ACEEE Int. J. on Network Security , Vol. 5, No. 1, January 2014

Secure Multi-Party Negotiation: An Analysis for Electronic Payments in Mobile Computing

Dr.K.Ravikumar[1], A.Udhayakumar[2]

[1]Asst professor & UGC-NET Coordinator, Dept of Computer Science, Tamil University (Established by Government of Tamil Nadu), Thanjavur-613010
ravikasi2001@yahoo.com

[2]Asst professor A.M.JAIN College Meenambakkam-114,Research Scholar, Karpagam University,Coimbatore-641021
umaudhaya83@gmail.com

***Abstract*— This paper is an attempt to base on auctions which presents a frame work for the secure multi-party decision protocols. In addition to the implementations which are very light weighted, the main focus is on synchronizing security features for avoiding agreements manipulations and reducing the user traffic. Through this paper one can understand that this different auction protocols on top of the frame work can be collaborated using mobile devices. This paper present the negotiation between auctioneer and the proffered and this negotiation shows that multiparty security is far better than the existing system.**

***Index Terms*—Multi-party signature, Mobile authentication, Mobile Privacy, Negotiation, Electronic Payments.**

I. Introduction

Mobile communications are currently provisioned by a small number of large independent mobile networks with extensive coverage areas, often encapsulating entire regions or countries. Mobile users chose a single mobile network operator (NO) who provides connectivity and charges for usage. In order to allow their users to roam and use the services of a foreign mobile network, the home operator must establish a bilateral roaming agreement, for billing purposes, with that foreign network. User payment for calls is always settled with the home network operator, including services used while roaming in distant networks. Usage bills are produced in the case of credit-based subscribers, while the accounts of prepaid users are automatically debited. The user charge for a call placed in a foreign network is set by the home operator. This can result in two users being charged significantly different amounts for making an identical call, based on which home network they come from. As part of the thesis we propose a solution which removes the need for payment to be handled by a single home operator and allows all users to be charged equal amounts within the same mobile network.In this environment users will always be in range of one or more mobile networks and will be able to select one that best meets their requirements at the time. Roaming between independent networks will occur daily, 2 even for those mobile users who never venture out of their home city. The mobile infrastructure for such a mobile communications environment will grow from the evolution of existing wide-area cellular communications together with the emergence of low-cost local area wireless technologies. Mobile users will continue to pay for shared use of the scarce bandwidth provided by wide-area cellular communications. However, they will now also have the option of obtaining temporary access to the fixed network through a nearby local area wireless network into which they have roamed. A local wireless link can provide higher bandwidth and better quality-of-service than a wide-area mobile protocol and at a lower cost[1]. The user benefits by obtaining a better less expensive service while the local provider gains by receiving revenue for providing wireless access. In turn, use of local independent picocells will alleviate some of the traffic load from wide-area mobile providers, allowing increased reliability and service for those parties who are not in range, or who are moving too fast to use a local wireless network. The emergence and success of local area providers will require that they can receive payment for the services they provide [2], [4].

II. Multi-Party Security And Efficiency In Mobile Authentication Policy

The various negotiation policies, which are provided by applications, are installed on the proffers. Identical policies may be defined in terms of multiple applications. Each and every policy is unique and grouped by type. Each type of proffers negotiation policies negotiate sequentially during negotiation [3], [6].

Table I: Examples of Negotiation

Negotiation Identifier Type	Encryption Conference	Decryption Conference
Negotiation	10	10
Protection	10	1
Needed Bandwidth	2	1
Power saving	1	6

The given Table I contains example of Negotiation for the encryption conference and decryption conference scenarios given in the subsection.

This paper presents a scope for a policy which describes a data structure biding attributes to each value. Each and every negotiation policy of the same type consists the same attributes. Each attribute is a unique and has a value $i^{o}N$: $1<i<10$. For instance, the policy “encryption Conference” features the possibility that cooperate using encrypted telephony. Using this, the user can cooperate very efficiently [5].

DOI: 01.IJNS.5.1.22

III. Electronic Payment System Problem Definition

Through billing each party is allowed to involve in a call to eventually receive a share of the revenue generated. One can reveal a number of critical short comings and emerging problems by examining the network billing techniques [16].

IV. Multi-Party Negotiation

The design and implementation of every auction algorithm is in a form of a separate component. The procedure to implement new protocol based on finite state machines are defined using the role mentioned frame work. Model driven approach is used to design this frame work. Negotiation alone is described using this frame work, whereas the price calculations which are resulting different auctions protocols and reimbursing proffered the reimbursement component should be used.

For accessing remote services the auctions protocols depends on the previously described mechanism [9]. By initialization, proffered has to wait for the availability of an auctioneer. Once an auctioneer is as such, the value for the attribute "co-operation" is the maximum value of 10. Since the communication is encrypted, the parameter "security" also has the value 10.However, encryption and encryption telephony require a lot of power, hence the value for "power saving" is 1(Table 1). Here, these values are just now estimations for the sake of demonstration but in we have shown how convert values for such attributes can be determined and used[20], [14].

V. User Interface Negotiation

In order to allow application developers using the frame work to provide user with the ability to modify the behavior and to set individual performance values, support for user interfaces is required and as a result auction is based on individual performance [17]. In the prototype the integration of front ends for proffered running on a normal desk top PC. To start and stop the host and to export on easily phrasal history of negotiation process, the mobile device frontends provide comprehension integration with the framework, whereas the auctioneer frontends features rudimentary user interface [8]. These frontends have just been created for demonstration purposes and application developers who use framework might want to provide their own GUI's or even let the negotiation run without any possibility of user interaction at all available, all proffered try to authenticate the auctioneer using credentials based on asymmetric cryptography, which bind the auction service to a verifiable identify. It is the auctioneer who provides the credentials [12], [10].

Auctions like the key auctions help to recover each proffer to provide exactly on bid. As each message exchange takes time and drains the battery of mobile devices the proposed system is a desirable one in terms of economy[7], [13]. Whereas in other protocols in core messages is required to establish the result service multiple between rounds are conducted, example for these protocols are the English or

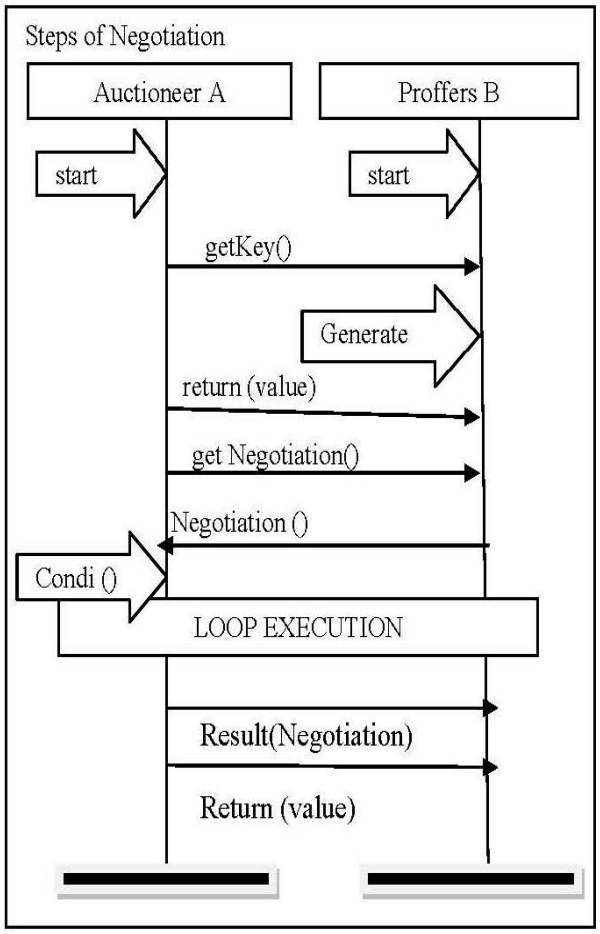

Figure 1: Steps of Negotiation

the Dutch auction for limitedly available, those protocols are less suitable [Figure 1].

The authentication process, the proffers short the auctions state machine and provide an auction service to the auctioneer. One can use this service to register negotiation, besides it is used to get bids from participate, to announce the result of an auction. The confidentiality is very much maintained between the auctioneer and the proffers. In order to provide the confidentiality, a symmetric key session is exchanged between each proffer and the auctioneer using the auctioneer credentials[8].

VI. Merits Of The Proposed Scheme

Among the various merits of the proposed system, one of the important merits lies in the confirmation of received data between auctioneer and the proffer. The other merits of this scheme are, firstly, the delay between both the ends (i.e.) between auctioneer and the proffer is very less when compared to other schemes. Secondly, the delivery ratio of the key storage is very sharp. Thirdly, it enhances the reliability and its performance is very high. In addition, it avoids the packet collision and it also reduces the broadcast redundancy and user traffic [15].

DOI: 01.IJNS.5.1. 22

VII. Performance Analysis For Multi-Party Algorithm Ecc

Domain parameters define the parameters of an elliptic curve used for the ECC calculations. Those domain parameters will have to be identical, if ECC keys are to be compatible [22].

Algorithm 1: ECC Key pair generation

Key pair generation is described by the following steps:

Step 1. Select integer w randomly in the range [1, n " 1]

Step 2. Calculate W = wP n is the order of the used elliptic curve and P is its base point. Both values are defined in the domain parameters. The private key consists of the integer value w and the domain parameters. The public key includes the point W on the elliptic curve and the domain parameters. There are several methods for encryption with ECC. We will use the Elliptic Curve Integrated Encryption Scheme (ECIES) introduced by Bellare and Rogaway and which has been standardized in ISO/IEC 15946-3 and ANSI X9.63. It is based on the ElGamal public-key encryption scheme. The following process realizes encryption.

Step 3. Select k randomly in the range [1, n " 1]

Step 4. Compute R = kP and Z = hkW. If Z = 1, go to first step

Step 5. Compute (k1, k2) = KDF(xZ,R), xZ is the x-coordinate of point Z

Step 6. Calculate c = ENCk1(m) and t = MACk2(c)

Parameters n, P and h are provided by the domain parameters and W is the public key of the host which needs to decrypt the data later on. KDF is a key derivation function based on a hashing algorithm which produces two distinct keys. ENCk1 represents a symmetric encryption cipher taking k1 as key for encryption.MACk2 is a message authentication code algorithm and uses k2 as its key. The encrypted plaintext m is located in c, and t is the authentication tag, so that the Decrypting host can validate the authenticity and correctness of the calculation. The algorithm sends R, c and t to the decrypting host. ECIES decryption is described below:

Step 7. Validate R as public key

Step 8. Compute Z = hdR. If Z = 1 reject cipher text.

Step 9. Calculate (k1, k2) = KDF (xZ, R), xZ is the x-coordinate of the point Z

Step 10. Compute t0 = MACk2(c). If t0 6= t then reject cipher text.

Step 11. Compute m = DECk1(c).The algorithm receives R, c and t from the encrypting host and the domain parameters are already known. KDF and MAC are identical to the functions used for encryption and DEC is the corresponding decryption function to ENC. The algorithm results in the plaintext m if all verifications were successful. Encryption and decryption do not need to use the MAC for authenticity verification. If data is encrypted with a different key, the resulting clear text will be different after decrypting. The different key on both hosts will prevent further successful communication. We did not test the MACs in the benchmark, as they rely on hashing and thus do not have a relevant performance impact compared to asymmetric cryptographic operations.

Multiplications are the most computationally expensive operations in ECC. Encryption performs two and decryption one multiplication. All other operations

in the ECIES encryption and decryption algorithm require only marginal computational time.

XTR

XTR is a more efficient version of RSA using traces. Like RSA, it is also based on the discrete logarithm problem.

Algorithm 2: XTR Key pair generation

The following steps produce a key pair:

Step 1. Select w randomly in the range [1; q " 2]

Step 2. Compute W = gw mod p q is the subgroup order of the trace, p is the modulo value and g is the predefined base point for calculations. All of the three values have to be predefined and used by all certificates which have to be compatible (similar to the domain parameters

for ECC). The above algorithm results in the key pair with private key w and public key W.

It is similar to the ElGamal encryption algorithm and thus also similar to ECIES. Encryption works as follows:

Step 3. Select k randomly in the range [1, q " 2]

Step 4. Compute r = gb mod p

Step 5. Compute z = Wb mod p = gkb mod p

Step 6. Compute k = KDF(z)

Step 7. Compute c = ENCk(m) The parameters q, p and g are public. W is the public key of the host which needs to decrypt the resulting data. KDF is a key derivation function based on a hash algorithm which generates a key for symmetric encryption in the desired size. ENC is a symmetric encryption algorithm which encrypts the plaintext m. Required values for decryption are r and c. Decryption of the cipher text c is performed in the following way:

Step 8. Compute z = rw mod p(= gkb mod p)

Step 9. Compute k = KDF(z)

Step 10. Compute m = DECk(c) r and c are the output of the encryption algorithm. KDF is the same function as before and DEC is the corresponding symmetric decryption algorithm to ENC. m contains the plaintext. Like ECC, XTR-ElGamal encryption needs 2 exponentiations for encryption and 1 exponentiation for decryption. Thus, encryption needs roughly twice as long as decryption.

RSA

RSA was invented by Rivest, Shamir and Adleman in 1978 . It was the first Proposed asymmetric algorithm. Many certificates and authentication methods in use today perform RSA computations. RSA is based on modular integer operations and its underlying infeasible mathematical problem is called the discrete Logarithmic problem. RSA makes use of a public parameter N which sets its maximum key size and the security of the system. N defines the modulo value of the system. All Calculations are performed modulo that value. If several key pairs are used in one RSA calculation, the key

DOI: 01.IJNS.5.1.22

pairs will have to rely on the same N.

Algorithm 3: RSA Key pair generation

The following steps show generation of a keypair for RSA without using a predefined N:

Step 1. Select two large prime numbers p 6= q.

Step 2. Calculate ‘ = (p “ 1)(q “ 1) and N = pq

Step 3. Choose an integer e in the range] 1; ‘] which does not share a factor except 1 and “1 to’

Step 4. Compute d such that de _ 1 mod ‘ Then the public key contains N and e and the private key contains N and d. Generation of the primes p and q has very high computational costs and takes the most time of the key pair generation operation.

Encryption in RSA requires only one exponentiation:

$$c = m^e \bmod N$$

The clear text m, represented as a number in the range [0;N], is risen to the power of the public key exponent e and results in the corresponding cipher text c. Decryption is performed similar to encryption:

$$m = c^d \bmod N$$

The cipher text c is risen to the power of the private key exponent d and returns the clear text m. Most popular implementations choose a small public exponent e and a large private exponent d. Then encryption is much faster than decryption, as encryption uses the small value e for exponentiation and decryption uses the larger value d.

A. Key Size Comparison

RSA, ECC and XTR use different underlying mathematical concepts to provide security. This fact results in different key sizes for a similar security level. A security level is defined as a measure, how much difficulty it causes to break a cipher with a certain key size.

TABLE II: EXAMPLES OF SYMMETRIC ALGORITHMS RSA ECC XTR

Symmetric Key	RSA	ECC	XTR
80	1024	163	170
128	3072	283	512
192	7680	409	1280
256	15,360	571	2560

Table II key sizes of similar security levels for different cryptographic algorithms. For different security levels. It is based on data from Lenstra and from Lauter . RSA based encryption are considered secure with 1024 bit keysizes at the moment. They are as secure as an EC cryptographic algorithm with a much lower key size of 163 bit and XTR with a key size of 170 bit. Symmetric ciphers are in 35 comparison more secure, they only need 80 bit keys. The higher the key size of an algorithm, the more calculations are required in its operations. That explains why XTR and ECC gain performance advantages through the use of lower keysizes. As mobile devices have very limited computational capabilities, we will not test higher keysizes than the equivalent of 1024 bit for RSA. RSA uses much higher keysizes than ECC and XTR, so its graph does not relate to the x-axis in the figures. The values for the tested keysizes are shown near the measured times of RSA. Calculation times in the RSA system sometimes showed large variances. Errorbars indicate the maximum and minimum values of the testset. The test results are summarized in the following sections. We will only discuss the keysizes 1024 bit for RSA, 160 bit for ECC and 170 bit for XTR, as smaller keysizes are considered insecure nowadays and larger keysizes require more computational[Table II] overhead, which is not readily available on our target platform.

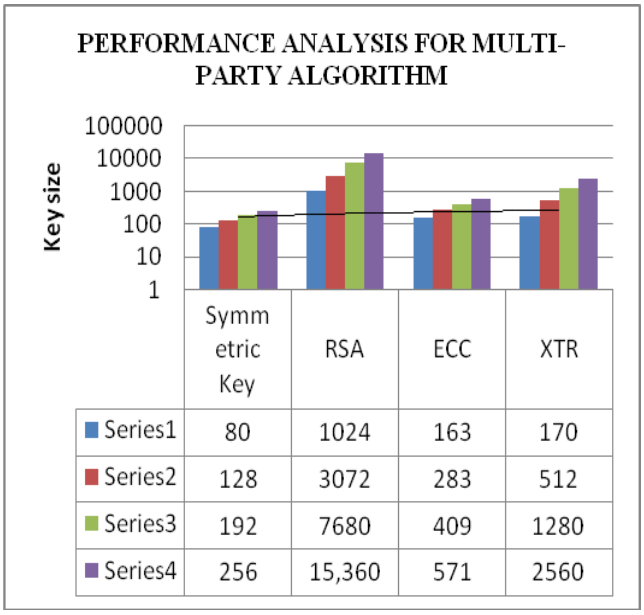

Figure 2.Key Storage Efficiency ratio

The above graph (Figure 2) demonstrates the efficiency of the ECC algorithm when it comes to the minimization of the size of the key generated in the process. Among the various algorithms which are put to use, Elliptic Curve Cryptography generates keys in the smallest possible way when compared with the other cryptographic algorithms, whose keys are the larger among the keys generated by various Cryptography systems. In the X-axis various cryptographic algorithms have been mentioned and in the Y-axis key sizes are mentioned. The graph clearly points that ECC’s key consumes lesser amount of keys [16].

VIII. CONCLUSIONS AND FUTURE WORK

The design and implementation phase for the research analysis has been desired with the frame work architecture for the secure multiparty electronic payments in mobile computing in order to support self protection in Aml scenarios. The construction of the proposed frame work is to device a platform which will be inadequate and which only focuses on the use of auctions protocols for electronic payments negotiations. The security of the auctions is considered as the top priority, the secret information which are exchanged between the auctions are encrypted and protected against illegitimate modifications. In order to shows the sensibility

DOI: 01.IJNS.5.1.22

of the architecture design, a prototype implementation has been made for Google and for the J2SE Desktop platform. This prototype features two parameterized auction protocols which were designed in a model-driven way and added in form of modules to the proposed framework. Example, an all pay auction and a key auction protocol. With the help of the proposed prototype implementation one can able to show that the examples setting of a mobile collaboration platform in both of the protocols are suitable for a secure multi-party negotiation. Overall this paper specifically considered the mechanism design in general and auctions in particular as a planning approach to cope with the ambient intelligence applications heterogeneity. In the further research one can extend to allow individual utility functions and a replaceable module for determining the winning negotiations [18], [19].

IX. Algorithm Of ECC Method In Multiparty Electronic Payments

Some comments are in proper order, before the derivation of the running time of the ECC. By selecting the random integers a and b and modulo n, a random curve E is chosen. It turns out that taking a as single-precision integers and b=1 works quite well in practice.

```
Given: A composite integer's nºN (with no small prime factors).
Output: A non-trial divisor d of n
Procedure:
While (1) {
Select a random curve E: Y2=x3+aX+b modulo n.
Choose a point p'"Q in E (Zn).
Try to compute mP./*where m is as defined in the text */
If (the computation of mP fails) {
/*we have found a divisor d>1 of n*/
If (d'"n) {Return d}}}
```

Figure 3: Algorithm 1 for Elliptic Curve method

The ECM can be effectively parallelized, since different processors can carry out the trials, that is, computation of mP together with the second stage with different sets of random elliptic curves (Figure 3).

The following graph clearly tells that, in order to do the transaction, it needs three passwords, one from the service provider and the second one from the bank and the final one the is from the account holder. Even if one fails to give the password the transaction won't be completed. And this same procedure is followed, if one want to reverse the transaction [12].

X. Real Elliptic Curves Over Real Number Adding P And Q

Elliptic curves additive groups, addition defined geometrically or algebraically (Figure4).

As we have added new instructions, the performance ratio of inversion and multiplication has changed. As a result, adopting projective coordinates may become preferable. Using the new instructions, the inversion is 3.4 times faster

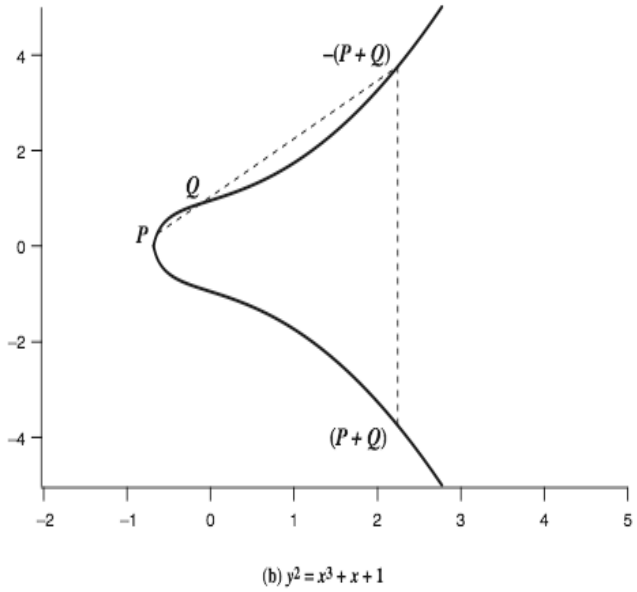

Figure 4.Multiparty electronic payments

than in the baseline architecture, in spite of that multiplication operation has been accelerated even more, when compared with the baseline architecture. As a result, the new instructions are a speed up 9.6 for multiplications. Consequently, the performance ratio of inversion and multiplication has been increased significantly from 5.6 to 15.9 .

Adopting projective coordinates to represent the points on the curve is desirable, because the inversion is now much more expensive than the multiplication. This paper adopted the projective co-ordinates discussed Figure 2 and implemented the ECC algorithm with the point addition in mixed coordinates. The performance comparison between affine and projective co-ordinates is shown in the Figure 2.As expected; the projective coordinates have better performance. They are more than twice faster than affine coordinates and achieve a speedup of 8.23 over the base line architecture. Assuming a clock rate of 16MHz, a 163-bit ECC can be performed in about 0.85 seconds [19], [20].

Over the finite field, one can't frame an obvious geometric interpretation of Elliptic Curve Arithmetic over real numbers does readily carry, this we take as an example to prove the above mentioned one.

For set $E_{17}(3,5)$,we are only interested in the nonnegative integers in the quadrant from (0,0) through (p-1,p-1) that satisfy the equation mod p. Table III list the points other than O that are part of $E_{17}(3,5)$.

Table III. Example Points Over The Elliptic Curve E_{17} (3, 5)

(1,3)	(1,4)	(2,6)	(2,11)	(4,8)	(4,9)
(5,3)	(5,14)	(6,1)	(6,16)	(9,8)	(9,9)
(10,7)	(10,10)	(11,3)	(11,14)	(12,1)	(12,16)
(15,5)	(15,12)	(16,1)	(16,16)		

Since that the number of points in E_p (a, b) is approximately equal to the number of elements in Z_p, namely p elements [Table III].

DOI: 01.IJNS.5.1. 22

a. Graphical representation of Elliptic Curve $y^2=x^3+x+1$

Some comments are in proper order, before the derivation of the running time of the ECC. By selecting the random integers a and b and modulo n, a random curve E is chosen. It turns out that taking a as single-precision integers and b=1 works quite well in practice.

Elliptic curves are not ellipses (the name comes from elliptic integrals). An elliptic curve over real is the set of points (x, y) which satisfy the equation $y^2 = x^3 + a\cdot x + b$, where x, y, a, and b are real numbers. If $4\cdot a^3 + 27\cdot b^2$ is not 0 (i.e. $x^3 + a\cdot x + b$ contains no repeated factors), then the elliptic curve can be used to form a group. An elliptic curve group consists of the points on the curve and a special point O. Elliptic curves additive groups, addition defined geometrically or algebraically (Figure 4).

As we have added new instructions, the performance ratio of inversion and multiplication has changed. As a result, adopting projective coordinates may become preferable. Using the new instructions, the inversion is 3.4 times faster than in the baseline architecture, in spite of that multiplication operation has been accelerated even more, when compared with the baseline architecture. As a result, the new instructions are a speed up 9.6 for multiplications. Consequently, the performance ratio of inversion and multiplication has been increased significantly from 5.6 to 15.9.

Adopting projective coordinates to represent the points on the curve is desirable, because the inversion is now much more expensive than the multiplication [21]. This paper adopted the projective co-ordinates discussed Figure 2 and implemented the ECC algorithm with the point addition i *mixed coordinates.*

Step 1: for i = 1 to k do ,set $s_i = x(s_{i-1} P)$,
set $r_i = lsb_{240}(x(s_i Q))$ End for
Return $r_1 . . . r_k$Step1. An elliptic curve is defined
by an equation in two variables x & y, with coefficients
Step 2: Consider a cubic elliptic curve of form,
$y^2 = x^3 + ax + b$, where x,y,a,b are all real numbers,
also define zero point O
Step 3: Consider set of points E (a, b) that satisfy
Step 4: Have addition operation for elliptic curve
Step 5: Geometrically sum of P+Q is reflection of the intersection R

Step 6: Have two families commonly used: prime curves $E_p(a,b)$ defined over Z_p, use integers modulo a prime, best in software, binary curves $E_{2m}(a,b)$ defined over $GF(2^n)$, use polynomials with binary coefficients, ,best in hardware

Step 7: ECC additions is analog of modulo multiply

Step 8: ECC repeated addition is analog of modulo exponentiation

Step 9: Need "hard" problem equiv to discrete log, Q=kP, where Q,P belong to a prime curve, is "easy" to compute Q given k,P , but "hard" to find k given Q,P,,known as the elliptic curve logarithm problem

There is no obvious geometric interpretation of elliptic curve arithmetic over finite fields. The algebraic interpretation used for elliptic curve arithmetic over real numbers does readily carry over, and this approach we take (Figure 4).

XI. Performance Analysis Of Cryptography Algorithm

The security of ECC depends on how difficult it is to determine k given kP and P. This is referred to as the elliptic curve logarithm problem. The fastest known technique for taking the elliptic curve logarithm is known as the pollard rho method. Table 2 compares various algorithms by showing comparable key sizes in terms of computational effort for cryptanalysis. As can be seen, a considerably smaller key size can be used for ECC compared to RSA. Furthermore, for equal key lengths, the computational effort required for ECC and RSA is comparable. Thus; there is a computational advantage to using ECC with a short key length than a comparably secure RSA.

Table IV. Comparable Key Sizes For Eqiivalent Security

Symmetric scheme (key size in bits)	ECC-based scheme (size of *n* in bits)	RSA/DSA (modulus size in bits)
56	112	512
80	160	1024
112	224	2048
128	256	3072
192	384	7680
256	512	15360

The above mentioned Table IV shows that the ECC-based key sizes is much lesser and productive, when compared with RSA/DSA key sizes. The different in the key size goes in an ascending order in 1:4 ratios.

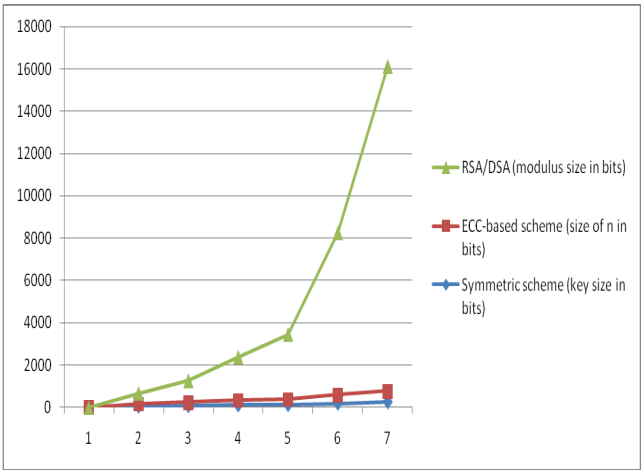

Figure 5: Comparable Key Sizes for Equivalent Security

The above mentioned graph shows that ECC based keys can be used with lesser amount of key size when compared with RSA/DSA keys. In the x- axis the key size is mentioned and in the y-axis key storage with scale of 2000 ms is mentioned (Figure 5).

XII. Performance Analysis Conclusions

It is noted that, an equal number of operations can be

DOI: 01.IJNS.5.1.22

performed by RC6 and SHA and this shows that conventional belief that hashing can be performed much faster than encryption no longer holds true for some algorithm . Number of the micropayment schemes examined based their design on this assumption. A symmetric cipher can be used as a one-way hash function. encrypt the message block *Mi*, using a function of the previous encrypted block *Mi-1* as the key *K*:

$$Hi\ (Mi) = EK\ (Mi)\ ?\ Mi$$

$$Where\ K = Hi\text{-}1(Mi\text{-}1)$$

The size of the hash result is equal to the cipher block length. Thus, for RC6 we would have a 128 bit hash, but DES would yield only a 64 bit hash, which may be too short to prevent collision attacks in many Applications. Our measurements suggest that RC6 is as efficient as SHA and could be used as both the encryption and hash unction in an implementation. Our graph shows speeds for processing large data blocks with the same key, where the key setup overhead is therefore negligible. However, from the above equation, one can see that hash functions based on block ciphers require a key change after every encryption. A surprising observation is that 3DES is only approximately twice as slow as DES, even though 3DES consists of a DES encryption, followed by a DES cryption, and a final DES encryption Overall, hashing can be up to an order of magnitude faster than symmetric encryption, three orders of magnitude faster than signature verification and four orders of magnitude faster than signature generation [11].

TABLE V. PERFORMANCE EVALUATION OF CRYPTOGRAPHY ALGORITHM

Algorithm	Key size (bits)	Key generation	Encrypt (ms)	Decry (ms)	Completion Time(sec)	CPU seconds /Node
DES	180	78	25	13	6.89	1.3
AES	190	87	27	11	19.23	2.52
RSA	1020	1224	5	40	78	4.6
ECC	160	1224	5	40	298	8.34
MD5	190	1345	7	33	23	3.45
XTR	190	56	9	32	21	4.56
DSA	170	73	23	11	158	7.23
SHA	180	72	21	13	167	8.2
RC6	200	89	27	17	253	2.45

In the Table V, Performance evaluation of cryptography algorithm is explained in a detailed manner. In those, key sizes of various cryptography algorithms like DES, AES, RSA, ECC, MD5, XTR, DSA, SHA, RC6 is mentioned. In addition, their performance evaluation is shown on the basis of their completion time and CPU seconds. As a result of this evaluation, it is very clear that the key size and key generation of ECC algorithm is much better than the other algorithms. Also, it is understandable that ECC's 160 bits is equal to RSA's 1020 bits [22].

The above graph (Figure 6) demonstrates the efficiency of the ECC algorithm when it comes to the minimization of the size of the key generated in the process. Among the various algorithms which are put to use, Elliptic Curve Cryptography

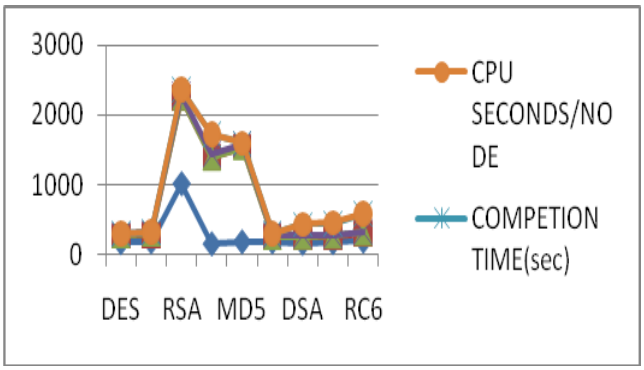

Figure 6: Performance Evaluation of Cryptography algorithm

generates keys in the smallest possible way when compared with the other cryptographic algorithms, whose keys are the larger among the keys generated by various Cryptography systems. In the X-axis various cryptographic algorithms have been mentioned and in the Y-axis key sizes are mentioned. The graph clearly points that ECC's key consumes lesser amount of keys.

XIII. CONCLUSION

This paper is a study of performance evaluation of ECC and it has been examined that how some architectural features, such as key size and ISA, which affect the performance of ECC. The algorithm, for ECC over binary field has been first examined, and after comparing algorithms for the major field operations that are required in ECC, the identification of set of efficient algorithms suitable for resource constrained systems has been done. Besides, the performance of these algorithms for different word sizes has been compared. As a result, the change of word sizes result in different choices of algorithms. The stimulation of the implementation is on an 8-bit micro controller. The proposed implementations are more than twice faster than previous results without instruction set architecture extensions or hardware accelerations.

In addition, this paper has evaluated three instructions for accelerating ECC: binary field multiplication, shift with an arbitrary shift amounts, and index of most significant combining all three instructions, one can achieve a speed up of 3.89. Important of all, the new instructions make the projective coordinators a better choice for point representations. The projective co-ordinates achieve a speed up of 8.23 over the base line architecture. It takes about 0.85 seconds to perform a 163-bit scalar point multiplication on 8-bit AVR processors at 16MHz.

This paper focuses mainly on performance analysis of ECC. Application specific hardware can be integrated into processors to accelerate the multiplication and inversion operations further. The performance of multiplication and inversion should be evaluated to choose the best point representation for better performances, when the new hardware is implemented. Finally, one can come to the conclusion that ECC is more secure than other Cryptographic algorithms and its executed speed and key storage is much better than others.

DOI: 01.IJNS.5.1.22

REFERENCES

[1] N. gura, a. patel a.wanter "comparing elliptic curve cryptography and RSA on 8-bit cpu, "proceedings of cryptographic hardware and embedded system "2004.

[2] D. Hankerson and A. Menezes "software implementation of elliptic curve cryptographic over binary fields," proceedings of workshop cryptography hardware embedded system "2000.

[3] D.J. malan , m.welsh " a public key infrastructure for key distribution in tinyOS based on elliptic curve cryptographic," 2004.

[4] Atmel corporation, 8-bit microcontroller with 128K bytes in-system programmable flash: AT mega 128, 2004.

[5] N. koblitz, elliptic curve cryptosystem," mathematics of computation, vol. 48, pp. 203-209, 1987.

[6] I. blake, g. seroussi, and n. smart, elliptic curves in cryptography, Cambridge University press, 1999.

[7] L.lopcz and r.dahab, "high speed software multiplication in F(2m)," proceedings of idocrypto '00, pp. 203-212, 2000.

[8] V.miller, "uses of elliptic curves in cryptography," advance in cryptology: proceedings of crypto '85, pp. 471-426, 1986.

[9] National institute of standards and technology, digital signature standard, FIPS publication 186-2, feb. 2000

[10] J.solinas " efficient arithmetic on koblitz curves, " designs, codes and crypto graphy, vol. 19, pp.195-249, 2000.

[11] N. koblitz, elliptic curve cryptosystem," mathematics of computation, vol. 48, pp. 203-209, 1987.

[12] I. blake, g. seroussi, and n. smart, elliptic curves in cryptography, Cambridge University press, 1999.

[13] L.lopcz and r.dahab, "high speed software multiplication in F(2m)," proceedings of idocrypto '00, pp. 203-212, 2000.

[14] V.miller, "uses of elliptic curves in cryptography," advance in cryptology: proceedings of crypto '85, pp. 471-426, 1986.

[15] National institute of standards and technology, digital signature standard, FIPS publication 186-2, feb. 2000

[16] J.solinas "Efficient arithmetic on koblitz curves, " designs, codes and crypto graphy, vol. 19, pp.195-249, 2000.

[17] I. blake, g. seroussi, and n. smart, elliptic curves in cryptography, Cambridge University press, 1999.

[18] L.lopcz and r.dahab, "high speed software multiplication in F(2m)," proceedings of idocrypto '00, pp. 203-212, 2000.

[19] V.miller, "uses of elliptic curves in cryptography," advance in cryptology: proceedings of crypto '85, pp. 471-426, 1986.

[20] National institute of standards and technology, digital signature standard, FIPS publication 186-2, feb. 2000

[21] J.solinas " efficient arithmetic on koblitz curves, " designs, codes and crypto graphy, vol. 19, pp.195-249, 2000.

[22] N. gura, a. patel a.wanter "comparing elliptic curve cryptography and RSA on 8-bit cpu, "proceedings of cryptographic hardware and embedded system "2004.

DOI: 01.IJNS.5.1.22