Network Security Based Attacks
ECC over RSA for SSL Certification

Saransh[1], Saurabh Kumar[2], Mythili N[3]

[1,2]Master of Computer Applications

[3]Assistant Professor, SITE

Vellore Institute of Technology, Vellore

Saransh.2018@gmail.com

ABSTRACT – MITM attack is a type of cyber-attack by which an attacker or hacker comes into the connection between the user and the server, and gains access to all information exchange between both parties. Man-In-The Middle attack allows attacker to steal login password, files from device and also trying to manipulate requests made by user to server. The main target of Man in the Middle attack is actual data flows between end points and the confidentiality and integrity of data itself. Some common types of performing MIMT attack are Email Hijacking, Wi-Fi Eavesdropping, Session Hijacking, Port Stealing and DNS Spoofing. In this paper the focus is on how to perform MITM attack on SSL and how to prevent from Man in the Middle attack using different Key exchange algorithm. From the public Key exchange algorithm RSA and ECC it's needed to find out the best one to prevent communication between user and server from attackers. Finally, in order to provide defence strategy regarding how to decrease losses against MITM attack.

Keyword- MITM, SSL, ECC, RSA, ECDSA

I. INTRODUCTION

In today's world, market is spread across the globe. When internet is a thing for every individual irrespective of where they live, data communication without getting tapped is a challenge and a difficult task. When one tries to have a secure line for private data transmission, someone at the same time tries to compromise its privacy resulting in loss, manipulation and stealing of important and private data. It is hard to believe whether the sent or received data even over a secured line is correct or not. An intruder can easily steal, damage or manipulate the seemingly private correspondences just by catching radio frequencies by placing an antenna in the region of communication. Data transmission plays an important role in today's most of the businesses, banking system and even militaries activities of a nation. SSL (Secured Socket Layer) protocol is now widely used by almost every genuine websites to establish a secure connection using key exchange algorithms. The most vulnerable point in a SSL is when client and server tries to exchange keys. SSL protocol uses RSA and ECC algorithms for key exchange. Certification Authority plays an important role in data transmission since it verifies the digital signature sent by the server to

the client. This paper is divided into three sections. 1st section covers the significance of SSL in networks. 2nd section covers the comparison of RSA and ECC Algorithms that SSL uses. In the 3rd section and the final section of the paper, it will be discussed how ECC is still more beneficial and efficient than RSA.

II. SSL (Secured Socket Layer) - OVERVIEW

SSL is a standard security protocol which is used for establishing a secured and encrypted link between the server and the browser. In SSL, All data transmitted between server and client remain encrypted. For creating an SSL connection, it is necessary to have an SSL certificate.

Following steps are followed in HTTP over SSL

Step1:-Client requests secure pages (HTTP) from web server.

Step2:-Server sends its public key and SSL certificate (Digitally signed by CA).

Step3:-Browser verifies the certificate digital signature since browsers are installed with many CA's public key.

Step4:-Client creates one private session key and encrypt it using server public key. Client sends the copy of its key to the web server.

Step5:-Web server gets the client's encrypted key and uses its private key to decrypt it.

Now, all the traffics between client and server will be encrypted and decrypted with their keys.

III. RSA Algorithm

RSA Algorithm is used for encrypting and decrypting messages. RSA algorithm uses public key crypto system. In public key crypto system, sender encrypt data using "public key" and receiver decrypt data using "private key". Here, public key is known to everyone but private key is only known to receiver. In this RSA algorithm, private key vary from receiver to receiver.

Encrypting Message:-

"$c=m^e \bmod n$"

Here, 'c' is Cipher text, 'm' is Plain text while 'e' and 'n' are public key provided by receiver to sender. Here, 'm' always smaller than 'n'. Now, sender send cipher text 'c' to receiver.

Decrypting Message:-

Here, receiver recover plain text 'm' from cipher text 'c' by private key 'd' by using following procedure:

"$m=c^d \bmod n$"

By applying Chinese Remainder Theorem, to recover original distinct prime number:

"$m^{ed}=m \bmod p*q$" and, "**$c^d=m \bmod n$**"

Here, 'p' and 'q' are any two different prime numbers where 'p' always less than 'q'.

Thus, in this way RSA algorithm encrypt and decrypt message, and, this whole process is called as "Public Key Crypto System".

An Example:-

Here, an example is illustrated to explain the working of RSA algorithm.

1. Any two random prime number is chosen.
2. Let p=7 and q=11, then, n=pq
3. n= 7 * 11 = 77
4. Now compute, $\phi(n)=(p-1)(q-1)$
5. $\Phi(n)=(7-1)(11-1)=60$
6. Choose any 'e', such that e>1 and co-prime to 60
7. Let e=13
8. Choose any 'd' that satisfy, "d*e=1 mod $\phi(n)$"
9. d=37, is obtained using "Extended Euclidean Theorem"

The public key is (n=77 and e=13) and private key is (n=77 and d=37).

The Encryption function, **$c=m^e \bmod n$** becomes **$c=m^{13} \bmod 77$**, and Decryption function, **$m=c^d \bmod n$** becomes **$m=c^{37} \bmod 77$**

Now, let m=5 to encrypt, we get-

$c=5^{13} \bmod 77= 26$

To decrypt c=26, we calculate

$m=26^{37} \bmod 77= 5$

By these calculation encryption and decryption of data is shown.

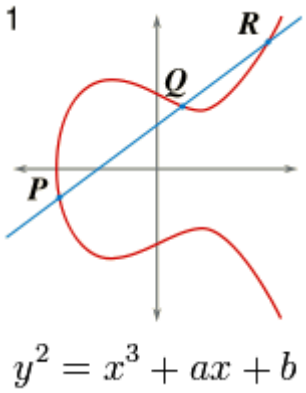
IV. ECC Algorithm

ECC stands for Elliptic Curve Cryptography, and is an algorithm which is used for asymmetric key cryptography based on elliptic curves over finite fields. Its dependency lies over the finite field of elliptical algebraic curve. Assumption is made that finding out the discrete log of a random EC element in connection to a familiar public bottom end is impractical. In 2004, algorithm entered in common use. ECC uses small key sizes which makes it very bearable for devices with limited storage or processing power. It is becoming increasingly common in the IoT. Because of its smaller key sizes, times get reduced during handshake protocol compare to RSA.

Elliptic Curve (EC) is represented by equation, **$y^2=x^3+ax+b$**

E is EC, **P** is the curve point, **n** is threshold

FIGURE I:- **EC** shown on a graph



Key Generation

First a random number '**d**' is selected which belongs to the limits of '**n**'. Using following equation, key (public) is generated:-

"**Q = d * P"**

Where, **P** is any point on curve, '**Q**' is public key and '**d'** is private key.

Encryption

Assuming that 'm' is the message which is going to be send. 'm' needs to be represented on the curve. And 'M' (say) is a point of message 'm' on the curve E.

Now, any random number 'k' is selected from 1 to (n-1). Cipher text "C_1" and "C_2" will be generated as follows:

"C_1=k * P" and "C_2=M + k * Q"

Now, sender will send C_1 and C_2 to receiver.

Decryption

Now, the actual data is retrieved from the cipher text that we send,

"M = C_2 – d * C_1"

From sender end , M which is the Original message will be sent.

Verification

Showing how message will be retrieved,

"M = C_2 – d * C_1"

"C_2 – d * C_1 = (M + k * Q) – d * (k * P) " [Since, C_2 = M + k * Q and C_1 = k * P]

= "M + k * d * P – d * k *P" = M

V. Comparison between RSA and ECC

As comparing on the basis of implementation, ECC has some advantages that it can be implemented on internet based applications, devices that have low memory and computational strength and in the area of smart cards and cryptographic tokens. The only difficulty in the implementation of ECC are on hardware and software because implementation on software required higher power consumption as compared to hardware implementation. On the basis of encryption and decryption, RSA is faster in key generation and encryption but slower in decryption While ECC is slower in key generation and encryption but faster in decryption. But, in the aspects of security ECC is more stronger than RSA.

TABLE I:-This table shows factors affecting key generation and certificate verification of RSA and ECC.

RSA	ECC
Very slow key generation	Fast key generation
Generated key size is large(Max 15360 bits)	Generated key is small(Max 521 bits)
All services and applications are interoperable with RSA based SSL Certificates.	Interoperability of all services and applications are not possible.
Fast and simple encryption and verification.	Moderately fast encryption and verification.

VI. Comparison of ECC and RSA keys

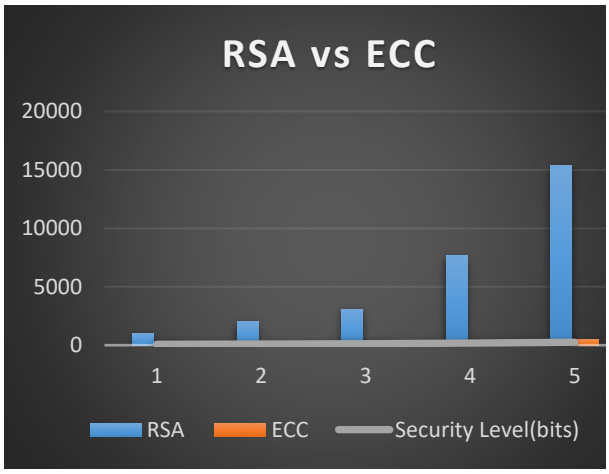
RSA keys are most commonly used in SSL Certificates and to maintain sufficient cryptography strength its key size increasing continuously ranging from 1024 to 2048 bits. While ECC is an alternative to RSS. Here, ECC provide same level of strength of cryptography as compared to RSA at very less key size. ECC improved the security as well as reduced the computational requirements. ECC with smaller key size becomes compatible for those devices that have limited storage and processing power which are becoming very familiar in IoT. For achieving stronger security and speedier SSL handshake the main requirement is smaller key size.

TABLE II:- This table shows the comparison of key size generated and security levels(bits) using RSA and ECC.

Key Size (Bits)		Security Level (Bits)
RSA	ECC	RSA and ECC
1024	160	80
2048	224	112
3072	256	128
7680	384	192
15360	521	256

Here, we say that ECC can provide same level of security and cryptographic capability in very less key size.

FIGURE II:- This figure shows that difference in size of RSA and ECC on the basis of size(bits).



VII. Related Work

In paper[1], author use SSL and HTTPS effectively to avoid the MITM attack. Author used the combination of Diffie-Hellman algorithm for establishing a shared secret connection between two parties and blowfish algorithm for Key expansion and Data Encryption, DH for key generation which enhances the data security over SSL and HTTPS for online banking transactions. Author work is mainly for email services and online banking applications which rely on HTTPS to establish encrypted and secure communications between web browser and their servers. Author uses SSL protocol over HTTPS because SSL itself has good security features.

In paper[2], Author showed that most deployed user authentication mechanisms fail to provide protection against this type of attack. More specifically, they started with a basic implementation that employs impersonal authentication tokens, addressed extensions and enhancements, gave an informal security analysis, and presented possibilities for implementing SSL/TLS session-aware user authentication in software. As of this writing, they are prototyping SSL/TLS session-aware user authentication for Internet banking. The implementation renders and displays the first digits of the Hash value as it appears in the execution of the SSL/TLS handshake protocol. Author also hopes that this ‘‘trusted observer’’ extension will someday become standard in browsers.

In paper[3], Author perform a systematic study of browser cache poisoning attacks against HTTPS connections, which persistently compromise the victim’s web sessions with the target site by poisoning the victim’s browser cache. Through experiments on mainstream desktop browsers and popular mobile browsers, they find the inconsistency of SSL warnings and incoherence of browser caching policies. In particular, the majority of mobile browsers do not deploy SSL warnings properly, and always cache resources over broken HTTPS. In our evaluation, they demonstrate that popular browsers are susceptible to BCP attacks. Author also find that Android browser-like applications do not display SSL warning and are vulnerable to BCP attacks.

Meanwhile, only five sites of Alexa Top 100 and 1.63% of 31,377 HTTPS websites have partial protections. Furthermore, they discuss pros and cons of potential defenses, and provide guidelines for users and browser vendors to defeat BCP attacks. Author also propose defense techniques for web developers to mitigate the impact of these attacks on the existing deployment of browsers.

VIII. Proposed Methodology

Elliptical Curve Cryptography (ECC) is for data encryption, for exchanging key and it is also for generation of digital signature. Now a days, ECC gaining much more popularity in the application because its performance provides advantages over other public key encryption algorithm. ECC needs some more security level because Certificate provided by the server also contain a Digital Signature. The necessity for providing security to Digital Signature is that in 2010 a hackers group called failOverow provided a way to copy sign software of Sony's Game Console, Playstation 3 in the 27^{th} Chaos Communication. And the resultant outcome is that they download copied and licenced games from Playstation 3 and malicious software that was officially signed by Sony. Thus, ECDSA is introduced that helps in code authentication that checks and verifies Digital Signature of a binary files before it runs on processor. Therefore, proposing ECDSA for digital signature provides better security while having connection through server.

A. ECDSA Algorithm

Elliptical Curve and the Digital Signature Algorithm offers high security with elliptical curve. ECDSA provides variety of the DSA which uses ECC. ECDSA requires public key twice the size of security level where as the size of signature is four times the size of its security level in bits.

Signature Generation Algorithm:

Let sender wants to send message to receiver. For that they agreed on a parameters (CURVE,G,n). Here, 'CURVE' is the elliptical curve field and equation used, 'G' is elliptical curve base point and a large prime number act as a generator of elliptical curve, and 'n' is integer act as multiplicative order of 'G' that results 'O' the identity element.

Now the sender creates a "private key" pair "d_A" chosen from 1 to n-1. Also creates a "public key" curve point "$Q_A=G * d_A$". For creating "digital signature" for message m, following steps are followed by the sender:-

- Here calculate e is equal to "HASH(m)", which is the "cryptographic hash function"
- Left most bit of 'e' is L_n be 'z'
- Now a integer number 'k' be selected in range [1,n-1] that must be cryptographically secure random
- x_1 and y_1 are the point on the curve G is calculated
- "$r=x_1$ mod n". Check whether r is equals to 0 ,then jump to 3^{rd} step
- "$s = k^{-1}(z+rd_A)$ mod n". Check whether s is equals to 0, then jump to 3^{rd} step
- Thus, generated pair of signature is r and s.

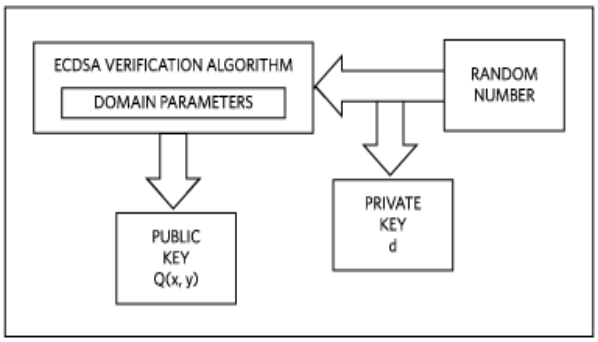


FIGURE III: Key pair generation process

Signature Verification Algorithm:

Here, receiver to authenticate senders signature must require Q_A, the public key curve point. Now, receiver follows these steps to check validity of curve point:

- Identity element O must not be equal to Q_A
- Q_A must lies on curve
- n * Q_A must equal to O

Now, receiver also follows these steps:

- Check 'r' and 's' are in range of [1,n-1] . If not matched then signature is considered as invalid
- "e = HASH(m)" is calculated
- Let L_n left most bit of 'e' be 'z'
- "$w = s^{-1} \bmod n$", is calculated
- "u1 = zw mod n" and "u2 = rw mod n"
- "The Curve Point (x1,y1) which is equals to $u_1 * G + u_2 * Q_A$" is calculated. . Check, if $(x_1,y_1) = O$ then signature is invalid
- Now, signature is said to be valid, if 'r' is equivalent to $x_1 \bmod n$

From here we get a function that is sum of scalar multiplication of $u_1 * G + u_2 * Q_A$

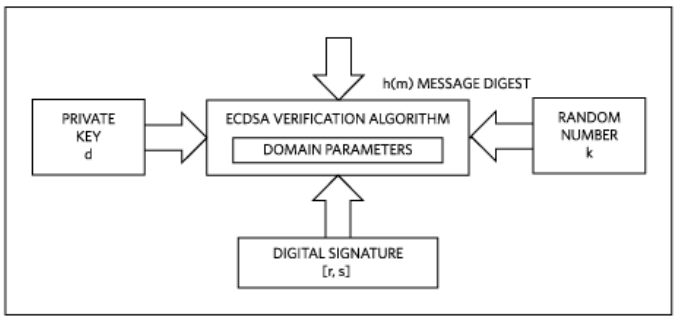


FIGURE IV: Signature Computation Process

Correcteness of the Algorithm:

Even the function is correct its correctness is checked. Let C the curve point, then

- $C=u_1 * G + u_2 * Q_A$
- Since, $Q_A=d_A * G$. Thus, $C=u_1 * G + u_2 * d_A * G$
- $C = (u_1 + u_2 * d_A)$
- Putting value of u1 and u2 from avobe in this equation, we get $C=(zs^{-1} + rd_As^{-1}) * G$
- $C = (z + rd_A)s^{-1} * G$
- Substituting the value of 's' ,we get $C = (z + rd_A) (z + rd_A)^{-1}(k^{-1})^{-1} * G$
- C = k * G

Thus it is verified that a message that is correctly signed is verified correctly.

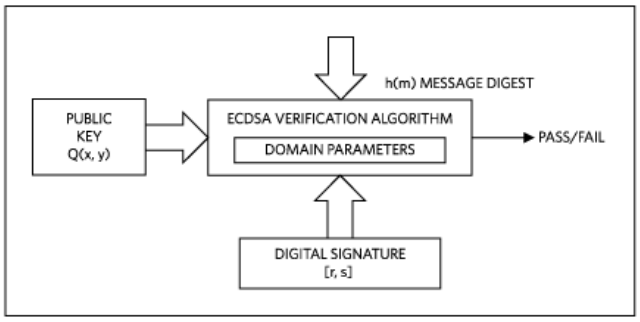


FIGURE V: Signature Verification Process

B. Security Level of ECDSA

The objective of security of ECDSA is unforgeable against random choosen message attack. The goal of ECDSA is to provide a better signature that can't be attacked by random generation of prime numbers. Against digital signature there are two attacks, one attack against the key also known as public key and another against the signature that they can get access through week loop hole. Applying ECDSA proves itself secure because hash function developed using it is collision resistant. Any attack that does not uses Elliptical Curve finds collision in the hash function. Essential security condition for ECDSA are:

- The algorithm used in calculation of 'G' is very hard to crack. That shows that one can't easily get solution to proposed algorithm and obtain security key.
- The function called hash used is mainly a one way hash function. This means that one cannot reverse the process to acquire the encrypted signature.
- The way 'k' is generated is unpredictable. The secret key 'k', 'r', and 's' cannot be obtained without it.

In the situation of ECDSA, the entire concentration of attacker is to attack against hash function used within signature generation. Thus, ECDSA is the best way to provide better security along with ECC on HTTP over SSL.

IX. Conclusion

ECDSA which is an important part of ECC proposed for an alternative to security in the field of Digital Signature. The main reason that leads to suggest alternative that there is not any method that can solve elliptical curve discrete algorithm problem for any chosen elliptical curve. Thus, it has better methodology to solve the problem of digital signature arises under ECC. The generated digital signature using ECDSA is highly secure and it also consumes very less bandwidth because very less key size used by the elliptical curves. In comparison to others, ECDSA uses small parameters than other competitive methodology like RSA with higher level of security. Benefits provided by smaller key size also helps solving problem of processing power reduction, less storage space consumption, less bandwidth usages faster computation time. All these benefits makes ECDSA also an ideal methodology for pages, PDAs, and also for recent smart cards. Although, choices for algorithm are based on the needs of the system and its supporting hardware. But still ECC is preferred over RSA for asymmetric cryptography. There are few drawbacks of ECC also but if we consider its efficiency compared to what RSA performed for larger server and application, ECC will be preferred. ECC just needs more research to look into its never ending dimensions that can be useful for providing a better security for exchanging keys over SSL. With a better support from US government, ECC is still improving. It is used widely for securing, maintaining and authenticating the information over wireless communication medium in field of defense and military strategies.

Reference

1.Tulika Shubh, **"Man-In-The-Middle-Attack Prevention Using HTTPS and SSL",** International Journal of Computer Science and Mobile Computing, IJCSMC, Vol. 5, Issue. 6, June 2016, pg.569 – 579

2.Rolf Oppliger, Ralf Hauser, David Basin, **"SSL/TLS session-aware user authentication – Or how to effectively thwart the man-in-the-middle"**, Department of Computer Science, ETH Zurich, Haldeneggsteig 4, CH-8092 Zu¨ rich, Switzerland, Computer Communications 29 (2006) 2238–2246.

3. Yaoqi Jia, Yue Chen, Xinshu Dong, Prateek Saxena, Jian Mao, Zhenkai Liang, **"Man-in-the-browser-cache: Persisting HTTPS attacks via browser cache poisoning"**, School of computing, National University of Singapore, 13 Computing Drive, COM1 #3-27, Singapore