American Journal of Applied Sciences 11 (2): 171-179, 2014
ISSN: 1546-9239
©2014 Science Publication
doi:10.3844/ajassp.2014.171.179 Published Online 11 (2) 2014 (http://www.thescipub.com/ajas.toc)

CERTIFICATE AUTHORITY SCHEMES USING ELLIPTIC CURVE CRYPTOGRAPHY, RSA AND THEIR VARIANTS-SIMULATION USING NS2

[1]Shivkumar, S. and [2]G. Umamaheswari

[1]Research Scholar, Anna University, Chennai, India
[2]Assistant Professor (Sr Grade), Department of Electronics and Communication Engineering,
PSG College of Technology, Coimbatore, India

Received 2013-09-16; Revised 2013-09-30; Accepted 2013-12-18

ABSTRACT

A PKI (public key infrastructure) enables users of a basically unsecure public network to securely and privately exchange data through the use of a public and a private cryptographic key pair that is obtained and shared through a trusted authority. The public key infrastructure provides for a digital certificate that can identify an individual or an organization and directory services that can store and, when necessary, revoke the certificates. Although the components of a PKI are generally understood, a number of different vendor approaches and services are emerging. The two major digital signature algorithms are Elliptic Curve Digital Signature Algorithm (ECDSA) which is the elliptic curve analogue of the Digital Signature Algorithm (DSA) and RSA algorithm. The two algorithms are used for generating the certificates exchanged between computer systems. Elliptic curve based systems can give better security compared to RSA with less key size. This study compares the performance of ECC based signature schemes and RSA schemes using NS2 simulation. It is observed that ECC based certificate authority schemes gives better speed and security. Elliptic curve based schemes are the best for time and resource constrained wireless applications.

Keywords: Integer Factorization, Discrete Logarithm Problem, Elliptic Curve Cryptography, Digital Signature, ECDSA, Public Key Cryptosystem

1. INTRODUCTION

The public key infrastructure assumes the use of *public key cryptography*, which is the most common method on the Internet and other applications for authenticating a message sender or encrypting a message. Traditional cryptography has usually involved the creation and sharing of a secret key for the encryption and decryption of messages. This secret or private key system has the significant flaw that if the key is discovered or intercepted by someone else, messages can easily be decrypted. For this reason, public key cryptography and the public key infrastructure is the preferred approach on the Internet. The private key system is sometimes known as *symmetric cryptography* and the public key system as *asymmetric cryptography* (Dou *et al*., 2012).

In public key cryptography, a public and private key are created simultaneously using the Same Algorithm (RSA) by a Certificate Authority (CA). The private key is given only to the requesting party and the public key is made publicly available (as part of a digital certificate) in a directory that all parties can access. The private key is never shared with anyone or sent across the network. The private key is used to decrypt the text that has been encrypted with the public key by someone else (who can find out the public key from a public directory). In addition to encrypting messages (which ensures privacy), a user can authenticate himself by using the private key to encrypt a digital certificate.

Corresponding Author: Shivkumar, S., Research Scholar, Anna University, Chennai, India

Table 1. Simulation parameters

No. of nodes	: 50
Area size	: 1500×500 m
MAC	: 802.11
Radio range	: 250 m
Simulation time	: 50 sec
Traffic source	: CBR
Packet size	: 512 bytes
Mobility model	: Random way point

A digital signature is a cryptographically secure method of establishing with a high degree of certainty that the person who electronically signs a message can be verified as the signer with the same confidence as that provided by a witness to a handwritten signature. A digital signature is similar to the Message Authentication Code (MAC) used with symmetric (secret) key systems. The signature is a cryptographic checksum computed as a function of a message and the user's private key. Because public-key systems tend to be slow, digital signatures are often used to sign a condensed version of a message, called a message digest, rather than the message itself. A message digest can be readily generated by a hashing function.

Figure 1 illustrates a generalized signature generation and verification process. The two users must agree to use the same hash function and have access to each other's public key.

It may also be useful to be able to encrypt the message. If that is the case, a digital signature is used to exchange a secret key with authentication, integrity, non-repudiation. Following the exchange of the secret key, messages are encrypted with the secret key and exchanged. Each transmission can also contain a digest with signature to afford continued integrity and non-repudiation assurance. As indicated in **Fig. 1** a message digest produced by a hash function is used to confirm that the message was not changed in transit and that it truly represented the original message.

The Elliptic Curve Digital Signature Algorithm (ECDSA) is the elliptic curve analogue of the DSA. ECDSA was first proposed by Scott Vanstone in the year 1992 in response to National Institute of Standards and Technology (NIST's) request for public comments on their first proposal for DSS. It was accepted in 1998 as an International Standards Organization (ISO) standard (ISO 14888-3), accepted in 1999 as an American National Standards Institute (ANSI) standard (ANSI X9.62) and accepted in 2000 as an Institute of Electrical and Electronics Engineers (IEEE) standard (IEEE 1363-2000) and a FIPS standard (FIPS 186-2) Digital signature schemes can be used to provide the following basic cryptographic services (FIPSP, 2000):

- Data integrity (the assurance that data has not been altered by unauthorized or unknown means)
- Data origin authentication (the assurance that the source of data is as claimed)
- Non-repudiation (the assurance that an entity cannot deny previous actions or commitments)

The rest of the paper is organized as follows: Section 2 discusses RSA algorithm. Elliptic curve digital signature algorithm is given in section 3. Related works are discussed in section 4. Simulation results and discussions are given in section 5 and conclusion is presented in section 6.

1.1. RSA

RSA is one of the oldest and most widely used public key cryptographic algorithms. The algorithm was invented in 1977 by Ron Rivest, Adi Shamir and Leonard Adleman. The RSA cryptosystem is based on the assumption that factoring is a computationally hard task. This means that given sufficient computational resources and time, an adversary should not be able to "break" RSA (obtain a private key) by factoring. This does not mean that factoring is the only way to "break" RSA. In fact, breaking RSA may be easier than factoring.

1.2. RSA Key Generation

A RSA public and private key pair can be generated using the algorithm below:

- Choose two random prime numbers p and q
- Compute n such that $n = p * q$
- Compute φ (n) such that $\varphi (n) = (p-1)*(q-1)$.
- Choose a random integer e such that $1< e < \varphi (n)$ and gcd (e, φ (n)) = 1, then compute the integer d such that: $e*d \equiv 1 \bmod \varphi (n)$
- (e, n) is the public key and (d, n) is the private key

1.3. RSA Signature Generation and Verification

Signature of a message m is a straightforward modular exponentiation using the hash of the message and the private key. The signature s can be obtained by:

$$s = \text{hash (m) d (mod n)}$$

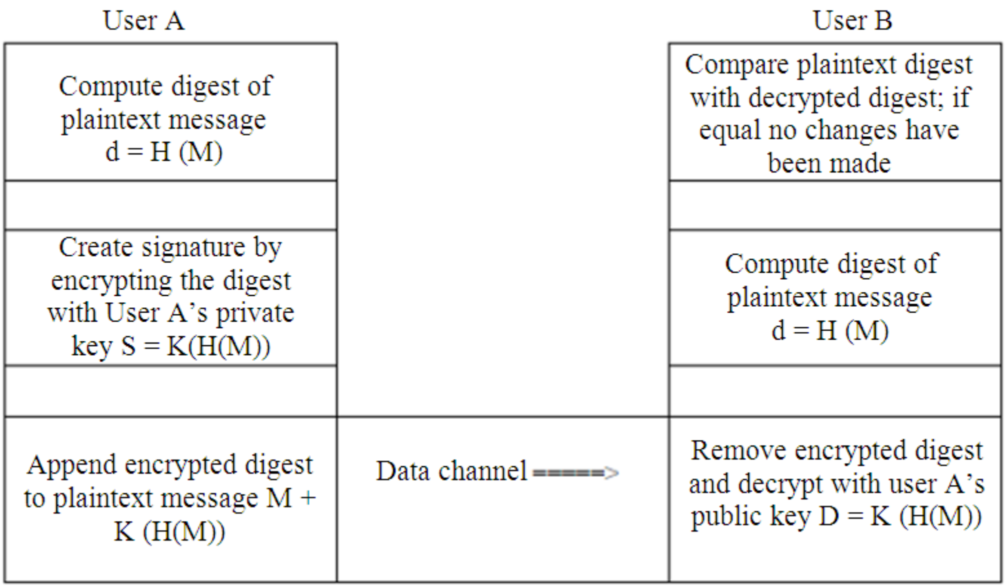

User A
User B
Compute digest of plaintext message d = H (M)
Compare plaintext digest with decrypted digest; if equal no changes have been made
Create signature by encrypting the digest with User A's private key S = K(H(M))
Compute digest of plaintext message d = H (M)
Append encrypted digest to plaintext message M + K (H(M))
Data channel =====>
Remove encrypted digest and decrypt with user A's public key D = K (H(M))

Fig. 1. Generalized signature generation and authentication

A common hash algorithm used is SHA-1.To verify a signature s for message m, the signature must first be decrypted using the author's public key (e, n). The hash h is thus obtained by:

$$h = s^e \pmod n$$

If h matches hash (m), then the signature is valid. The message was signed by the author and the message has not been modified since signing.

2. ELLIPTIC CURVE DIGITAL SIGNATURE ALGORITHM (ECDSA)

Elliptic Curve Digital Signature Algorithm is implemented over elliptic curve P-192 as mandated by ANSI X9.62 in C language. The Project contains necessary modules for domain parameters generation, key generation, signature generation and signature verification over the elliptic curve. ECDSA has three phases, key generation, signature generation and signature verification (Hankerson *et al*., 2004).

2.1. ECDSA Signature Generation and Verification

To sign a message m, an entity A with domain parameters D = (q, FR, a, b, G, n, h) does the following:

- Select a random or pseudorandom integer k in the interval [1, n-1]
- Compute k.G = (x_1, y_1) and convert x_1 to an integer $x_{1'}$
- Compute r = x_1 mod n. If r = 0 then go back to step 1
- Compute k^{-1} mod n
- Compute hash (m) and convert this bit string to an integer
- Compute s= k^{-1} {e + d.r} mod n. If s = 0, then go to first step.
- A's signature for the message m is the pair of integers (r, s)

To verify A's signature (r, s) on m, B obtains an authenticated copy of A's domain parameters D = (q, FR, a, b, G, n, h) and associated public key Q and does the following:

- Verify that r and s are integers in the interval [1, n- 1]
- Compute hash(m) and convert this bit string into an integer e
- Compute w = (s^{-1}) mod n
- Compute u_1 = e w mod n and u_2 = r w mod n
- Compute X = u_1G + u_2G
- If X = 0, then reject the signature, else convert the x coordinate of X to an integer $x_{1'}$ and compute v = $x_{1'}$ mod n
- Accept the signature iff v = r

3. RELATED WORK

Internet Key Exchange (IKE) protocol is the most common usable mechanism to exchange keying materials and negotiate security associations between two distant entities. This study proposes a new flexible approach for complexity reduction and security improvement of the IKE implementation. In this study, an initial secret key negotiation based on Elliptic Curve Cryptography (ECC) for phase 1 of IKE has been proposed, which instead of RSA, uses ECC-based public key certificate for authentication of the entities (Ray and Biswas, 2012).

Establishing a distributed virtual CA is an important tool to ensure the security of the wireless mesh networks. In these scenarios, several nodes jointly reserve the system's private key. This article proposes a RSA key sharing scheme based on dynamic threshold secret sharing algorithm (Min and Ting-Lei, 2010).

Secure Electronic Transaction (SET) is a standard protocol for the credit card transaction in e-commerce. Adopting Elliptic Curve Cryptography (ECC) instead RSA performed authentication and verified the integrity of data and the public key and private key of cardholder, merchant, payment gateway and certificate authority were distributed based on ECC. Security analysis shows that this scheme has high security and efficient authentication (Cao, 2011).

The use of X.509v3 certificates to carry out authentication tasks is an approach to improve security. These certificates are usually employed with the RSA algorithm. Elliptic Curve Cryptography (ECC) is a cryptographic technique eminently suited for small devices, like those used in wireless communications and is gaining momentum. The main advantage of ECC versus RSA is that for the same level of security it requires a much shorter key length. The purpose of this study is to design and implement a free open-source Certification Authority able to issue X.509v3 certificates using ECC. The result of this research may assist organizations to increase their security level in wireless devices and networks, in a costless way, by including authentication techniques based on ECC digital certificates (Cano *et al*., 2007).

Elliptic Curve Cryptography (ECC) is emerging as an attractive alternative to traditional public-key cryptosystems (RSA, DSA, DH). ECC offers equivalent security with smaller key sizes resulting in faster computations, lower power consumption, as well as memory and bandwidth savings. While these characteristics make ECC especially appealing for mobile devices, they can also alleviate the computational burden on secure web servers. This article studies the performance impact of using ECC with Secure Sockets Layer (SSL), the dominant Internet security protocol. We benchmark the Apache web server with an ECC-enhanced version of open SSL under a variety of conditions. Our results show that an Apache web server can handle 11-31% more HTTPS requests per second when using ECC rather than RSA at short-term security levels. At security levels necessary to protect data beyond 2010, the use of ECC over RSA improves server performance by 110-279% under realistic workloads (Gupta *et al*., 2004).

Lee and Kim (2002) proposed a two-pass hybrid key distribution and authentication protocol. The proposed protocol minimizes the number of message exchanges and the key management problem as it eliminates KDC, by using both symmetric-key and asymmetric-key schemes. In addition, it guarantees explicit entity and key authentication via a signature scheme based on Elliptic Curve Cryptosystems (ECC) whose efficiency is superior to existing signature schemes with only two-message exchanges.

Savari *et al.* (2012) compared elliptic curve cryptography with RSA algorithm on a multipurpose smart card. ECC is compared with 160 and 1024 bit key size with RSA.

4. SIMULATION RESULTS

NS2 is used for simulation. In our simulation, the channel capacity of mobile hosts is set to the same value: 2 Mbps. The Distributed Coordination Function (DCF) of IEEE 802.11 for wireless LANs is used as the MAC layer protocol. It has the functionality to notify the network layer about link breakage.

In our simulation, 100 mobile nodes move in a 1500×500 m region for 50 sec simulation time. It is assumed that each node moves independently with the same average speed. All nodes have the same transmission range of 250 m. The number of attacking nodes varies from 2-10. The simulated traffic is Constant Bit Rate (CBR). The simulation settings and parameters are summarized in **Table 1**.

4.1. Performance Metrics

The performance is evaluated according to the following metrics.

4.2. Average End-to-End Delay

The end-to-end-delay is averaged over all surviving data packets from the sources to the destinations.

4.3. Average Packet Delivery Ratio

It is the ratio of the number of packets received successfully and the total number of packets transmitted.

4.4. Drop

It is the number of packets is dropped during the transmission.

The various combinations of ECC and RSA along with other algorithms used for hash function, encryption/decryption and signature generation are given in **Table 2**. Each combination is given a code. For key generation ECC and RSA are used. SHA 1 is the commonly used algorithm for message digest generation and it is compared with MD5. These two algorithms are used with ECC and RSA and its performance is compared. For encryption and decryption we have chosen ECIES which is an elliptic curve based algorithm and RSA. ECDSA and RSA are used for digital signature generation and their performance is compared based on the combination of algorithms as given in **Table 2**.

The time delay involved in generating the certificate for all the combination of ECC based algorithms and RSA algorithm is measured. It is found that the ECC based schemes have less delay compared to RSA algorithm. The simulations are done using Network Simulator NS2 and the measured delay for all the combinations are given in **Table 3**.

A Certificate Authority server generates certificates upon the request from the client. The delay between the client requesting for the certificate and the server issuing the certificate to the client is measured as end to end delay between the client and server. This end to end delay for some combinations of ECC and RSA algorithms is given in **Table 4** along with throughput. There is a little variation in throughput between the two schemes but the time delay is manifold for RSA compared to ECC.

A screen shot showing the simulation is shown in **Fig. 2**. This gives the delay in communication between wireless node 3 and 9.The process involves decryption, hash generation and verification of the certificate. The simulated results of throughput, delay and jitter are shown in the screenshot.

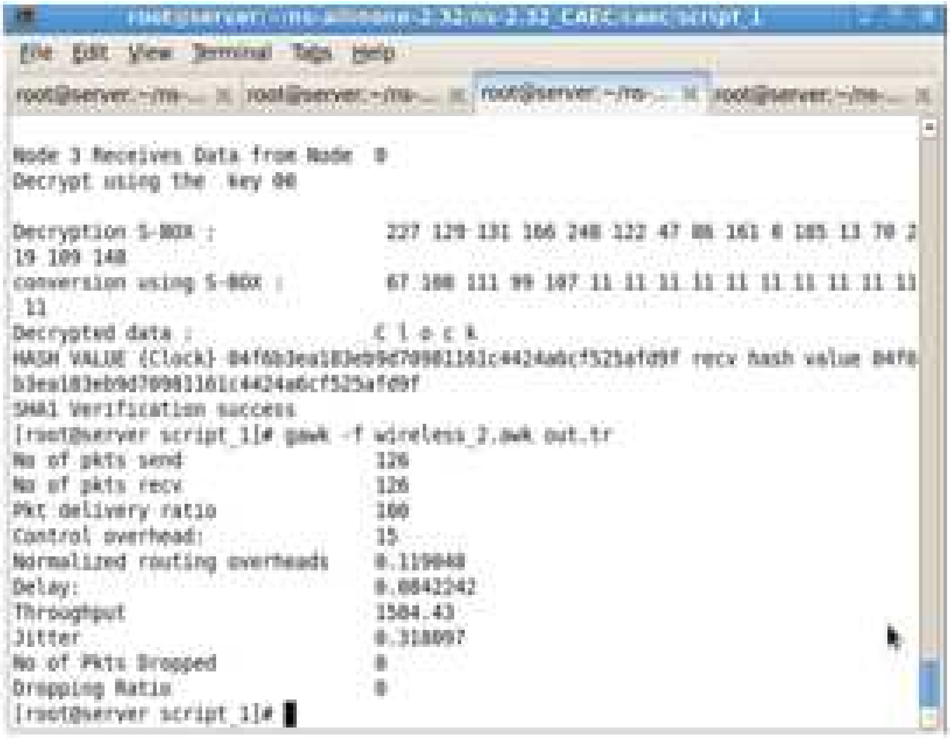
Figure 3 shows the delay involved in the generation of certificates using various combinations of ECC scheme. It may be noted that both DSA and ECDSA with MD5 message digest algorithm produces less delay in generating the certificates. This is because of the less complexity of MD5 message digest algorithm compared to SHA 1. **Figure 4** shows the delay involved in the generation of certificates using various combinations of RSA scheme. In this, the encryption scheme used is ECIES and this leads to lesser delay compared to other combinations. The main reason for the attractiveness of ECDSA is the fact that there is no sub exponential algorithm known to solve the elliptic curve discrete logarithm problem on a properly chosen elliptic curve.

Hence, it takes full exponential time to solve while the best algorithm known for solving the underlying integer factorization for RSA and discrete logarithm problem in DSA both take sub exponential time.

Figure 5 gives the delay involved in various combinations of algorithms using RSA for key generation and encryption. The combinations using ECIES and MD5 involve less delay than SHA 1 and RSA algorithms. Higher delay in RSA is due to the calculation of exponents for getting private and public keys.

Figure 6 compares the throughput of ECIES and RSA algorithms. The combination of Elliptic curve and SHA-I algorithm provides strong cryptographic strength and optimizes the computational speed as well as space. As the proposed method is based on the strength of the elliptic curve discrete logarithm problem, it is not vulnerable for cryptanalytic attacks which are readily available.

The key generated by the implementation is highly secured and it consumes lesser bandwidth because of small key size used by the elliptic curves. Significantly smaller parameters can be used in ECDSA than in other competitive systems such as RSA and DSA but with equivalent levels of security. Some benefits of having smaller key size include faster computation time and reduction in processing power, storage space and bandwidth. This makes ECDSA ideal for constrained environments such as wireless networks. These advantages are especially important in other environments where processing power, storage space, bandwidth, or power consumption are lacking. The end-to-end delay measured for ECC scheme is shown in **Fig. 7**.


root@server:~/ns-allinone-2.32/ns-2.32_CAEC/caec/script_1
File Edit View Terminal Tabs Help
root@server: ~/ns-...
root@server: ~/ns-...
root@server: ~/ns-...
root@server: ~/ns-...

```
Node 3 Receives Data from Node  0
Decrypt using the  key 00

Decryption S-BOX :              227 129 131 166 248 122 47 86 161 6 185 13 70 2
19 109 148
conversion using S-BOX :        67 108 111 99 107 11 11 11 11 11 11 11 11 11 11
 11
Decrypted data :               C l o c k
HASH VALUE (Clock) 84f6b3ea183eb9d70981161c4424a6cf525afd9f recv hash value 84f6
b3ea183eb9d70981161c4424a6cf525afd9f
SHA1 Verification success
[root@server script_1]# gawk -f wireless_2.awk out.tr
No of pkts send                126
No of pkts recv                126
Pkt delivery ratio             100
Control overhead:              15
Normalized routing overheads   0.119048
Delay:                         0.0842242
Throughput                     1504.43
Jitter                         0.318097
No of Pkts Dropped             0
Dropping Ratio                 0
[root@server script_1]# █
```

Fig. 2. Screenshot showing delay and jitter

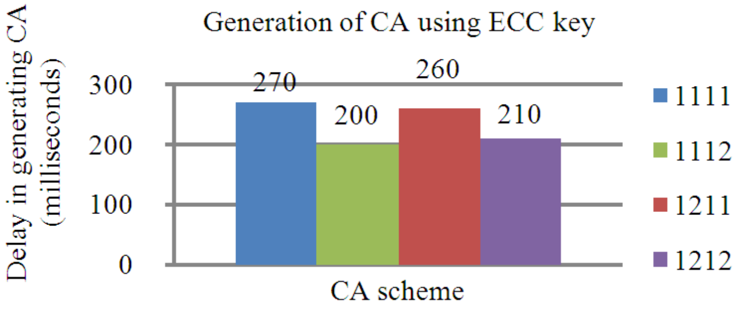


Fig. 3. Delay in generating certificates using ECC schemes

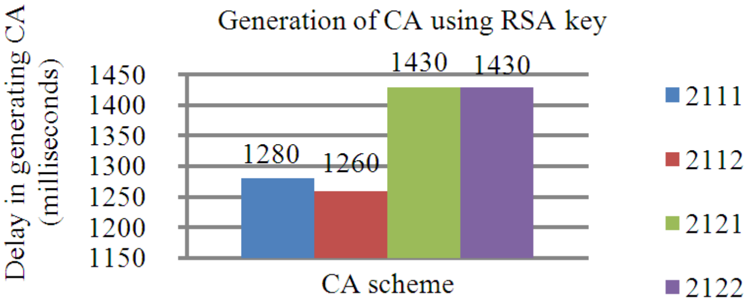


Fig. 4. Delay in generating certificates using RSA schemes

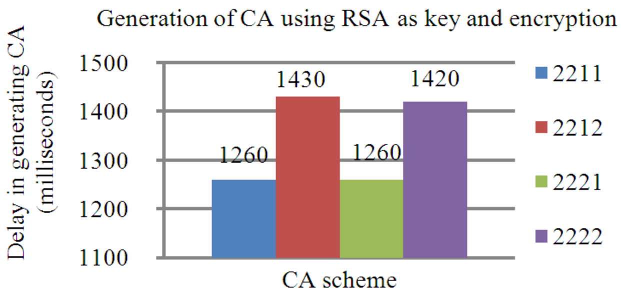


Fig. 5. RSA key generation and encryption delay

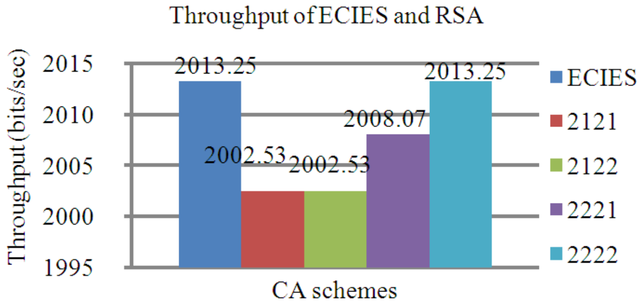


Fig. 6. Throughput for ECIES and RSA schemes

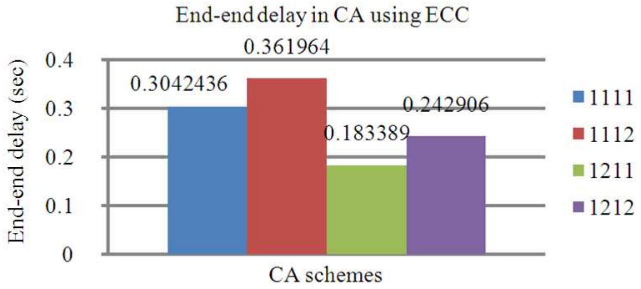


Fig. 7. End-to-end delay involved in generation and issue of certificate to clients in ECC schemes

Table 2. ECC, RSA and combination of other algorithms.

Key generation	Hash function	Encryption and Decryption	Digital signature	Algorithm combination code
ECC	SHA 1	ECIES	ECDSA	1111
	SHA 1	ECIES	DSA	1112
	MD5	ECIES	ECDSA	1211
	MD5	ECIES	DSA	1212
RSA	SHA 1	ECIES	DSA	2111
	SHA 1	ECIES	RSA	2112
	SHA 1	RSA	DSA	2121
	SHA 1	RSA	RSA	2122
	MD5	ECIES	DSA	2211
	MD5	ECIES	RSA	2212
	MD5	RSA	DSA	2221
	MD5	RSA	RSA	2222

Table 3. Measured delay for various combinations

Algorithm combination code	Time delay (ms)
1111 (ECC)	270
1112 (ECC)	200
1211 (ECC)	260
1212 (ECC)	210
2111 (RSA)	1280
2112 (RSA)	1260
2121 (RSA)	1430
2122 (RSA)	1430
2211 (RSA)	1260
2212 (RSA)	1430
2221 (RSA)	1260
2222 (RSA)	1420

Table 4. Throughput and delays

CA Scheme	Throughput (bits/sec)	Delay (milliseconds)
1111	2013.25	0.302436
1112	2013.25	0.361964
1211	2013.25	0.183389
1212	2013.25	0.242906
2121	2002.53	109.886
2122	2002.53	110.063
2221	2008.07	110.121
2222	2013.25	110.057

Table 5. Comparison of RSA (1024 bits)

Parameters	Alese *et al*. (2012)	Our results
Key generation	1312.7 (ms)	1274 (ms)
Encryption	166.9 (ms)	146 (ms)
Total time	1479.6 (ms)	1420 (ms)
Throughput	NA	2013.25 (bits/sec)
Delay	NA	110.057
Certificate generation delay	NA	1430 (ms)
End to end delay	NA	110.057 (sec)

Table 6. Comparison of ECC (P-224)

Parameters	Alese *et al*. (2012)	Our results
Key generation	208.3 (ms)	131 (ms)
Encryption	95.9 (ms)	79 (ms)
Total time	304.2 (ms)	210 (ms)
Throughput	NA	2013.25 (bits/sec)
Delay	NA	0.361964 (ms)
Certificate generation delay	NA	210(ms)
End to end delay	NA	0.1833 (sec)

4.5. Comparison with Other Works

Alese *et al*. (2012) has done the comparison of ECC and RSA algorithms. In our paper two different encryption and digital signature algorithms are used along with RSA algorithm. One of the combination uses RSA algorithm for encryption, decryption and digital signature generation. The key size is 1024 bits. **Table 5** gives the comparison of RSA algorithm.

For the elliptic curve cryptography, we have used the EC group P-224. The comparison of simulation results are given in **Table 6**.

5. CONCLUSION

In this study, we presented the simulation of ECC and RSA algorithms for various combinations of ECDSA, DSA, RSA, MD5, SHA-1 used for encryption, decryption and digital signature operations. The certificate generation delay, encryption delay, throughput, end-to-end delay in generating and issuing certificate to clients are measured for all the combinations. It is found that ECC based combinations outperform RSA based combinations of algorithms in terms of encryption, throughput and end-to-end delay.

This concludes that ECC is best suited for wireless applications which demands speed, time and bandwidth. Our results are compared with other works which shows that we are able to obtain lesser key generation time, encryption time and throughput because of our optimized code simulation. Our work implies that ECDSA used for generating certificates is more efficient than other combinations. ECC gives the same level of security with less key size when compared to RSA.

ECC based authentication protocol for wireless applications is recently proposed. Wireless applications require low power, less memory space and bandwidth. ECC suits the best for this application because of its speed and security. With these timings, the execution of the ECC-based wireless authentication protocol takes around 140 ms on the ARM7TDMI processor, which is a widely used, low-power core processor for wireless applications.

Using this processor our combination codes can be implemented in future and required combination can be selected from the library. This will reduce cost and time.

Also in future low power ASICs can be designed which could be customized to meet the wireless requirements.

6. REFERENCES

Alese, B.K., E.D. Philemon and S.O. Falaki, 2012. Comparative analysis of public-key encryption schemes. Int. J. Eng. Technol., 2: 1552-1568.

Cano, M.D., R. Toledo-Valera and F. Cerdan, 2007. A certification authority for elliptic curve X.509v3 certificates. Proceedings of the 3rd International Conference on Networking and Services, JUN. 19-25, IEEE Xplore Press, Athens, pp: 49-49. DOI: 10.1109/ICNS.2007.3

Cao, L.C., 2011. Improving security of SET Protocol based on ECC. Proceedings of the International Conference on Web Information Systems and Mining, Sept. 24-25, Taiyuan, China, Springer Berlin Heidelberg, pp: 234-241. DOI: 10.1007/978-3-642-23971-7_31

Dou, B., C.H. Chen, H. Zhang and C. Xu, 2012. Identity based sequential aggregate signature scheme based on RSA. Int. J. Innov. Comput. Inform. Control, 8: 6401-6413.

FIPSP, 2000. Digital signature standard. Federal Information Processing Standards Publication.

Gupta, V., D. Stebila, S. Fung, S. Chang and N. Gura *et al*., 2004. Speeding up secure web transactions using elliptic curve cryptography. Proceedings of the 11th Network and Systems Security Symposium, (SSS' 04), Internet Society, pp: 231-239.

Hankerson, D., S. Vanstone and A.J. Menezes, 2004. Guide to Elliptic Curve Cryptography. 1st Edn., Springer, New York, ISBN-10: 038795273X,pp: 311.

Lee, S.M. and T.Y. Kim, 2002. Two-pass hybrid key distribution protocol based on ECC. J. Inform. Sci. Eng., 18: 125-139.

Min, Z. and H. Ting-Lei, 2012. A RSA keys sharing scheme based on dynamic threshold secret sharing algorithm for WMNs. Proceedings of the International Conference on Intelligent Computing and Integrated Systems, Oct. 22-24, IEEE Xplore Press, Guilin, pp: 160-163. DOI: 10.1109/ICISS.2010.5656800

Ray, S. and G.P. Biswas, 2012. Establishment of ECC-based initial secrecy usable for IKE implementation. Proceedings of the World Congress on Engineering, (WCE' 12), pp: 530-535.

Savari, M., M. Montazerolzohour and Y.E. Thiam, 2012. Comparison of ECC and RSA algorithm in multipurpose smart card application. Proceedings of the International Conference on Cyber Security, Cyber Warfare and Digital Forensic, Jun. 26-28, IEEE Xplore Press, Kuala Lumpur, pp: 49-53. DOI: 10.1109/CyberSec.2012.6246121