves Over Finite Fields and the Computation of Square Roots mod p. – Mathematics of Computation, Vol. **44**, 1985, No 170, pp. 483-494.
23. H a n k e r s o n, D., A. M e n e z e s, S. V a n s t o n e. Guide to Elliptic Curve Cryptography. Springer, 2003. 332 p. (web). ISBN: 0-387-95273-X.
24. M e n e z e s, A. J., T. O k a m o t o, S. A. V a n s t o n e. Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field. – IEEE Transactions on Information Theory, Vol. **39**, 1993, No 5, pp. 1639-1646.
25. S m a r t, N. P. The Discrete Logarithm Problem on Elliptic Curves of Trace One. – Journal of Cryptology, Vol. **12**, 1999, No 3, pp.193-196.
26. V a n O o r s c h o t, P., M. W i e n e r. Parallel Collision Search with Cryptanalytic Applications. – Journal of Cryptology, Vol. **12**, 1999, pp. 1-28.
27. V i e g a, J. Practical Random Number Generation in Software. – In: Proc. of 19th Annual Computer Security Applications Conference, 2003, IEEE, 8 December 2003, pp. 129-140.
28. D o d i s, Y., D. P o i n t c h e v a l, S. R u h a u l t, D. V e r g n i a u d, D. W i c h s. Security Analysis of Pseudo-Random Number Generators with Input: /Dev/Random is not Robust. – In: Proc. of 2013 ACM SIGSAC Conference on Computer & Communications Security, 4 November 2013, pp. 647-658.
29. G u t t e r m a n, Z., B. P i n k a s, T. R e i n m a n. Analysis of the Linux Random Number Generator. – In: 2006 IEEE Symposium on Security and Privacy (S&P'06), IEEE, 21 May 2006, pp. 15-32.
30. **https://www.2uo.de/myths-about-urandom/**

Received: 02.02.2021; Accepted: 05.03.2021 (fast track)