PEC - AN ALTERNATE AND MORE EFFICIENT PUBLIC KEY CRYPTOSYSTEM

Rahul Krishnan

Mass Academy of Math and Science
85 Prescott Street, Worcester, MA 99999, USA
E-mail: rkrishnan3@wpi.edu.

ABSTRACT

In an increasingly connected world, security is a top concern for Internet of Things (IoT). These IoT devices have to be inexpensive implying that they will be constrained in storage and computing resources. In order to secure such devices, highly efficient public key cryptosystems (PKC) are critical. Elliptic Curve Cryptography (ECC) is the most commonly implemented PKC in use today. In this paper, an alternate and a more efficient PKC, called the PEC (Pells Equation Cryptography) has been proposed based on Pells equation: $x^2 - D * y^2 \equiv 1$ (mod p). It is shown that scalar multiplication in PEC is significantly more efficient compared to ECC. It is also shown that the Discrete Logarithm Problem - computing the private key from the public key - in PEC is at least as hard as that of ECC.

KEYWORDS

Public Key Cryptography, Elliptic Curve Cryptography, Scalar Multiplication, Pell's equation, Internet of Things

1 INTRODUCTION

Efficient and secure public-key cryptosystems (PKC) are critical to protect Internet communications. Recently, there have been several instances of malware infecting thousands of cheap IoT (Internet of Things) devices such as IP security cameras and thereby flooding the Internet with traffic and shutting down a lot of popular websites. In an increasingly connected future, such attacks could result in shutting down our water and electricity supplies, disabling security systems and even loss of life when medical devices are impacted. As a result, security is a top concern for IoT devices, almost as essential as affordability. The fact that these IoT devices have to be cheap implies that they can have only limited storage and computing resources. In order to secure and authenticate such devices, highly efficient public key cryptosystems that use a combination of public and private keys are critical. Public-key cryptography, or asymmetric cryptography, is an encryption scheme that uses two mathematically related, but not identical keys a public key and a private key [Hel02]. Unlike symmetric key algorithms that rely on one key to both encrypt and decrypt, each key in public key cryptography performs a unique function. The public key is used to encrypt plaintext whereas a private key is used to decrypt cipher text. The effectiveness of this cryptographic system is based on the fact that it is computationally infeasible to compute the private key from the public key. Consequently, public keys can be freely shared, allowing users an easy and convenient method for encrypting content and verifying digital signatures, and private keys can be kept secret, thereby ensuring that only the owners of the

DOI:10.5121/ijcis.2018.8101

private keys can decrypt content.

Elliptic Curve Cryptography (ECC) has emerged as one of the popular cryptosystems after Miller and Koblitz independently introduced elliptic curves into cryptography [Kob87]. The security framework of most of the modern Internet is based on ECC, not to mention that of Bitcoin and other cryptocurrencies. The biggest differentiator between ECC and RSA (another well-known public key cryptosystem) is key size compared to cryptographic strength. ECC can offer the same level of cryptographic strength at much smaller key sizes, thereby offering improved security with reduced computational requirement [Kob87]. In [Pad06], an RSA type public key cryptosystem based on Pell's equation is proposed and compared against the RSA scheme. In [RALB15], a cryptosystem based on Pell's equation with Jacobi symbol is proposed along with an efficient implementation. In this paper, we present PEC - a public key cryptosystem based on Pell's equation in a finite field and compare the scalar multiplication performance between PEC and ECC. We show that the scalar multiplication operation in PEC is significantly more efficient and faster than that of ECC for different values of k, the scalar multiplier and p, the prime field. To our knowledge, no prior work exists that shows this result.

The paper is organized as follows: Section 2 provides an overview of ECC in a finite field, including scalar multiplication and the discrete logarithm problem. Section 3 introduces Pell's equation and proves how the solution space to Pell's equation in a finite field forms a closed abelian group. In this section, we prove certain properties of scalar multiplication in PEC and also discuss how points in the group form a closed cyclic subgroup. In addition to describing an approach to find the order of such a subgroup, we also outline an algorithm to choose an optimal base point. Section 4 describes both the theoretical as well as simulation results showing the performance difference between ECC and PEC scalar multiplication algorithms. Conclusions and some directions for future research are presented in Section 5.

2 ELLIPTIC CURVE CRYPTOGRAPHY

Elliptic curves belong to a class of cubic equations of the form $y^2 = x^3 + Ax + B$, where A and B are integer coefficients and $4A^3 + 27B^2 \neq 0$. This equation is the Weierstrass normal form for elliptic curves. The interesting property of elliptic curves is that points ($P(x, y)$) on the curve form a closed abelian group [ST92]. The inverse of a point $P(x, y)$ is the one symmetric about the x axis, $P'(x, -y)$. The identity element is the point at infinity $\mathcal{O}$. The binary operation represents the addition of two points on the curve resulting in a third point on the curve. This third solution can be generated either through a geometrical construction or algebraically using polynomial equations [ST92].

Most ECC implementations are based on elliptic curves restricted to finite fields [Lu04] [SK12]. A finite field, denoted as F_p, is a set with a finite number of elements. An example of a finite field is the set of integers modulo p, where p is a prime number. This set consists of all the integers from 0 to $p - 1$. Let $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ be two points on the elliptic curve E defined by $y^2 = x^3 + Ax + B$. A new point $R(x_3, y_3) = P \oplus Q$ is given by equation (1)

$$x_3 = m^2 - x_1 - x_2 \pmod{p} \tag{1}$$

$$y_3 = m(x_1 - x_3) - y_1 \pmod{p}$$

$$m = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} & \text{if } P \neq Q \\ \\ \frac{3x_1^2 + A}{2y_1} & \text{if } P = Q \end{cases}$$

An elliptic curve defined over a finite field has a finite number of points that satisfy the elliptic curve equation, called the order of the group. The interesting result is that elliptic curves in F_p still form a closed abelian group.

One of the most important operations in ECC is scalar multiplication which is used to generate the public key. This term refers to the operation of multiplying an integer by a point on the elliptic curve as shown in the equation below:

$$Q = kP = \underbrace{P + P + \cdots + P}_{k \text{ times}} \tag{2}$$

In this equation, both P and Q are points on the elliptic curve and k is typically a large integer. The underlying challenge is that in a real world scenario, the integer multiplying the point is very large, so the ability to do the multiplication efficiently and in a high-performing fashion is critical [BL07]. This is especially important if the end devices have limited storage and compute resources such as the emerging Internet of Things (IoT) devices such as IP cameras and medical sensors. Existing ECC implementations support operations for doubling a point and adding two points. Thus ECC scalar multiplication operation is typically done by doing a series of point doublings and additions until the required point is multiplied by the large integer. One of the best known algorithms is the Binary Double-Add algorithm (BDA) which is also referred to as binary exponentiation [Iye12] [KAT06] and [GV11]. This algorithm works by converting the integer into binary form. The algorithm then traverses the binary representation of the integer and decides to do a Double or Add based on whether the value of the rightmost bit in the representation is a 1 or a 0. If doubling and adding are both $O(1)$ operations, then the BDA algorithm is $O(\log k)$ or $O(m)$ where there are m digits in the binary representation of k.

Given k and P, the BDA algorithm provides a polynomial time algorithm for computing $Q = kP$. In ECC, the non-secret parameters E, p, P, and n are first chosen and an integer k is chosen randomly from 1 to n. In this case, k is the private key of the sender and Q is the public key. Q is transmitted over a potentially insecure channel where the messages could be intercepted. The problem of finding k if P and Q are known is the discrete logarithm problem (DLP) for elliptic curves restricted to finite fields. This problem is believed to be a **hard** problem in that there is no known polynomial time algorithm that can solve this problem [Men12]. This problem is referred to as discrete because they involve finite sets and cyclic subgroups. Clearly, if EC-DLP is easy, then a Man in the Middle can deduce k from Q. Therefore, the hardness of EC-DLP is critical for the security of ECC. Although no proof exists that the EC-DLP is a hard problem, the problem has been extensively studied by researchers for the last 16 years and no general-purpose sub-exponential time algorithm has been discovered. [Men12] surveys the research work done on EC-DLP.

3 PELL'S EQUATION CRYPTOGRAPHY

Pell's Equation Cryptography (PEC), an alternate and a more efficient public key cryptosystem, based on the solution space to the Pells equation in a prime field has been studied in this paper.

3.1 Pell's Equation

Pells equation is any quadratic Diophantine equation of the form $x^2 - Dy^2 = 1$, where D is a positive non-square integer, with integer solutions for x and y. Pell's equation has an infinity of positive integral solutions for x and y. The infinite solutions can be computed from the minimal positive solution (also called fundamental solution) in a recursive fashion [Kos14]. If (x_1, y_1) is the solution with smallest x_1, then every solution (x_k, y_k) can be obtained by taking powers as follows:

$$(x_1 + y_1\sqrt{D})^k = x_k + y_k\sqrt{D} \quad \text{for } k = 1, 2, 3, .. \tag{3}$$

where $x_k = \frac{1}{2}\Big((a + b\sqrt{D})^k + (a - b\sqrt{D})^k\Big)$ and $y_k = \frac{1}{2\sqrt{D}}\Big((a + b\sqrt{D})^k - (a - b\sqrt{D})^k\Big)$. The solutions can also be derived in a recursive fashion. For example, if (x_1, y_1) and (x_2, y_2) are both solutions to the Pell's equation $x^2 - Dy^2 = 1$, another solution (x_3, y_3) can be generated as follows:

$$(x_3, y_3) = (x_1, y_1) \oplus (x_2, y_2) = (x_1x_2 + Dy_1y_2, x_1y_2 + x_2y_1) \tag{4}$$

where $\oplus$ can be treated as a binary operation. Generation of the fundamental solution to Pells equation can be used to construct infinitely many solutions. Considering a simple example: $x^2 - 2y^2 = 1$ where (3,2) is the fundamental solution to this Pells equation. Other solutions can be recursively obtained as follows:

$$(x_2, y_2) = (3, 2) \oplus (3, 2) = (3.3 + 2.2.2, 3.2 + 2.3) = (17, 12)$$

$$(x_3, y_3) = (17, 12) \oplus (3, 2) = (99, 70)$$

Through the $\oplus$ operation and the fundamental solution (3,2), it can be seen that infinitely many solutions can be generated.

3.2 Pell's Equation in a Finite Field

The goal is to design a more efficient public key cryptosystem compared to that of ECC. In this section we will prove that the solution space to the Pell's equation represented as (x, y) forms a closed abelian group under a binary operation. In addition to proving that the binary operation is associative and commutative, proof of existence of an identity element and an inverse element for every element in the solution space is necessary for this step. This forms the basis of the Pell's Equation Cryptosystem (PEC). Similar to elliptic curves in a finite field. Pell's equation in a finite prime field can be represented as follows:

$$x^2 - Dy^2 \equiv 1 \pmod{p} \tag{5}$$

where p is a prime number. A finite field is a set with a finite number of elements. An example of a finite field is the set of integers modulo p, where p is a prime number. It is generally denoted as GF(p) or $\mathbb{F}_p$. Let C_p represent the set of solutions $(x, y) \in \mathbb{F}_p \times \mathbb{F}_p$ to the Pell's equation represented in (5). The solutions (x, y) also belong to the field $\mathbb{Q}(\sqrt{D}) = \{x + y\sqrt{D}; x, y \in \mathbb{F}p\}$.

This solution space (x, y), which was originally

$$\Big\{(x, y) \in \mathbb{Z}^2 \quad | \quad x^2 - Dy^2 = 1\Big\}, \text{ becomes}$$

$$\Big\{(x, y) \in \mathbb{F}_p^2 \quad | \quad x^2 - Dy^2 \equiv 1 \pmod{p}\Big\}$$

A group is a set of elements with a binary operation that satisfies four properties: closure, associativity, existence of an identity element and existence of an inverse element for every element in the group. Additionally, if the group is also commutative, it is called an abelian group. [Rei09] provides a rigorous introduction to basic number theory, rings, fields, groups, algebraic geometry, Pell's equation and elliptic curves. In addition, proof that the binary operation satisfies the **closure** property is also provided in [Rei09]. The basic elements of the proof are outlined in this section.

$$(x_3, y_3) = (x_1, y_1) \oplus (x_2, y_2) = (x_1x_2 + Dy_1y_2, x_1y_2 + x_2y_1) \tag{6}$$

It needs to be shown that (x_3, y_3) belongs to C_p. i.e., $(x_3^2 - Dy_3^2 = 1)$

$$(x_1x_2 + Dy_1y_2)^2 - D * (x_1y_2 + x_2y_1)^2 = x_1^2x_2^2 + D^2y_1^2y_2^2 - Dx_1^2y_2^2 - Dx_2^2y_1^2$$

$$= x_1^2 * (x_2^2 - Dy_2^2) - Dy_1^2(x_2^2 - Dy_2^2) = x_1^2(1) - Dy_1^2(1) = 1$$

Equation (7) shows that the binary operation is **commutative**.

$$(x_1, y_1) \oplus (x_2, y_2) = (x_1x_2 + Dy_1y_2, x_1y_2 + x_2y_1) = (x_2x_1 + Dy_2y_1, x_2y_1 + x_1y_2) = (x_2, y_2) \oplus (x_1, y_1) \tag{7}$$

Equation (8) shows that the binary operation is **associative** as well.

$$\begin{aligned}\Big((x_1, y_1) \oplus (x_2, y_2)\Big) \oplus (x_3, y_3) &= (x_1x_2 + Dy_1y_2, x_1y_2 + x_2y_1) \oplus (x_3, y_3) \\ &= \Big((x_1x_2 + Dy_1y_2) * x_3 + D(x_1y_2 + x_2y_1) * y_3, (x_1x_2 + Dy_1y_2) * y_3 + \\ &x_3 * (x_1y_2 + x_2y_1)\Big) \\ &= (x_1 * (x_2x_3 + Dy_2y_3) + Dy_1(x_2y_3 + x_3y_2), x_1 * (x_2y_3 + x_3y_2) \\ &+ (x_2x_3 + Dy_2y_3) * y_1) \\ &= (x_1, y_1) \oplus \Big((x_2, y_2) \oplus (x_3, y_3)\Big)\end{aligned} \tag{8}$$

$$(x_1, y_1) \oplus (1, 0) = (x_1 * 1 + Dy_1 * 0, x_1 * 0 + 1 * y_1) = (x_1, y_1) \tag{9}$$

$$(x_1, y_1) \oplus (x_1, -y_1) = (x_1^2 - Dy_1^2, x_1(-y_1) + x_1y_1) = (1, 0) \tag{10}$$

Clearly, the trivial solution (1,0) is the **identity element** as shown in equation (9), while equation (10) shows that an **inverse** exists for every element in C_p. ($(x, -y)$ is the inverse of (x, y)). Therefore, the solution space to the Pells equation in a finite field forms a closed abelian group under the operation $\oplus$, similar to the addition of points on elliptic curves.

3.3 Order of Pell's Equation Group

The next step is to show that the scalar multiplication of a point P from this solution space by a large scalar multiplier k results in a point Q that also belongs to the same solution space. In this step, it is necessary to show that irrespective of the initial chose point P, multiples of P always form a closed cyclic subgroup. The number of points in any group is called the order of the group. The order of the group defined by the solution space to the Pell's equation $x^2 - Dy^2 \equiv 1 \pmod p$ is given by $p + \left(\frac{D}{p}\right)$, where $\left(\frac{D}{p}\right)$ is called the Legendre symbol and is equal to $\{-1, 0, +1\}$. Legendre symbol is a number theoretic function defined to be equal to ± 1 depending on whether a is a quadratic residue modulo p and has value 0 if p divides a [Sil01].

Scalar multiplication in PEC is defined in the same way as in ECC. For instance

$$kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$$

The elements in the solution space of the Pell's equation form a field as described in the previous section. Therefore, if $u = (x_1 + \sqrt{D}y_1)$, then from equation (3) the following is true:

$$k \otimes (x_1, y_1) = \underbrace{(x_1, y_1) \oplus (x_1, y_1) \oplus \cdots \oplus (x_1, y_1)}_{k \text{ times}} = \underbrace{u \oplus u \cdots \oplus u}_{k \text{ times}} = u^k \tag{11}$$

Scalar multiplication over this solution space (x, y) in C_p has several interesting properties that can be proved as follows:

Property 1:

$$(a \times b) \otimes (x, y) = a \otimes \big(b \otimes (x, y)\big) = b \otimes \big((a \otimes (x, y)\big) = (b \times a) \otimes (x, y) \tag{12}$$

Proof:

$(a \times b) \otimes (x, y) = (x, y)^{(a \times b)} = \big((x, y)^a\big)^b = \big((x, y)^b\big)^a = (x, y)^{(b \times a)} = (b \times a) \otimes (x, y)$

Property 2:

$$a \otimes \big((x_1, y_1) \oplus (x_2, y_2)\big) = \big(a \otimes (x_1, y_1)\big) \oplus \big(a \otimes (x_2, y_2)\big) \tag{13}$$

Proof:

$$\begin{aligned} a \otimes \big((x_1, y_1) \oplus (x_2, y_2)\big) &= \underbrace{\big((x_1, y_1) \oplus (x_2, y_2)\big) \oplus \big((x_1, y_1) \oplus (x_2, y_2)\big) \oplus \big((x_1, y_1) \oplus (x_2, y_2)\big)}_{a \text{ times}} \\ &= \underbrace{\big((x_1, y_1) \oplus \cdots \oplus (x_1, y_1)\big)}_{a \text{ times}} \oplus \underbrace{\big((x_2, y_2) \oplus \cdots \oplus (x_2, y_2)\big)}_{a \text{ times}} \ (\oplus \text{ is commutative}) \\ &= \big(a \otimes ((x_1, y_1)\big) \oplus \big(a \otimes ((x_2, y_2)\big) \end{aligned}$$

where $\oplus$ is the binary operation and $\otimes$ is the symbol for the scalar multiplication operation. These properties will be used in the later sections when the PEC version of the Diffie-Hellman algorithm is described.

Consider the Pell's equation, $x^2 - Dy^2 \equiv 1 \pmod{19}$ and the point P = (2,7). All the multiples of P are shown in Figure 1: $2P = (7, 9)$; $3P = (7, 10)$; $4P = (2, 12)$; $5P = (1, 0)$ which is the identity element; $6P = (2, 7)$; $7P = (7, 9)$; $8P = (7, 10)$; $9P = (2, 12)$ and $10P = (1, 0)$. It is clear that only five distinct points appear cyclically and the other points representing the other solutions to the Pell's equation do not appear at all. The fact that they repeat cyclically can be represented by the equation

$$kP = (k \bmod 5)P \tag{14}$$

It can also be verified that these five points are closed under addition. Irrespective of how these points are added, the result is always one of these five points. The other points in $\mathbb{F}_p$ never appear in the results. This same holds for every point, not just for P = (2,7). In fact,

$$kP + mP = \underbrace{P + \cdots + P}_{k \text{ times}} + \underbrace{P + \cdots + P}_{m \text{ times}} = (k + m)P$$

which implies that if two multiples of P are added, another multiple of P is generated (i.e. multiples of P are closed under addition). This shows that the set of the multiples of P is a cyclic subgroup of the group

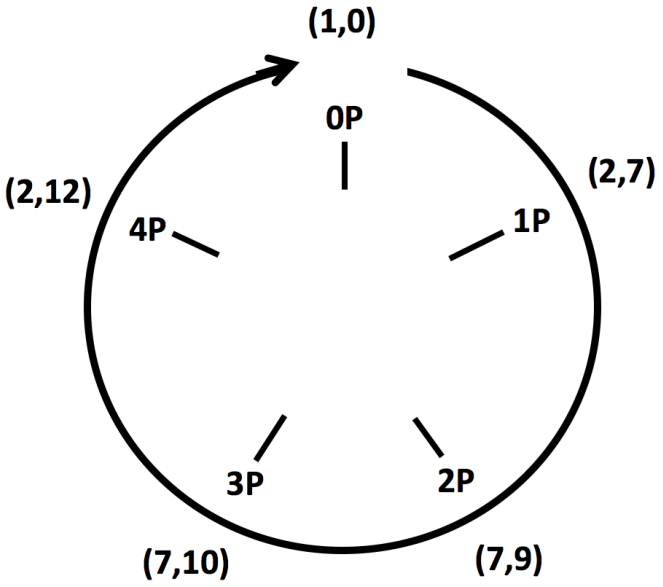

Figure 1: Multiples of $P = (2, 7)$ that are shown repeating in a cyclical fashion.

formed by the solution space to the Pell's equation (C_p). A "cyclic subgroup" is a subgroup where elements are repeating cyclically as shown in this example. The point P is called generator or base point of the cyclic subgroup. Each cyclic subgroup generated by a point P may have an order which is different from the order of the full group represented by the solution space. This leads to the definition of the order of a cyclic subgroup as follows: the order of the subgroup of which P is a member, is the smallest positive integer n such that $nP = 0$. In the previous example, the subgroup contained five points, which implies $5P = 0$.

3.3.1 Order of a Subgroup

For a cryptographically strong PEC, subgroups with a high order need to be chosen. The question becomes - given a point P, what is the order of the subgroup(s) generated by that point? The order of the subgroup of which P is a member is linked to the order of the full group by Lagrange's theorem, which states that for any finite group S, the order (number of elements) of every subgroup H is a divisor of the order of the parent group S [Rot01]. This implies that if the order of C_p is N and one of the subgroups comprises n points, then n is a divisor of N. This theorem provides an approach to determine the order of a subgroup containing the point P.

- Calculate the order N of C_p using $\left(p + \frac{D}{p}\right)$
- Compute all the divisors of N.
- For every divisor n of N, compute nP.
- The smallest n where $nP = 0$ is the order of the subgroup containing the point P.

For example, the Pell's equation $x^2 - 2y^2 \equiv 1 \pmod{61}$ has order 62. The subgroups can have order $n = 1, 2, 31$ or 62. If $P = (17, 49)$, $1.P \neq (1, 0), 2.P \neq (1, 0)$ but $31.P = (1, 0)$, implying that the order of P is 31. On the other hand, if $P = (3, 59)$, $1.P \neq (1, 0), 2.P \neq (1, 0), 31.P \neq (1, 0), 62.P = (1, 0)$, so the order of

P is 62. Yet another example is $x^2 - 2y^2 \equiv 1 \pmod{37}$, whose order is 38. The subgroups can have order $n = 1, 2, 19$ or 38. If $P = (17, 25)$, $1.P \neq (1, 0), 2.P \neq (1, 0)$ but $19.P = (1, 0)$, implying that the order of P is 19. On the other hand, if $P = (3, 35)$, $1.P \neq (1, 0), 2.P \neq (1, 0), 19.P \neq (1, 0)$ while $38.P = (1, 0)$, so the order of P is 38.

3.3.2 Choosing a Base Point

For a strong PEC, subgroups with a high order need to be chosen. The order N of a Pell's equation in a finite field is computed, after which a high divisor is selected as the order n of the subgroup. A base point P belonging to this subgroup is then chosen as the generator.

Lagrange's theorem implies that the number $h = N/n$ is always an integer (because n is a divisor of N). The number h is called the cofactor of the subgroup. For every point P in C_p, $N.P = 0$, given that N is the order of the group. Given the definition of cofactor,

$$n(hP) = 0$$

$$nG = 0$$

The last equation implies that the point G belongs to a subgroup of order n except when $G = hP = (1, 0)$, when n = 1. The following algorithm selects an appropriate base point.

- Select a number n which is prime and a divisor of N. n is the order of the subgroup.
- Calculate the cofactor $h = \frac{N}{n}$.
- Choose a random point P belonging to C_p. Compute $G = hP$. If $G = (1, 0)$, choose another point P. If $G \neq (1, 0)$, G is the base point or the generator belonging to a subgroup of order n and cofactor h.
- n has to be a prime. If n was not a prime, the order of G could be one of the divisors of n.

It is useful to compare the order of PEC with that of ECC for the same prime field. The order of an elliptic curve is given by the following inequality (also known as Schoof's algorithm [Sch85].

$$p + 1 - 2\sqrt{p} \leq (\text{Order of elliptic curve } E \text{ in prime field } p) \leq p + 1 + 2\sqrt{p} \tag{15}$$

Figure 2 shows the order values for both ECC and PEC for different values of p, the prime field. The Sage routine, `E.order()`, was used to determine the group order for ECC. As shown in the figure, the group order values for both ECC and PEC are in the same range for different values of p. This implies that it is possible to choose a generator G belonging to a cyclic subgroup of a very high order in the case of PEC.

3.4 PEC Diffie-Hellman Algorithm

The PEC version of the Diffie-Hellman algorithm will be described to show how keys are exchanged and a shared secret is established between two parties. Scalar multiplication operations are used in this step as well and any efficiency and performance advantage discussed in the results section will be applicable at this stage. PEC Diffie-Hellman algorithm is a variant of the Diffie-Hellman algorithm that defines how keys should be generated and exchanged between the two parties, specifically adapted for PEC. A detailed description of the Diffie-Hellman algorithm is described in [DH76]. Alice and Bob want to exchange information securely,

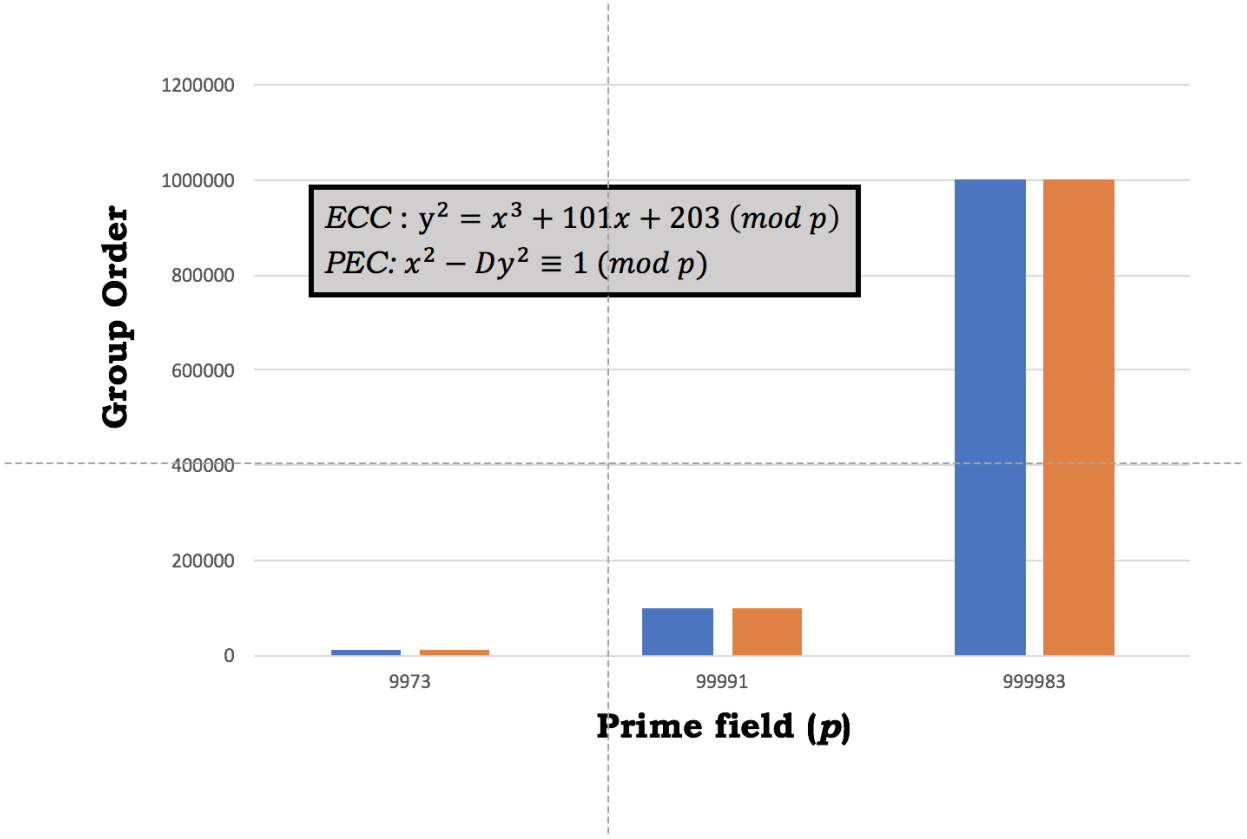

Figure 2: Group order comparison for different values of p.

so that a Man in the Middle may intercept them, but should not be able to decipher the keys. The algorithm (shown in Figure 3) works as follows:

- Firstly, Alice and Bob generate their own private and public keys respectively. Alice generates the private key d_A and the public key $P_A = d_A.G$ whereas Bob generates his private key d_B and public key $P_B = d_B.G$. where G is the generator for the same Pell's equation on the same finite field. Alice and Bob exchange the parameters D and p beforehand.
- Alice and Bob exchange their public keys P_A and P_B over an insecure channel. The Man in the Middle can intercept the public keys P_A and P_B, but will not be able to compute the private keys, d_A or d_B without solving the Discrete Logarithm Problem, which is known to be a non-polynomial time algorithm for large values of p.
- Alice calculates the shared secret $S_A = d_A.P_B$ (using her private key and Bob's public key) while Bob calculates his shared secret $S_B = d_B.P_A$ (using his own private key and Alice's public key). It can be seen that both calculate the same shared secret, because $d_A.(d_B.G) = d_B.(d_A.G)$. (From equation (12) which proves that scalar multiplication in PEC is commutative.)

$$S_A = d_A.P_B = d_A.(d_B.G) = d_B.(d_A.G) = d_B.P_A = S_B$$

- The Man in the Middle knows both public keys P_A and P_B but will not be able to find out the shared secret S because he does not know the private keys of Alice and Bob.

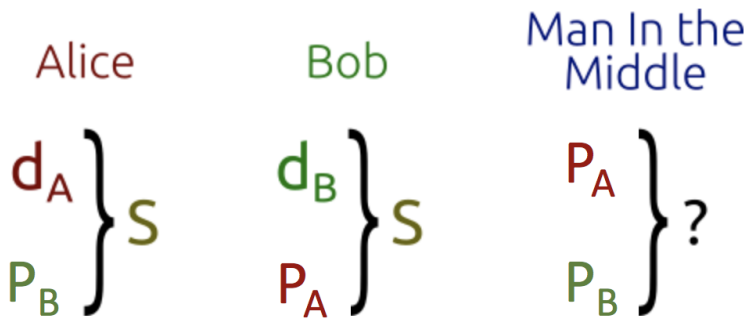

Figure 3: The PEC Diffie-Hellman key exchange: Alice and Bob can easily compute the shared secret. The Man in the Middle has to solve a ”hard” problem.

The PEC Diffie-Hellman algorithm can be stated as follows: given three points, $G, d_A.G$ and $d_B.G$, how can $d_A.(d_B.G)$ or $d_B.(d_A.G)$ can be computed? Given that Alice and Bob have obtained the shared secret, they can now exchange data using symmetric encryption. For example, they can use the x-coordinate of S as the key to encrypt messages using symmetric secure cryptosystems such as AES or 3DES.

4 RESULTS

4.1 Theoretical Results

As described in Section 2, the BDA algorithm is commonly used to implement scalar multiplication in ECC and will be used to benchmark performance between PEC and ECC in this section. For example, let k = 151 whose binary representation is 10010111_2. This can be represented as a sum of powers of two as follows:

$$151 = 1.2^7 + 0.2^6 + 0.2^5 + 1.2^4 + 0.2^3 + 1.2^2 + 1.2^1 + 1.2^0 = 2^7 + 2^4 + 2^2 + 2^1 + 2^0 \tag{16}$$

As a result, $151.P = 2^7.P + 2^4.P + 2^2.P + 2^1.P + 2^0.P$, which can be computed by doubling P seven times and four additions. The elliptic curve formulas for point doubling and point addition, where $P = (x_1, y_1)$ and $Q = (x_2, y_2)$, are as follows:

$$x_3 = m^2 - x_1 - x_2 \pmod p \tag{17}$$

$$y_3 = m(x_1 - x_3) - y_1 \pmod p \tag{18}$$

$$m = \begin{cases} (y_2 - y_1)(x_2 - x_1)^{-1} \pmod p & \text{if } P \neq Q \text{ (Point Addition)} \\ (3x_1^2 + A)(2y_1)^{-1} \pmod p & \text{if } P = Q \text{ (Point Doubling)} \end{cases} \tag{19}$$

On the other hand, the Pell's equation formulas for point doubling and addition are as follows:

$$P + Q = (x_3, y_3) = (x_1x_2 + Dy_1y_2, x_1y_2 + x_2y_1) \pmod p \tag{20}$$

$$P + P = 2P = (2x_1^2 - 1, 2x_1y_1) \pmod p \tag{21}$$

The number of computations required for point doubling and point addition for the case of PEC and ECC are detailed in Table 1, where S = Squaring, M = Multiplication and I = Inversion. According to [BSS99] and [ELM03], inversion is a very expensive operation in finite fields and costs anywhere between 3 and 10

Table 1: Point Doubling and Addition Comparison between PEC and ECC

Type of Cryptosystem	Point Doubling	Point Addition
ECC	2S+ 2M+ 1I = 10M	1S + 2M + 1I = 9M
PEC	1S + 1M =2M	4M

Table 2: Comparison of Scalar Multiplication Operations between PEC and ECC

Type of Cryptosystem	k = 151 (7 doublings 4 additions)	k = 155551 (17 doublings 12 additions)	k=1555555551 (30 doublings 20 additions)
ECC	106M	278M	480M
PEC	30M	82M	140M

multiplications. For this analysis, it is assumed that $I/M = 6$ and 1S = 1M. In addition, additions and subtractions are assumed to have negligible cost compared to multiplications. Table 2 shows the performance difference (in terms of computations) between PEC and ECC for scalar multiplication for different values of k, the scalar multiplier.

4.2 Experimental Results

SageMath, a free open-source mathematical software [S$^+$17] was used to perform operations on elliptic curves and Pell's equation. Sagemath was executed on a MacBook Pro with the following specifications: 2.5GHz Intel Core i5 and 4GB 1600MHz DDR3 memory running OS X El Capitan version 10.11.6. The `timeit()` command based on the Python *timeit* module has been used to determine the best time for executing the scalar multiplication operation for each cryptosystem.

Scalar multiplication results for both PEC and ECC are shown for a variety of input parameters: p, the prime field, k the scalar multiplier, and P the generator point. Each experiment was run 800 times and the best results were averaged to generate the final value. For ECC computation, the values of A and B were varied as well. Tables 3 and 4 show the scalar multiplication performance results of ECC and PEC respectively.

Table 4: PEC Scalar Multiplication Performance Results

p	k	D	P (Generator point)	Average execution time (in μs)
9973	10000	2	(3,9971)	4.14
9973	1000000	2	(3,2)	3.49
99991	10000	3	(7,99987)	4.16
99991	1000000	3	(7,4)	3.45
999983	1000000	5	(9,4)	4.88
999983	10000	5	(9,99979)	4.04
999983	1000000	5	(9,99979)	5.38

Table 3: ECC Scalar Multiplication Performance Results

p	k	$< A, B >$	Average execution time (in μs)
9973	10000	$< 101, 203 >$	55.9
9973	1000000	$< 2828, 5784 >$	793.0
99991	10000	$< 101, 203 >$	55.9
99991	1000000	$< 10191, 30483 >$	539.1
999983	10000	$< 10118, 30337 >$	582.0
999983	1000000	$< 10118, 30337 >$	788.1

Figure 4 compares the performance difference between PEC and ECC for the scalar multiplication operation for different values of p, the prime field and k, the scalar multiplier.

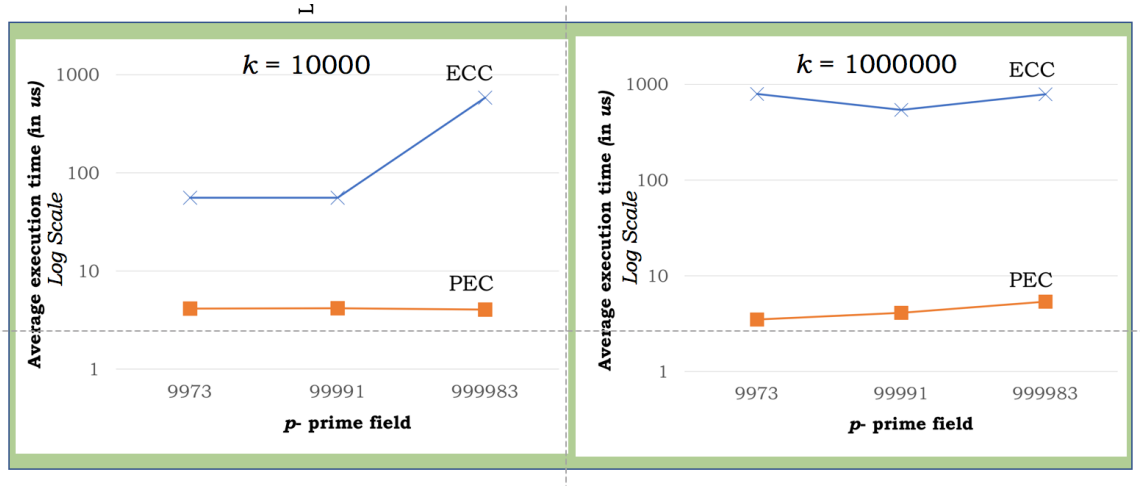

Figure 4: Performance comparison of scalar multiplication. The Y-axis is shown using a log scale.

Both the theoretical as well as the experimental results show that the scalar multiplication operation in PEC is significantly more efficient and faster compared to ECC. The theoretical results did not take into account p, while the experimental results show the effect of varying both p and k. In the case of PEC, the average execution time for the scalar operation is independent of both p and k. However, for ECC the same metric increases with increasing values of both p and k. One reason for the significantly worse performance for ECC in the case of the experimental results could be due to the fact that SageMath may not have implemented the efficient Double-and-Add algorithm for scalar multiplication for elliptic curves and could be using a suboptimal algorithm instead. Another reason is that the theoretical results reflect a first-order approximation and could be understating the penalty for inversion operations. It has been assumed that the I/M is 6 on an average. Based on p and k, it is possible that the ratio is much higher.

5 CONCLUSION

In this paper, Pell's Equation Cryptography (PEC) - an alternate and high-performing public key cryptosystem was designed that is shown to be more efficient than Elliptic Curve Cryptography for the scalar multiplication operation. Both theory as well as Sagemath was used to show the efficiency advantage. PEC-DH, a version of the Diffie-Hellman algorithm for PEC - a key-agreement protocol - is also described in this paper. Finally, it is also shown that the Discrete Logarithm Problem in PEC is at least as "hard" as that of the problem in ECC, given that the order of the PEC and ECC groups are relatively similar. Given that the scalar multiplication operation in PEC is significantly faster than that of ECC, the PEC is a natural fit for securing IoT devices such as IP cameras and medical sensors, which have limited computing power, memory and battery capacity, and therefore require efficient and high-performing key exchange protocols. In conclusion, the cryptosystem designed in this project showed a large improvement in efficiency and speed (at least 3X times) compared to ECC, one of the most commonly used cryptosystems in use. We believe this has significant implications for the field of cryptography.

With regards to future extensions, it would be useful to implement PEC in actual hardware and confirm the efficiency and speed advantage with respect to ECC. This project looked at curves in Weierstrass form. The scalar multiplication comparisons could be extended for elliptic curves in non-Weierstrass form such as Edwards, Hessian and Montgomery curves. Finally, it would be useful to determine the PEC key size that is cryptographically equivalent to a 160-bit ECC key size.

ACKNOWLEDGEMENTS

I would like to acknowledge the thoughtful guidance of the Mass Academy faculty, Mr. Ellis and Mrs. Curran, for their helpful suggestions and pointers. Professor Martin from WPI also suggested new avenues of research direction. I also received some helpful suggestions from Burt Kaliski Jr.

Authors

Rahul Krishnan is currently a high school junior at one of the premier STEM schools in the United States, Mass Academy of Math and Sciences, affiliated with Worcester Polytechnic Institute (WPI). He is interested in mathematics, especially number theory, and computational linguistics. During the summer of 2016, he was awarded a US State Department scholarship to travel to Xiamen, China to learn Mandarin in an immersive environment. He is also a PROMYS alumni.

References

[BL07] D. Bernstein and T. Lange. Faster Addition and Doubling on Elliptic Curves. *Advances in Cryptology*, 13:29–40, 2007.

[BSS99] I.F. Blake, G Seroussi, and Smart.N.P. *Elliptic Curves in Cryptography*. Cambridge University Press, 1999.

[DH76] W Diffie and M. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 22:644–654, 1976.

[ELM03] K. Eisentrager, K. Lauter, and P.L. Montgomery. Fast Elliptic Curve Arithmetic and Improved Weil Pairing Evaluation. *Proceedings of RSA-CT 2003*, pages 343–354, 2003.

[GV11] Joye M. Miyaji A. Rivain M. Goundar, R.R. and A. Venelli. Scalar Multiplication on Weierstrass Elliptic Curves from Co-Z Arithmetic. *Journal of Cryptographic Engineering*, 1(161), 2011.

[Hel02] M. Hellman. An Overview of Public Key Cryptography. *IEEE Communications Magazine*, 40(5):42–49, 2002.

[Iye12] V. Iyengar. Novel Elliptic Curve Scalar Multiplication Algorithms for Faster and Safer Public-Key Cryptosystems. *International Journal on Cryptography and Information Security*, 2(3):57–66, 2012.

[KAT06] D. Kastanis, I.G. Askoxylakis, and A.P. Traganitis. An Efficient Authentication Scheme for Contactless Smartcards using Elliptic Curve Cryptography. *Proceedings of the IASTED International Conference on Communication, Network, and Information Security*, pages 28–33, 2006.

[Kob87] Neal Koblitz. Elliptic Curve Cryptosystems. *Mathematics of Computation*, 48(177):203–209, 1987.

[Kos14] Thomas Koshy. *Pell and Pell-Lucas Numbers with Applications*. Springer New York, 2014.

[Lu04] Yanpu C. Zhengzhong B. Lu, C. An Implementation of Fast Algorithm for Elliptic Curve Cryptosystem over GF(p). *Journal of Electronics (China)*, 21(4):346–352, 2004.

[Men12] A. Menezes. Evaluation of Security Level of Cryptography: The Elliptic Curve Discrete Logarithm Problem ECDLP. *Technical Report*, 2012.

[Pad06] S. Padhye. A Public Key Cryptosystem Based on Pell Equation. *IACR Cryptology ePrint Archive*, 2006.

[RALB15] M.K. Rao, P.S. Avadhani, and P. Lalitha Bhaskhari. Public Key Cryptosystem Based on Pell's Equation with Jacobi Symbol. *Journal of Applied Sciences and Engineering Research*, 4(1):68–76, 2015.

[Rei09] N. Reilly. *Introduction to Applied Algebraic Systems*, volume 1. Oxford University Press, 2009.

[Rot01] R. Roth. A History of Lagrange's Theorem on Groups. *Mathematics Magazine*, 74(2):99–108, 2001.

[S+17] W. A. Stein et al. *Sage Mathematics Software (Version 8.1)*. The Sage Development Team, 2017. `http://www.sagemath.org`.

[Sch85] R. Schoof. Elliptic Curves over Finite Fields and the Computation of Square Roots mod p. *Mathematics of Computation*, 44(170):483–494, 1985.

[Sil01] J.H Silverman. *A Friendly Introduction to Number Theory*. Prentice Hall Inc., 2001.

[SK12] Suneetha C.H. Chandrasekhar A. Sravana Kumar, D. Encryption of Data using Elliptic Curve over Finite Fields. *International Journal of Distributed and Parallel Systems*, 3(1):301–308, 2012.

[ST92] J.H Silverman and J Tate. *Rational Points on Elliptic Curves*. Springer-Verlag New York Inc., 1992.