A COMPARATIVE STUDY OF SAFE CURVES FOR ENHANCED POINT MAPPING IN ELLIPTIC CURVE CRYPTOGRAPHY

Sreekala M

Research Scholar, Department of Computer Applications, Cochin University of Science & Technology, Kochi, Kerala, India & Assistant Professor, Department of Computer Science, Vimala College, Thrissur

sreekalam@vimalacollege.edu.in

Dr. Varghese Paul

Professor, Department of Information Technology, Rajagiri School of Engineering and Technology, Kerala, India

varghesep@rajagiritech.ac.in

Abstract: *This paper selects a set of elliptic curves for point mapping in Elliptic Curve Cryptography and analyses our selection from a performance and security perspective. A collection of elliptic curves suggested by various cryptography standards are chosen, and the chosen curves are checked, concentrating on their security characteristics & performance. Computation time is the main criteria for the performance of each curve. Point mapping in ECC is the implementation point for each curve. The behaviour of different elliptic curves is observed and identified suitable elliptic curves for various computing applications.*

Keywords: ***Elliptic Curve Cryptography, Public Key, Plain Text, Cipher Text, Mapping.***

1. INTRODUCTION

The most important asset of today's world is data, and the most significant resource of the present world is information. Regardless of whether an individual or an association, the need for securing information is inescapable. A few issues in regards to information ruptures are going on around the globe. India has been the second most digital assaults influenced nation between 2016 to 2018, as indicated by the Data Security Council of India (DSCI) report. Cryptography is an art of utilizing science to encode and decode information. The motivation behind cryptography is to shield transmitted data from being perused and comprehended by anybody with the exception of the expected beneficiary. Cryptography incorporates many strategies for scrambling or camouflaging information with the goal that it is accessible just to somebody who can re-establish the information to its unique structure. It involves encryption and decryption. Encryption means converting a message called plaintext to ciphertext, an encrypted form. Decryption means converting back ciphertext to the original message. Cryptography can be broadly defined into two categories – Symmetric and Asymmetric

Cryptography. Symmetric key cryptography is also known as Private key cryptography or secret key cryptography. In symmetric key cryptography, a single key is used for both encryption as well as decryption. Asymmetric key cryptography is also named as public key cryptography or conventional cryptographic system. In asymmetric key cryptography, two keys are used, the public key for encryption and private key for decryption.

2. ELLIPTIC CURVE CRYPTOGRAPHY

Elliptic Curve Cryptography (ECC) is a public key encryption framework reliant on the theory of elliptic curves that can be used to make speedier, more modest, and progressively beneficial cryptographic keys. At the point when a straight line converges a vertical line at a fixed point and pivots about that fixed point, the surface acquired is known as a double right circular cone. It consists of two cones joined at a fixed point called the vertex. If a plane intersects a double right circular cone, we get two-dimensional curves of different conic sections. Depending on the angle made by the plane with the vertical axis of the cone, the plane can cut the cone in three different ways:

a. Ellipse
b. Parabola
c. Hyperbola

When the plane intersects the double right circular cone in such a way that the angle between the plane and the axis is greater than the vertex angle we get a closed curve called an ellipse. When the angle made by the plane to the vertical axis is exactly equal to the vertex angle we get an open curve called a parabola at the intersecting surface of the cone. If the plane intersects the vertical axis at an angle smaller than the vertex angle the plane intersects both the naps of the cone to form an open curve called a hyperbola which has two disjoint curves.

Elliptic curves were first presented for cryptography by Koblitz and Miller as it pursues public key encryption method [1]. Elliptic curve cryptography involves three major operations: key generation, encryption, and decryption. These three activities are especially required to define a substantial cryptosystem. Initially the message is mapped to a point P_m on the curve. That point Pm is then encrypted to obtain a pair of cipher points C_m. It is decrypted back to get the original message point P_m. A generator point G is needed for the key generation operation. Let $E_p(a, b)$ be the curve where a, b are elliptic curve parameters and p is a large prime number. Another large number $n_B(n_B < p)$ is kept as the Private Key. The point $P_B = n_B * G$ is declared as public [2].

Encryption

1. Let r be a random chosen positive integer such that r<p
2. Calculate Cipher point $C_m=\{r*G, P_m + r* P_B\}$
3. Send the pair of Cipher points C_m to receiver

Decryption

1. Multiply the first point with its private key n_B
   $n_B * C1$
2. Subtract above from second point C2
   $C2 - n_B * C1 = P_m + r* P_B - n_B * r*G$
   $= P_m + r* n_B * G - n_B * r*G$
   $= P_m$, the original point encrypted

2.1 Family of Elliptic Curves

The general types of curves used in ECC are Weierstrass Curve, Montgomery Curve and Edward Curve which are listed below [3]:

Table 1 General Forms of Elliptic Curves

Weierstrass Curve	Montgomery Curve	Edward Curve
• $y^2 = x^3 + ax + b$ • general form of the elliptic curve • NIST and Brainpool standards	• $By^2 = x^3 + Ax^2 + x$ • suited for x-coordinate-only addition • used in scalar multiplication	• $ax^2 + y^2 = 1 + dx^2 y^2$ • fastest arithmetic operations

Selection of Elliptic Curves has an important role in Elliptic Curve Cryptography. The majority of the distinctive global norms accessible today concentrate on the choice of protected and productive elliptic curves for implementing ECC. The security of ECC is reliant on the choice of the curve. In the event that a weak curve is chosen for any application, at that point its cryptographic security is decreased proportionally.

The following figure shows the various categories under the family of Elliptic Curves. Prime curves are mainly used in software applications and binary curves are mainly used in hardware applications. Binary curves are smaller and faster when compared prime curves. The reason behind this is binary curves have shorter formulas, and binary operations are easy to perform. Table 1 shows the equation of various types of curves under this family [4].

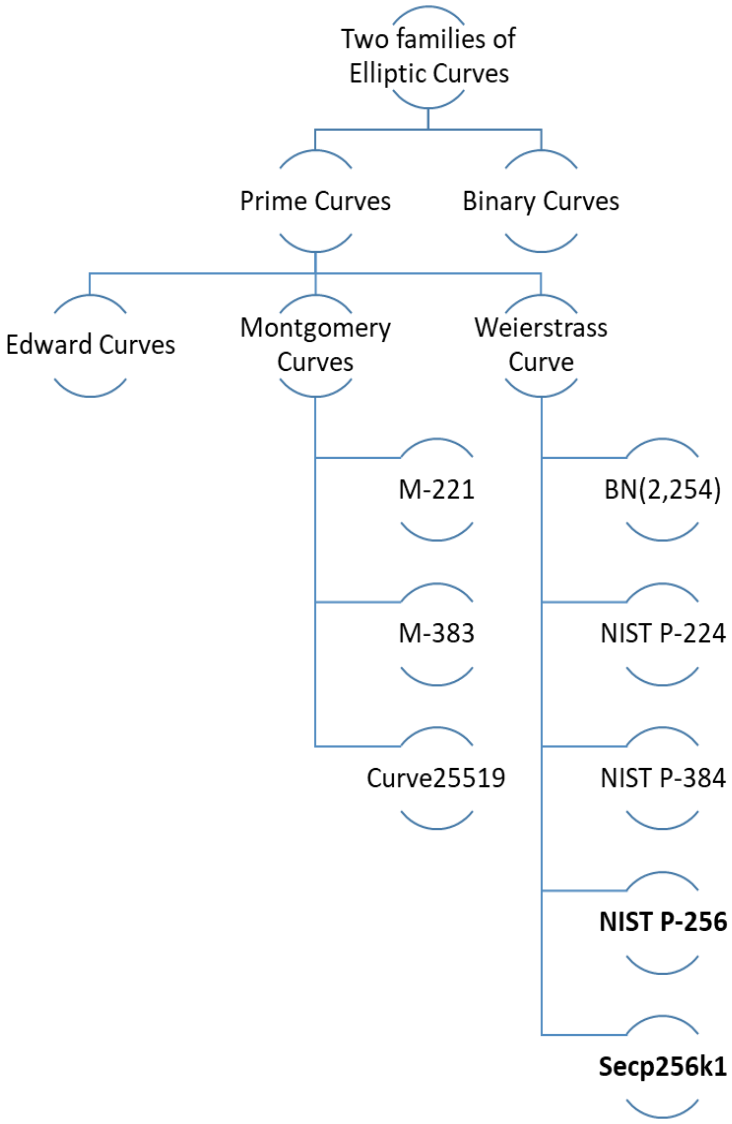

Figure 1 Family of Curves

Table 2 Types of Curve & their Equations

Type of Curve	Equation of the curve
M-221	$y^2 = x^3 + 117050x^2 + x$
M-383	$y^2 = x^3 + 2065150x^2 + x$
P-192	$y^2 = x^3 - 3x + c$ $c = 2455155546008943817740293915197451784769108058161191238$
P-224	$y^2 = x^3 - 3x + c$ $c = 18958286285566608000408668544493926415504680968679321078$
P-384	$y^2 = x^3 - 3x + c$ $c = 27580193559959705877849011840389048093056905856361568521$
P-256	$y^2 = x^3 - 3x + c$ $c = 18958286285566608000408668544493926415504680968679321078$
Curve25519	$y^2 = x^3 + 486662x^2 + x$
secp256k1	$y^2 = x^3 + 7$
BN(2, 254)	$y^2 = x^3 + 2$

2.2 Standards Available for ECC

The list of common standards for the choice of elliptic curve for ECC is given below:

1. ANSI X9.62;
2. IEEE P1363;
3. SEC 2;
4. NIST FIPS 186-2;
5. ANSI X9.63;
6. Brainpool;
7. NSA Suit B;
8. ANSSI FRP 256V1.

Every standard referenced above has a group of curves prescribed for the execution of ECC. Each curve has its particular highlights and points of interest. Choosing the correct curve is the most troublesome part while selecting parameters for the ECC to guarantee hard security of the application and better effectiveness.

2.3 Applications of various types of curves

Various curves are analysed in different papers and their applications and advantages were identified. The following table shows applications and advantages of different types of curves.

Table 3 Advantages & Applications of various Curves

Authors who proposed	Type of Curve	Applications	Advantages
[5] Dang et al., 2020	M-221	IoT, Cloud Computing	It is utilized in an authentication scheme giving secure control from powerful cloud servers to devices while also empowering the direct secure communication interchanges between them.
[6] Jinasena et al., 2017	M-383	Access Control, Authentication	It is reported as one of the latest safe curves It is utilized to execute a dynamic and adaptable access control technique that can guarantee the access privileges of sensitive clinical information without bargaining the quality of any.
[7] Liu et al., 2013	P-192	Elliptic Curve Cryptography (ECC) for wireless sensor nodes	Better performance
[8] Käsper, 2011	P-224	OpenSSL	It is the fastest elliptic curves used in Open SSL
[9] Liu et al., 2018	P-384	Lightweight implementation of public-key cryptography, IoT Security	Memory Efficient Implementation for ECC
[10] Baee et al., 2019	P-256	Authentication in Wireless Networks	To verify low latency secure critical messages in VANET

[11] Düll et al., 2015	Curve25519	Diffie–Hellman key exchange, Bitcoin Security	Bernstein utilizes the Montgomery form of Curve25519 for systematics, secure, and convenient elliptic-curve Diffie–Hellman key exchange. It is also used in the security implementation of Bitcoin.
[12]Chakraborty et al., 2018	secp256k1	Blockchain Security	Appropriate for digital signatures for securing people and machines in blockchain and IoT applications.
[13]Pereira et al., 2010	BN(2, 254)	Pairing-based cryptography	Pairing-based cryptography has been utilized to develop identity-based encryption (IBE), which permits a sender to encrypt a message without using a receiver's public key for certification and distributed before.

2.4 Type of Curves and the number of bits in the prime number P chosen

The graph shows the number of bits in prime number of various curves.

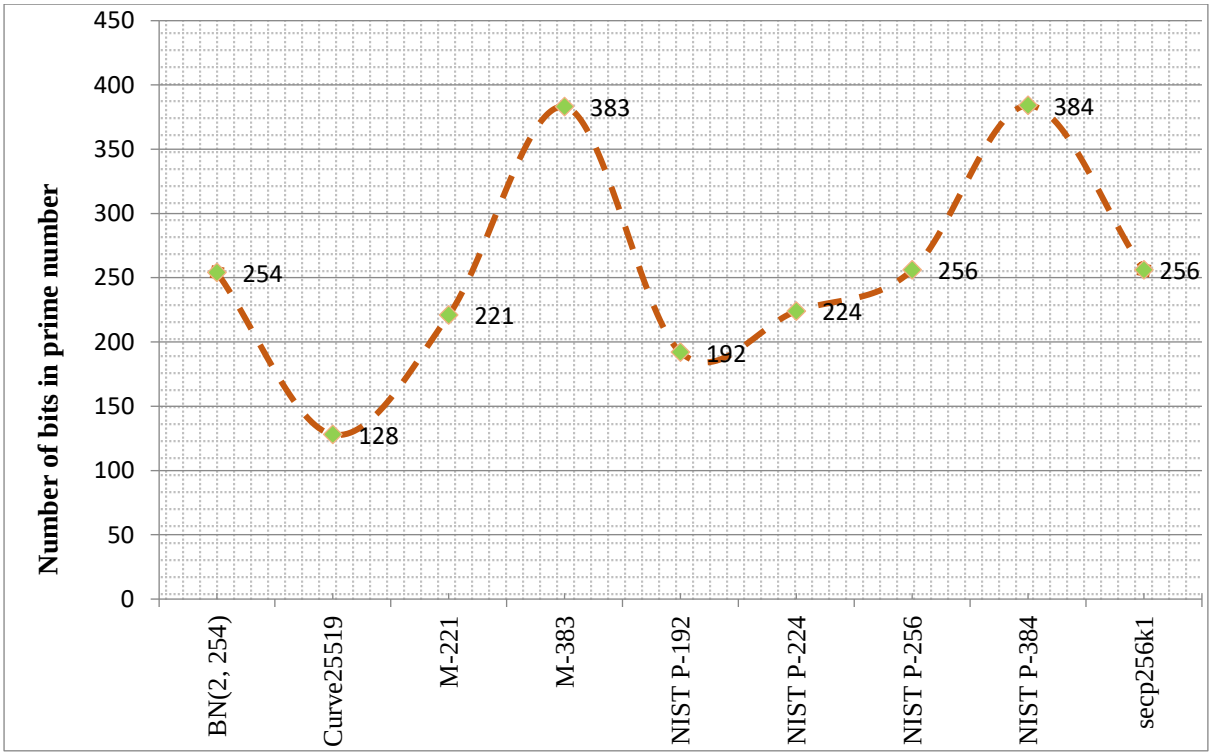

Figure 5 Number of bits in a prime number of various curves

3. GUIDELINES FOR A GOOD MESSAGE MAPPING SCHEME

The following are some of the guidelines for a good message mapping scheme:

1. Encryption using ECC is done by encoding the message to a point on the curve and then encrypting that point to a cipher point. So, the first and foremost requirement for a good encoding scheme is that the mapped points should be on the elliptic curve.
2. The second thing is that reverse mapping should be possible in order to decrypt the cipher points to get back the original message.
3. The next requirement is that an encoding scheme should able to resist different types of vulnerable attacks.
4. Another important guideline for a good mapping scheme is the use of minimum bandwidth while encoding the message.
5. The last, but very important point to be considered while choosing an encoding scheme is time. An efficient mapping scheme should not take too much time for encoding.

4. POINT MAPPING USING VARIOUS CURVES

Here we are generating first 10 points for the elliptic curve $y^2 = x^3 + ax + b$ using various curves.

Input: Let sv be the starting point and type_of_curve be the various ECC curves. Let p be the prime number. We use the Legendre symbol to check if there is a valid square root for y. In number theory, the Legendre symbol is a multiplicative function with values 1, −1, 0 that is a quadratic character modulo an odd prime number p: its value at a (nonzero) quadratic residue mod p is 1 and at a non-quadratic residue (non-residue) is −1. Its value at zero is 0 [8].

Let p be an odd prime number. An integer a is a quadratic residue modulo p if it is congruent to a perfect square modulo p and is a quadratic nonresidue modulo p otherwise. The Legendre symbol is a function of a and p defined as:

$$\left(\frac{a}{p}\right) = \begin{cases} 1 \;\; if \; a \; is \; a \; quadratic \; residue \; modulo \; p \; and \; a \;\not\equiv 0 \; (mod \; p) \\ -1 \;\; if \; a \; is \; a \; quadratic \; residue \; modulo \; p \\ 0 \;\; if \; a \;\equiv 0 \; (mod \; p) \end{cases}$$

Algorithm 1 – Point Mapping (Main function)

1 Let p = 2^256 – 2^224 + 2^192 + 2^96 - 1
2 Let a=-3
3 Let b=4105836372515214212932612978004726840911444101599372555483525631403946
7401291
4 sv=1
5 Check if (len(sys.argv)>1, if no goto step 7
sv=integer part of (sys.argv[1])
6 Check if (len(sys.argv)>2), if no goto step 9
type_of_curve=str(sys.argv[2])
7 Check if (type_of_curve=="P-192"), if no goto step 13
p = 2^192-2^64-1
a=-3
b=2455155546008943817740293915197451784769108058161191238065
8 Check if (type_of_curve=="P-224"), if no goto step 17
b=1895828628556660800040866854449392641550468096867932107578723467256
4
p = 2^224 – 2^96 + 1
a=-3
9 Check if (type_of_curve=="P-256"), if no goto step 21
p = 2^256 – 2^224 + 2^192 + 2^96 - 1
a=-3
b=4105836372515214212932612978004726840911444101599372555483525631403
9467401291
10 Check if (type_of_curve=="P-384"), if no goto step 25
a=-3
b=2758019355995970587784901184038904809305690585636156852142870730198
8689241309860865136260764883745107765439761230575
p = 2^384 – 2^128 – 2^96 + 2^32 - 1
11 Check if (type_of_curve=="Curve25519"), if no goto step 29
a=486662
b=1
p = 2^255 - 19
12 Check if (type_of_curve=="secp256k1"), if no goto step 33
a=0
b=7
p = 2^256 – 2^32 - 977

13 Check if (type_of_curve=="M-221"), if no goto step 37
a=117050
b=1
p = 2^221 - 3
14 Check if (type_of_curve=="BN(2,254)") , if no goto step 41
a=0

b=2
p = 167981087310158322849408041422317339098891871214390698489337154260727 53864723

15 Check if (type_of_curve=="M-383"), if no goto step 45
a=2065150
b=1
p = 2^383 - 187
16 Display type_of_curve
17 Display sv
18 Display p
19 Display a,b
20 Call function findpoint(sv,p,a,b)

Algorithm 1.1 – Point Mapping – Sub function - findpoint(sv,p,a,b)

1. Let x=sv
2. count=0
3. Repeat the following steps
4. Let value=(x^3 + a*x + b) % p
5. return_value= legendre_symbol(value, p)
6. Check if (return_value==1), if no goto step 9
result=modular_sqrt(value, p)
7. Display x and integer part of the result
8. count=count+1
9. x=x+1
10. Check if (count>20), if no goto step 13
Return
11. Check if (x-sv>200), if no goto step 4
Return

Algorithm 1.2 – Point Mapping – Sub function - legendre_symbol(value, p)

1. Compute the Legendre symbol a|p using Euler's criterion.
2. if p divides a, then a|p = 0
3. Returns 1 if a has a square root modulo p
4. -1 otherwise

Algorithm 1.3 – Point Mapping – Sub function - modular_sqrt(value, p)

1. Calculate a quadratic residue (mod p) of 'value', p must be an odd prime.
2. Solve the congruence of the form: x^2 = value (mod p) and returns x.
3. p - x is also a root.
4. 0 is returned is if no square root exists for these value and p.

5. SIMULATION ANALYSIS OF CURVES

The following figures (Fig.6 to Fig 14) illustrate the output of the algorithm 1 discussed above. The simulation of various curves was done in Python 3.7. A total of nine curves are represented as different cases ranging from M-221 to BN(2,254). Here 'sv' represents starting point of the curve and 'type_of_curve' represents various curves.

Case 1: sv=1, type_of_curve = M-221

```
Elliptic curve is:              M-221
Finding elliptic point closest to:      1
Prime number:                   3369993333939829974333376885877453834204643052817571560137951281149
a,b 117050 1
(1, 325102602581685666525384670472018947732917753206292254728152726803 6L)
(2, 148430385032941383956670176418328623244724500419676420265564278993 3L)
(3, 208859598104739870572125540100737613093318783045335120117166220254 8L)
(4, 284766469076543135639786648546059209901106639141996765747354472039 4L)
(6, 634441937963781379756103686440556347008104271366049736519248615962L)
(7, 175277923167716856349284080485801658375521916301162234452842825626L)
(8, 159018098361786514628715429764985227735434434257725264208433823315 3L)
(9, 153380017921619390001124318615539719646465469095355954366867828348 8L)
(10, 236401132760646294938681834182749901546736087473698923368903003473 6L)
(13, 331474287458385923289997603898959543683161152517234269744411219425 3L)
(15, 271763341645044321133691511307256292855567241553163311256338027693 5L)
```

Figure 6 Output of the curve M-221

Case 2: sv=1, type_of_curve = M-383

```
Elliptic curve is:              M-383
Finding elliptic point closest to:      1
Prime number:
197010030981972396061395200500718069025398696352327233339741467021228608857486053057071331274424578204033139951532 21
a,b 2065150 1
(1, 1549440025144539217039923244126812650665333445604495538315712062378550524824985072138579297740659053685143836038673 2L)
(2, 6183578763661543394752303668416902622380896362839918027610145870991579645738804184889246337608577492077342744958295 L)
(3, 9092793228590703863036406160560949042135891556484096884594182426048722570771628951382040445545081609418039126332 33L)
(5, 1676514995315199752721449185453201909215855927628679801652920191915584932073001869514650667024163643140712627552483 L)
(6, 1651448484423005167174115783480277826075242161234874627378020326648540460066013704380480879027923109515261562192305 2L)
(7, 1931147149692958201340376998585354182999181082723093600123824753285545116850721842421806605282237021355841466550792 1L)
(11, 2906083980039771148281648034890083603212426776061024869777988525186171901877159136127649328594029367289012367095366L)
(12, 2218055744650312464959552436450557329018568450894600998409133265805480605420803206789081776572553820581982694657294L)
(13, 1764204782051916227121596349038764352954585264009908387865683872096627738266309190862318127973146359891051730461475 9L)
(15, 1384745725307646592188445858058964388289844654558505344543117361526732295895612281640579363931473883362100473994502 8L)
(16, 1455640524467129917173471509559983670605593852076095382949981606155972986206853147816712959805665664358886275169053 4L)
```

Figure 7 Output of the curve M-83

Case 3: sv=1, type_of_curve = P-192

```
Elliptic curve is:              P-192
Finding elliptic point closest to:      1
Prime number:                   6277101735386680763835789423207666416083908700390324961279
a,b -3 2455155546008943817740293915197451784769108058161191238065
(2, 1126956676167578795924565825825899020268914906345645360775L)
(3, 2476168101441297080746512578325117519920374855425678540834L)
(5, 9367608244086091096095807319876623418457281620273455864 43L)
(6, 6137449450752967349736559844302093506477945719219949 4327L)
(8, 1539168359597512271047259505090133446672063593980132990812L)
(12, 3464753203279792192409824182683870253677262339932562461307L)
(13, 3288234558942609973454802567986887155175778959720199156770L)
(15, 4548834217212027584647316131131523554591911664904227806291L)
(17, 2148916484007672061843886225501299518817815267521173400039L)
(18, 1600977792967480259538850281480651298625682822082373614 67L)
(22, 1682016893107185458056834822961338463540516386180178478778L)
```

Figure 8 Output of the curve P-192

Case 4: sv=1, type_of_curve = P-224

```
Elliptic curve is:              P-224
Finding elliptic point closest to:      1
Prime number:                   269599466671506397946670150870196306735579162600263081435100662988 81
a,b -3 189582862855666080004086685444939264155046809686793210757872346725 64
(3, 1383040375886749682304983968593411026955186884767752347433161151276 3L)
(5, 1491636738264668039229334312547441746589749026796775994421327947864 6L)
(8, 4965857923175733260484895597117581643516090219174079172629924655251L)
(12, 1444077118427105103263644404122230208156034712473979207983303760335 5L)
(13, 2208551277252038879572618723741009766661730649178203259591524117870 4L)
(14, 2628838385755204661425800370328400895200206013155583047896001688107 2L)
(17, 2646922703047332294499654812643344470219374499628272578724011448218 9L)
(21, 1454226854563290891426816198518281461002169117987976052169249417759 8L)
(23, 3957915491394070007137098114645685866259675432925693359495011033647L)
(24, 6692484450366337655722046283874638874856383145858273867695031485369L)
(25, 2080151641734338597060357106289533951970134819562794264224626394150 0L)
```

Figure 9 Output of the curve P-192

Case 5: sv=1, type_of_curve = P-384

```
Elliptic curve is:              P-384
Finding elliptic point closest to:      1
Prime number:
39402006196394479212279040100143613805079739270465446667948293404245721771496870329047266088258938001861606973112319
a,b -3 27580193559959705877849011840389048093056905856361568521428707301988689241309860865136260764883745107765439761230575
(2, 216818525315853169258949453321254270055023727925093593837197096493577477280247538965142854706372965090303933711652601)
(3, 157569642692787727616421236377385389838893658538273232829685559262082563743478289629147757576835267998044162943449221)
(10, 198407413814969395812333847376123277139234596007726998485487036504783057614300484667122504658865791714271237689996501)
(11, 392055633984173329023551100393081188020017379490880484619243151408716389277931250892128042527882956485333283458675271)
(12, 242693901432920116647576685576992672182814589021588748421997747121143075142908431107787951665689900418723172030444491)
(13, 252207197917520980103869540356389315932208714536134150381907783866792486795160356875366583769983566072863860740273181)
(15, 253468583043285563058645864671453924244468338405450567984102196355890748263115115821966043253054509963558492805104021)
(17, 159478212724176099441506821358093402519822125709392133464291391893741973964472767954565790056735401171839148491588791)
(20, 232540628761740765662389358349153200393854473887432863276742632491037922828948536709935683768663264775704616950523501)
(21, 146086356483383552619078831232744866908827704200303080167849775900062848387805018765977730526268004253745552124911751)
(24, 100153605030211802412073890920421173550947148113855759666008969784182330236515311462567400367977389616647728590984741)
```

Figure 10 Output of the curve P-384

Case 6: sv=1, type_of_curve = P-256

```
Elliptic curve is:              P-256
Finding elliptic point closest to:      1
Prime number:                   115792089210356248762697446949407573530086143415290314195533631308867097853951
a,b -3 41058363725152142129326129780047268409114441015993725554835256314039467401291
(5, 31468013646237722594854082025316614106172411895747863909393730389177298123724L)
(6, 24739918328848417526480033809572464882617473122143770809273908485596236620747L)
(8, 82784132184742902406611976374283817475915538652053173371854007222633983132083L)
(9, 64265364456294082747112527522981594904960689488638802860996194286474266915479L)
(12, 66934920894945216687164320457392073305141033253131913951695301233381400331143L)
(13, 53404144414778303508799263379260966483386805595332806637100379275867514529459L)
(17, 43783450011834442217334481138474159509558335318535455861762733939497581045049L)
(23, 37114329264221932393644918276665526732952696771951479144529462133154348400658L)
(24, 26321841318174713433978379012486154109266602617658289202947244817969700794459L)
(25, 10184204942107190854003808625566833285403295251792038389878320675684580912981 4L)
(26, 24424587111845263483165880188626713615059877529618610383051097479852596134318L)
```

Figure 11 Output of the curve P-256

Case 7: sv=1, type_of_curve = Curve25519

```
Elliptic curve is:              Curve25519
Finding elliptic point closest to:      1
Prime number:                   57896044618658097711785492504343953926634992332820282019728792003956564819949
a,b 486662 1
(1, 48802004052532134862652268456126542835229456083994414501085850622543968879637L)
(2, 29431104703158288096864982604786648999290556974560528399675615387295040612074L)
(3, 34960950558069024548821502971686413705384287087061328771852820095789677562855L)
(5, 19542653868059760921847591570316221965383935077385412509085312178987381883897L)
(7, 52139073615963714128182841550688261362824874109587161969880557399263946675358L)
(11, 48588697587287577436312143076572554604209962030774609811965839387269765211119L)
(12, 18755344977709762028378120083385149623799950775464356991251499055102694743805L)
(13, 24894791340871123666627859068842439039390716542506923634184662519705529677942L)
(14, 21443695855108408054027676280356423536579982788511628870763597269990603030444L)
(15, 27155698639856329604496662207857588258024549260725221119004125263197344786480L)
(16, 16574584209150355334336528319008908878560631210960302130207262364916941852931)
```

Figure 12 Output of the Curve25519

Case 8: sv=1, type_of_curve = secp256k1

```
Elliptic curve is:              secp256k1
Finding elliptic point closest to:      1
Prime number:                   11579208923731619542357098500868790785326998466564056403945758400790883467166 3
a,b 0 7
(1, 29896722852569046015560700294576055776214335159245303116488692907525646231534L)
(2, 69211104694897500952317515077652022726490027694212560352756646854116994689233L)
(3, 94471189679404635060807731153122836805497974241028285133722790318709222555876L)
(4, 40508090799132825824753983223610497878605216745196355809233758402754120847507L)
(6, 19112057249303445409876026535760519114630369653212530612662492210011362204224L)
(8, 91736135629086734185706894124002126994554994840140056297753929940646699135966L)
(12, 84723224514829953401809146057688111907524826729556536824205543511930558249867L)
(13, 20267456347483554069520440766283645831919514026818877192810320909941447705364L)
(14, 44647516152956573151520437476900496747965187933683819148138195218441524686495L)
(16, 60866531496310134771902654926928742444353904873043965024657216552925553954 3L)
(20, 95115947350322555212584100192293494006877237570979160767752142956238074546829L)
```

Figure 13 Output of the curve secp256k1

Case 9: sv=1, type_of_curve = BN(2,254)

```
Elliptic curve is:              BN(2,254)
Finding elliptic point closest to:      1
Prime number:                   16798108731015832284940804142231733909889187121439069848933715426072753864723
a,b 0 2
(2, 57212089135968944833026788643158760228372197782356350261111874607125611292L)
(4, 15986298246004125411628702755211639862408689747246680608853205234811498289 28L)
(5, 15480189765701447289516751802138904866895721078261069373948290871173946263314L)
(6, 15671260109514665628305079334560146305582236017229835757764003816432076925707L)
(7, 91152126364727529077284686832268719979953730117717767300938168367341132418320L)
(10, 15455115114928941343833094596234451025292358410044222718878949597066298393003L)
(11, 10036874330547736099225805208683266935164430115643808045424070924516554038119L)
(12, 16275559481869820839621458847205562962910576163852633302906032771786623953 76L)
(14, 15975557995194537713538106486863994424664894427400811494204684179040733088811L)
(15, 15830133140812731237424957042447222935664006893438997038581934059525810651407L)
(18, 76296242872798288973460070187699550386208076204411871835844102508632953383 01L)
```

Figure 14 Output of the curve BN(2,254)

6. PERFORMANCE ANALYSIS OF CURVES

After compiling the point generation algorithm it is observed that the curve secp256k1 takes the least time to generate first 10 points than that of other types of curves. The graph below shows the time taken in microseconds to plot the points:

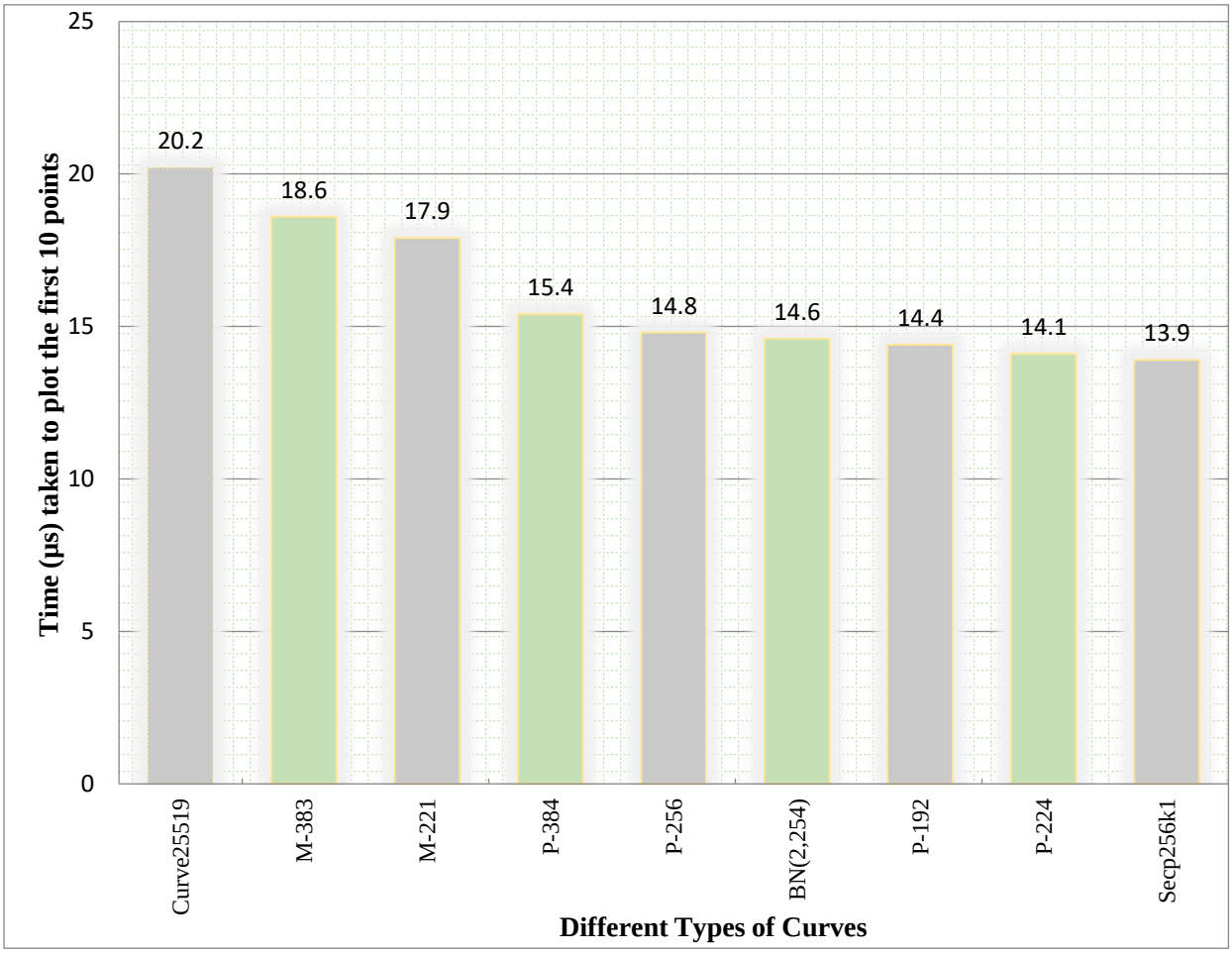

Figure 15 Comparison of curves

7. CONCLUSION AND FUTURE WORK

This paper includes various types of curves recommended by different standards that have been picked and evaluated. As point mapping is an important ECC criterion, each curve is analysed by performing point mapping. The comparative tables mentioned in this paper show the computation time taken by each curve. Depending on a particular application's security and computability prerequisites, the appropriate elliptic curve can be selected. We have shown that the secp256k1 curve appears to offer performance advantages over other curves used in ECC. This is not only in terms of overall performance but also in terms of storage memory

requirements. Depending upon the application, various ECC curves can be used. This analysis is based on the time taken for point mapping in Elliptic Curve Cryptography.

REFERENCES

[1] *Koblitz, N. (1987). Elliptic curve cryptosystems. Mathematics of computation, 48(177), 203-209.*

[2] *Sengupta, A., & Ray, U. K. (2016). Message mapping and reverse mapping in elliptic curve cryptosystem. Security and Communication Networks, 9(18), 5363-5375.*

[3] *Shaikh, J. R., Nenova, M., Iliev, G., & Valkova-Jarvis, Z. (2017, November). Analysis of standard elliptic curves for the implementation of elliptic curve cryptography in resource-constrained E-commerce applications. In 2017 IEEE International Conference on Microwaves, Antennas, Communications and Electronic Systems (COMCAS) (pp. 1-4). IEEE.*

[4] *Bernstein, D. J., & Lange, T. (2013). SafeCurves: choosing safe curves for elliptic-curve cryptography. Avialable online at http://safecurves. cr. yp. to.*

[5] *Dang, T. K., Pham, C. D., & Nguyen, T. L. (2020). A pragmatic elliptic curve cryptography-based extension for energy-efficient device-to-device communications in smart cities. Sustainable Cities and Society, 56, 102097.*

[6] *Jinasena, T., Meegama, R., & Marasinghe, R. (2017). Access Control of Medical Images using Elliptic Curve Cryptography through Effective Multi-Key Management in a Mobile Multicasting Environment, Computer Science and Engineering, 7(1), 1-11.*

[7] *Liu, Z., Seo, H., Großschädl, J., & Kim, H. (2013, November). Efficient implementation of NIST-compliant elliptic curve cryptography for sensor nodes. In International Conference on Information and Communications Security (pp. 302-317). Springer, Cham.*

[8] *Käsper, E. (2011, February). Fast elliptic curve cryptography in OpenSSL. In International Conference on Financial Cryptography and Data Security (pp. 27-39). Springer, Berlin, Heidelberg.*

[9] *Liu, Z., Seo, H., Castiglione, A., Choo, K. K. R., & Kim, H. (2018). Memory-efficient implementation of elliptic curve cryptography for the Internet-of-Things. IEEE Transactions on Dependable and Secure Computing, 16(3), 521-529.*

[10] *Baee, M. A. R., Simpson, L., Foo, E., & Pieprzyk, J. (2019). Broadcast Authentication in Latency Critical Applications: On the Efficiency of IEEE 1609.2. IEEE Transactions on Vehicular Technology, 68(12), 11577-11587.*

[11] *Düll, M., Haase, B., Hinterwälder, G., Hutter, M., Paar, C., Sánchez, A. H., & Schwabe, P. (2015). High-speed Curve25519 on 8-bit, 16-bit, and 32-bit microcontrollers. Designs, Codes and Cryptography, 77(2-3), 493-514.*

[12] *Chakraborty, S., Thomas, D., DeHart, J., Saralaya, K., Tadepalli, P., & Narendra, S. G. (2018, June). Solving Internet's Weak Link for Blockchain and IoT Applications. In Proceedings of the 1st ACM/EIGSCC Symposium on Smart Cities and Communities (pp. 1-5).*

[13] *Pereira, G. C., Simplício Jr, M. A., Naehrig, M., & Barreto, P. S. (2011). A family of implementation-friendly BN elliptic curves. Journal of Systems and Software, 84(8), 1319-1326.*