Computer Systems Science & Engineering
DOI:10.32604/csse.2022.018166
Article

Tech Science Press

Secure and Light Weight Elliptic Curve Cipher Suites in SSL/TLS

B. Arunkumar* **and G. Kousalya**

Coimbatore Institute of Technology, Coimbatore, 641014, India
*Corresponding Author: B. Arunkumar. Email: aruncit17@gmail.com
Received: 27 February 2021; Accepted: 27 April 2021

Abstract: In the current circumstance, e-commerce through an online banking system plays a significant role. Customers may either buy goods from E-Commerce websites or use online banking to move money to other accounts. When a user participates in these types of behaviors, their sensitive information is sent to an untrustworthy network. As a consequence, when transmitting data from an internal browser to an external E-commerce web server using the cryptographic protocol SSL/TLS, the E-commerce web server ensures the security of the user's data. The user should be pleased with the confidentiality, authentication, and authenticity properties of the SSL/TLS on both the user's web browser and the remote E-commerce web server. E-Commerce web servers should choose the best SSL/TLS cipher suites for negotiating the user in order to attain such optimistic scenarios, as the cipher suite used in SSL/TLS plays an important role in securing E-Commerce web servers. The paper primarily focuses on analyzing the SSL/TLS cipher and elliptic curves. The paper also recommends the best elliptic curve cipher suites for E-Commerce and online banking servers, based on their power consumption, handshake execution time, and key exchange and signature verification time.

Keywords: Elliptic curve; cipher suites; secp256r1; curve25519; ed25519

1 Introduction

The internet is the most important and fundamental component of any trending technology. E commerce plays a critical role in today's technological evolution, making significant contributions to e-shopping and online banking. Since the application is configured to access the web server via a web browser using the SSL/TLS protocol, such e-commerce applications rely on unauthorized web browsers. Confidentiality, integrity, and authentication should all be preserved in the information/communication flow between the web server and the web browser. Various cryptographic methods, which are broadly known as symmetric and asymmetric algorithms, may be used to ensure certain security parameters in the framework. Despite the fact that these algorithms are used in various OSI layers, the paper focuses on the security to be implemented in the application and transport layers, as online banking and e-shopping applications use SSL/TLS in the transport layer to migrate the most confidential data. SSL/TLS protection is achieved by combining symmetric and asymmetric algorithms. TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA256, also known as Cipher Suite, is a common security technique for securing communication [1]. The Cipher Suite technique could be enhanced even further by using a range of symmetric and asymmetric algorithms to achieve maximum security with minimal key bit processing. Owing to the widespread use of mobile devices for e-commerce, Cipher Suites should be concerned about compatibility operating mode in mobile without breaching security standards, as well as minimized processing time, clock cycles, and memory utilization. The paper discusses a detailed analysis on Safe curves and Safe primes on elliptic curve cryptography that improves the efficiency and performance of the Cipher Suites, taking into account the needs of secured online e-banking and e-commerce. In addition, the paper addresses numerous recommendations for underlying software usability, usage, adaptation, and versatility. SSL protocols were created by in response to the growing demand for secure E-commerce. Netscape gave it the name SSL 1.0, which was later changed to SSL 2.0 and SSL 3.0 over the course of a year. Since SSL 3.0 was so close to Paul Kocher's protocol design, it was revised and renamed TLS. Originating with the Secured Socket Layer (SSL), which secures client-server communication, and evolving into Transport Layer Security (TLS), all lead to SSL/TLS security in protecting the most sensitive data during communication. SSL/TLS is a protocol for securing traffic between an unauthenticated web browser and a trusted.

RFC 2246 was the first to design and describe the TLS protocol, which was called TLS 1.0 and resembled SSL 3.0 [2]. TLS 1.0's cipher suite is built on the RSA algorithm. RFC 4346 described the upgrade from TLS 1.0 to TLS 1.1 [3]. TLS 1.1 protects against the cipher-block chaining (CBC) attacks that were present in SSL 3.0. This also aimed to provide Forward Confidentiality (a solution that does not compromise past session keys if long-term keys are compromised) and Authenticated Encryption with Related Data (AEAD) [4]. RFC 5246 described TLS 1.1, which was later revised to TLS 1.2. TLS 1.2 adds security to authenticated encryption ciphers, which are mostly used in the Advanced Encryption Standard's Galois/Counter Mode (GCM) and Cipher block chaining-message authentication code (CCM) modes. RFC 6176 further refined all TLS versions [5]. TLS 1.3 is a working version that primarily focuses on providing security through the use of a combination of symmetric and asymmetric algorithms that provide optimal security with minimal key bit processing. The handshake protocol and the record protocol are also included in every implementation of the TLS protocol. The handshake protocol authenticates the initialization, certificate, and finish messages and uses them to authenticate the communicating parties. The standard port number of 443 for HTTPS should be used to implement Client| Server communication using SSL/TLS. Continuing with the HTTPS configuration, SSL/TLS deals with two layers of service, namely the handshake layer and the record layer, which include the handshake protocol and the record protocol on each operation. The general design structure of the cipher suite and their layers of operation are depicted in Fig. 1.

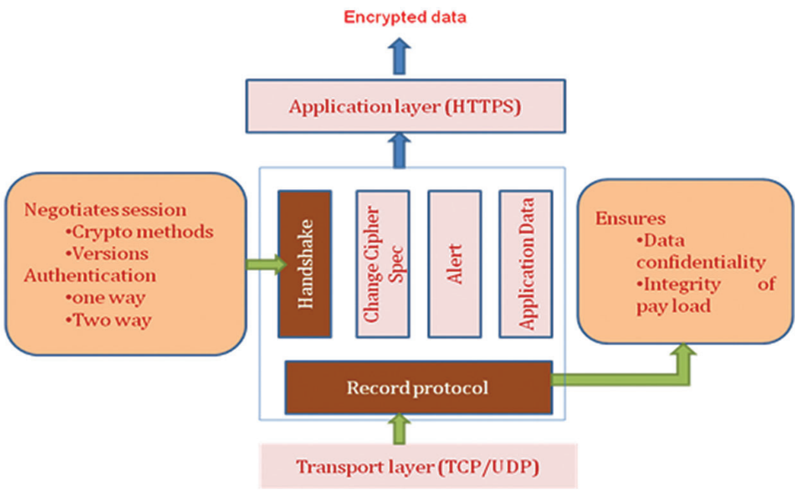

Figure 1: TLS cipher suite architecture

As shown in Fig. 2, the TLS handshake protocol is perhaps the most important step in establishing safe communication between a web browser and a web server. The handshake protocol is in charge of three main tasks: cipher suite negotiation, server authentication, and client and session key agreement and exchange.

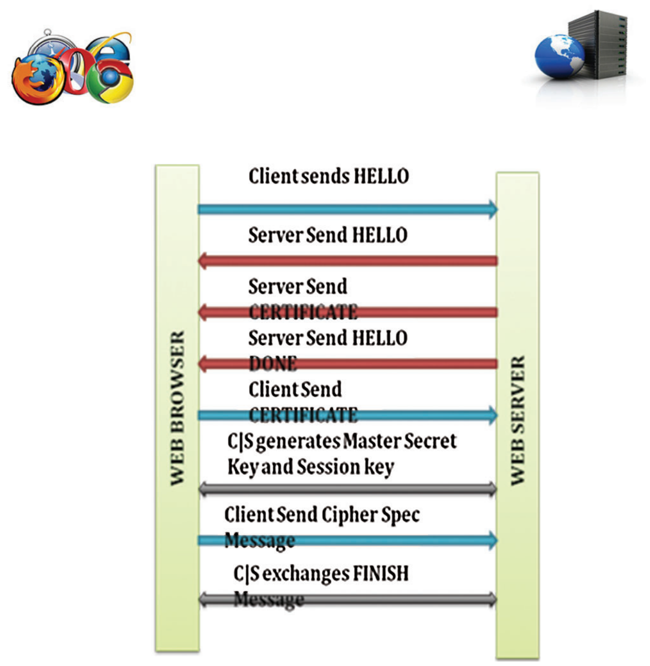

Figure 2: Handshake layer operations

i) The client sends the initial HELLO packet, which contains the SSL/TLS version, random bytes, cipher suites, compression algorithms, and extensions tags, in order to evaluate the handshake protocol.
ii) After receiving the client's encrypted HELLO packet, the server responds with a HELLO packet containing random bytes, cipher suits, compression algorithms, and extensions tags.
iii) After sending the HELLO packet successfully, the server will send the CERTIFICATE command and the HELLO DONE packet to the client in that order.
iv) If the server requests it, the client will give the CERTIFICATE command.
v) The client then generates a random pre-master secret key and sends it to the server in an encrypted format, using the same public key that was used to encrypt the CERTIFICATE.
vi) Based on the pre-master secret key generated and shared by the client, the server and client each generate a Master secret key and a session key.
vii) The server and client each generate a Master secret key and a session key based on the pre-master secret key created and shared by the client.
viii) Finally, both the client and the server exchange a FINISH packet to indicate the start of record layer communication.

The record layer is responsible for managing of securely transmitting data in a tightly encrypted format. The record layer secures communication by segmenting incoming data into 64-bit, 128-bit, or 256-bit segments, depending on the symmetric algorithms (block cipher and Stream cipher algorithms) used for encryption. The record layer uses MAC algorithms to ensure the confidentiality of the data segments after receiving the encrypted segments at the receiving.

2 Related Works

SSL/TLS protocol seems to be the most commonly used protection mechanism, and cipher suites are the most common. The Cipher suite has four main features: key exchange algorithms, authentication algorithms, encryption algorithms, and message authentication code algorithms. The Key Exchange Algorithms are responsible for secure key exchange between the sender and recipient (say Client and Server). RSA, DH, ECDH, and ECDHE are some of the most widely used key exchange algorithms in the cipher suite. RSA, DSA, and ECDSA are used in the authentication algorithms to ensure the sender and receiver's authenticity. Encryption algorithms are used to encrypt data transmitted between a web browser and a web server using encryption algorithms such as AES and DES. Using MD5, SHA1, SHA256, SHA384, and POLY1305, the Message Authentication Code algorithm guarantees the integrity constraints on both.

SSL 3.0 Cipher suites began with the SSL_DH_RSA_EXPORT_WITH_DES40_CBC_SHA, SSL_DH_RSA_WITH_DES_CBC_SHA, and SSL_DHE_DSS_WITH_DES_CBC_SHA specific suites [6]. Cipher Block Chaining (CBC) is the most common technique used in all cipher suites that have evolved. The key flaw in these CBC techniques is the poodle attack, which causes SSL 3.0 to break. The poodle attack was made possible by the encrypted initialization vector (IV) used in the chaining phase in CBC, which was implemented using the DES algorithm and kept 2^{56} combinations for the attacker [7]. TLS 1.0, the cipher suite used in this suite, is very similar to SSL 3.0, but without the issues that SSL 3.0 has. This was achieved by altering the encryption method used to encrypt the Initialization vector. The modification was based on a better implementation of symmetric algorithms, such as AES to IV, which gives the intruder a combination of 2^{128}, 2^{192}, and 2^{256} combinations. However, the cipher suite in TLS 1.0 was also prone to a poodle attack as well as a BEAST attack [8]. By replacing the implicit initialization vector with an explicit initialization vector, the cipher suite built for TLS 1.1 protected against all CBC attacks. TLS1.2 is the cipher suite used for the majority of today's web browsers and websites. The elliptic curves cryptographic techniques for exchanging keys and authenticating end users/browser/server are a significant and notable implementation of TLS1.2 that promises security problems in our day-to-day security issues. MD5/SHA1, which was used in TLS1.1 to verify the message's integrity, was replaced with SHA256, SHA384, and HMAC SHA 256 to enhance the message's integrity even further. The encryption algorithm used in TLS 1.2 is AES_GCM and AES_CCM. The best cipher suites of TLS 1.2 are TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256, TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384. TLS 1.3 is still under draft with much advancement like choosing a safe elliptic curves and safe primes [9]. The progress is also aided by the removal of older authentication algorithms such as MD5 and SHA224, as well as key exchange algorithms such as DH, ECDH, and ECDHE. The cipher suite also includes the CHACHA20 stream cipher with POLY1305, a new authentication encryption technique [10]. To ensure authentication, the Edwards curve digital signature algorithm (EdDSA) Ed25519 and Ed448 are used [11]. For key exchange, the suite employs Curve X25519 and Curve X448 [12]. In addition, TLS 1.3, the RTT value is reduced to 1 RTT, while in TLS 1.2, the RTT value is reduced to 2 RTT for negotiating the handshake operations between the web server and the web browser. When the same web site is accessed for the second time, a 0 RTT delay is guaranteed. TLS 1.3 uses the AEAD system, which combines the key exchange, authentication, and message authentication processes into a single handshake. The calculation time involved in the handshake process would be reduced as a result of this.

POODLE (Padding Oracle On Downgraded Legacy Encryption) attack, BEAST (Browser Exploit Against SSL/TLS) attack, CRIME (Compression Ratio Info Leak Made Easy) attack, BREACH (Browser Reconnaissance And Exfiltration Via Adaptive Compression Of Hyper Text) [13], Heart Bleed [14], Downgrade attack [15], Lucky13 attack [16] and RC4 attack [17] are some of the most popular SSL/TLS attacks. Any e-banking or e-commerce website that uses SSL/TLS communication will be subject to attacks due to the issue of using older versions of SSL/TLS in the web browser without upgrading it to the newer SSL/TLS versions adopted by the web server.

In their blog, Daniel Bernstein and Tanje Lange discuss secure curves and several principles for selecting curves for use in elliptic curve cryptography (ECC). Safe curves' review of norms and official documents shows that elliptic curve discrete logarithm (ECDLP) problem security is difficult, but not ECC security. According to Daniel Bernstein and Tanje Lange, elliptic curves designed to be ECDLP safe only if the attacker does not breach ECDLP's protection by producing incorrect results for certain unusual curve points, leaking secret data when the input is not a curve point, and leaking secret data through branch and cache timing attacks. As a result, the authors claim that none of these requirements do a good job of ECC security, which is applicable to ECDLP security. The author proposed new elliptic curves, which achieve improved protection and efficiency in ECC and ECDLP, based on a notable problem in ECC security, as shown in Tab. 1.

Table 1: Security parameters of famous elliptic curves

Curve	Parameters				ECDLP Security				ECC security			
	Safe	Field	Equ	Base	Rho	Transfer	Disc	Rigid	Ladder	Twist	Complete	ind
NIST P-256	Y	Y	Y	Y	Y	Y	Y	N	N	Y	N	N
Cruve25519	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Different elliptic curves from previous standards were evaluated by Daniel Bernstein and Tanje Lange based on the following security conditions, curve parameters, ECDLP security, and ECC security. The secure curve security specifications are divided into three categories: a) Basic curve parameters, b) ECDLP security, and c) ECC security.

3 Analysis of Cipher Suites

TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 and TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 are the most common cipher suites now used in e-commerce and online banking. A series of key exchange algorithms, authentication algorithms, encryption algorithms, and MAC algorithms were used to create this. TLS_RSA_WITH_AES_128_GCM_SHA256 was the first cipher suite in the TLS 1.2 cipher suite. The evolution of cipher suites from the simple adapted cipher suite is depicted in Fig. 3 [18].

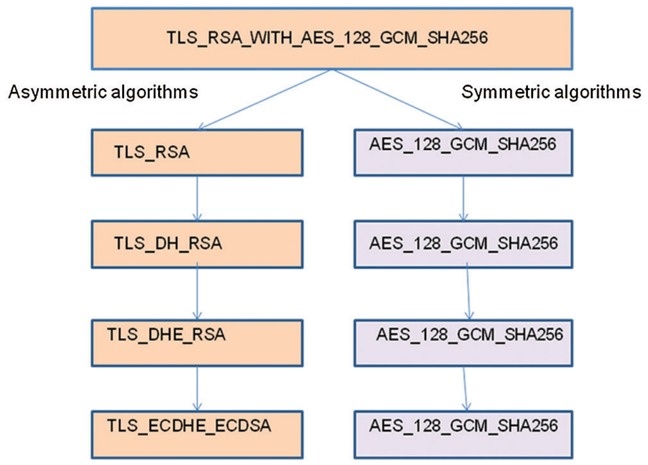

Figure 3: Evolution of SSL/TLS cipher suite

For their hidden session key encryption, SSL and TLS used the RSA algorithm when they first began protecting the web browser and web server. When selecting a powerful or safe prime number, this RSA uses a modular exponential approach in which the factors are difficult to identify. This is due to the fact that when selecting a strong prime number, the bits of representation begin at 2048 bits for the lowest strong prime number, ensuring greater confidentiality. While 2048 bits is a good number for ensuring privacy, it can be hacked with enough effort. As a result, attempts were made to enhance confidentiality at this stage of leakage. As a result, RSA changed the lowest strong prime number representation to 3072 [19]. The issue that arose was the memory requirement for storing the key, which was an even bigger issue for mobile devices. Even though several theoretical attacks are possible in RSA, the realistic timing attack should be more concerned because it can fully break the communication mechanism [20]. The timing attack is also known as known cipher text attack/side channel attack because of its operation.

In the key exchange component, RSA was replaced by Diffie Hellmen (DH) to ensure the authentication of the key exchange between the web browser and the web server [21]. The DH is more immune to mathematical and timing attacks because it employs the discrete logarithmic method (DLM). While the defendant's attack plan has been improved, the memory requirement problem remains a flaw. TLS_DH_RSA has big issues with LogJam attacks and Freak [22]. The DLP is to compensate for this log jam attack [23,24]. The main issue in DH is that the past session key may be compromised in the future. TLS_DHE_RSA (Ephemeral DH or DHE) includes Forward Secrecy (FS) methods to eliminate this issue [25]. For each session, the FS generates a unique session key (as decided by the Web Server).

Given the shortcomings of previous cipher suites, especially the time and memory requirements, TLS 1.2 was designed to include Elliptic curve cryptography, which included elliptic curve (EC) in their cipher suites for improved security in e-commerce applications, as shown in Fig. 4. The ECC is based on the ECDH system for TLS1.2 [26]. The DLP is the most difficult problem in this ECDH.

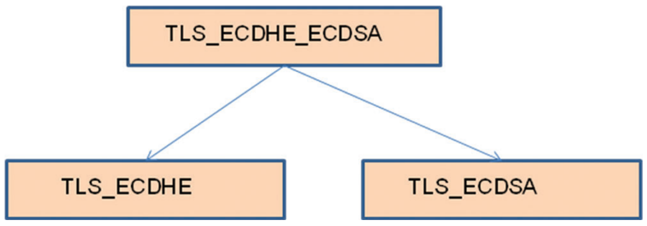

Figure 4: SSL/TLS Key exchange and Authentication Components

The basic ECDH is based on DH with a static key exchange. When comparing ECDH and ECDHE, ECHDE generates unique keys for each session, and the same key is never generated twice. The ECDHE has a drawback known as the Elliptic Curve Discrete Logarithm Problem (ECDLP), which TLS 1.1 failed to satisfy [27]. Despite this forward secrecy advantage, the ECDHE has a drawback known as the Elliptic Curve Discrete Logarithm Problem (ECDLP) [28]. The ECDLP issue is caused by the Discrete Logarithm (DL) being broken using the baby-step, giant-step, and Pollard-Rho methods [28]. The ECDSA technique uses a digital signature algorithm to provide authentication. DSA is an asymmetric algorithm that uses an elliptic curve to ensure web server and browser authentication. This is based on the Discrete Logarithmic process, which ensures a security level of 2^{80}. As all the current e-commerce application and online banking servers rely on elliptic curves, this paper deals in analyzing the various elliptic curves and their implications.

An Elliptic curve, E over a value k can be defined as follows [29]

1. A non-singular projective plain curve E over k of degree 3, together with a point O belongs to $E_{(k).}$

2. O is requires as a point of modulation.
3. A non-singular projective plain curve over k of the form as specified as Eq. (1),

$$y^2z + a_1xyz + a_3yz^2 = x_3 + a_2x_2z + a_4xz + a_6z^3 \tag{1}$$

4. A non-singular projective curve E of genus 1 together with the point of O belongs to E_k

Any elliptic curve over a finite field can be classified as an elliptic curve over GF (P) or an elliptic curve over GF (2m). Three subcategories of elliptic curves can be formed using the standard elliptic curve equation: Weierstrass curve Eq. (2), Montgomery curve Eq. (3) and Edwards curve Eq. (4). These equations take the form,

$$y^2 = x^3 + ax + b \tag{2}$$

where $4a^3 + 27b^2$ is non-zero,

$$by^2 = x^3 + ax^2 + x \tag{3}$$

where $b(a^2 - 4)$ is non-zero and

$$x^2 + y^2 = c^2(1 + dx^2y^2) \tag{4}$$

where d(1-d) is non-zero respectively.

In order to provide more security in e-commerce and online banking servers while also requiring less processing time, CPU cycles, and memory bandwidth, the applications must choose a better elliptic curve based on the properties of the three curves mentioned above. According to the NIST standard, any point of order n can be used as the starting point. A sample base point G is given for each curve (Gx, Gy). In light of this definition, users may wish to select their own base points in order to maintain cryptographic separation. Tab. 2 specifies the random elliptic curve domain parameters over secp256r1.

Table 2: Domain parameters of Secp256r1

P= $2^{224}(2^{32}\text{-}1) + 2^{192} + 2^{96}$
The curve E: $y^2 = x^3 + ax + b \bmod p$ over F_P defined by:
a = FFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF FFFFFFFF FFFFFF
b = 5AC635D8 AA3A93E7 B3EBBD55 769886BC 651D06B0 CC53B0F6 3BCE3C3E 27D260
seed S = C49D3608 86E70493 6A6678E1 139D26B7 819F7E
G = 04 6B17D1F2 E12C4247 F8BCE6E5 63A440F2 77037D81 2DEB33A0 F4A13945 D898C296 4FE342E2 FE1A7F9B 8EE7EB4A 7C0F9E16 2BCE3357 6B315ECE CBB64068 37BF51
n = FFFFFFFF 00000000 FFFFFFFF FFFFFFFF BCE6FAAD A7179E84 F3B9CAC2 FC632551
h = 01

Due to the various parameters in curve secp256r1, e-commerce applications now typically append SECP256r1 or NIST P256 in a random prime field curve that was built from the base of a short Weierstrass curve that relies on elliptic curve over GF (P).

Using the Montgomery ladder and the NIST P256 curve (secp256r1), we can perform fast scalar multiplication and addition. Pollards Rho can achieve this SECG256R1 curve, but it is 2^{128} times more challenging. Reliable internet transmission using the TLS_ECDHE_ECDSA cipher suite for key exchange and authentication using P256 or Secp256r1. The e-commerce framework and web browser should drift away from this normal curve. Most webservers and web browsers that use TLS 1.3 use Curve 25519 to perform key exchange in security protocols with maximum efficiency. The curve uses the prime field of this curve is $P = 2^{255} - 19$ and the curve used is Montgomery curve, $y^2 = x^3 + 486662x^2 + x$. This is due to the fact that Curve 25519 is a de-facto P256 that is used in a variety of applications. This curve has a number of benefits, including the lack of timing attacks, the use of short hidden keys (32 bytes) and short public keys (32 bytes), and the fact that it is based on free key validation. The twisted Edwards curve used in the Ed25519 signature scheme is bi-directionally equivalent to the Curve 25519 for key exchange. For signature verification, the security of curve Ed25519 is 2^{200}, while curve X25519 is 2^{128}.

4 Implementation and Results

The output of various elliptic curve cipher suites that help forward secrecy is examined in this section. The following elements were used to perform and test the experiments for different elliptic curves:

a. Energy consumption on both client/server

b. Handshake completion time between web browser and web server

c. CPU cycles/operations of various elliptic curve operations

d. Operations/seconds for elliptic curve key exchange and signature verification

Popular elliptic curves that were used in E-commerce and online banking servers are collected and tabulated in Tab. 3 based on the four elements listed above.

Table 3: E-commerce and online banking cipher suites

NO	CIPHER SUITES	KEY EXCHANGE CURVE	SIGNATURE VERIFICATION	FORWARD SECRECY	Cipher suite
1	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	secp256r1	RSA 3072	YES	TLS 1.2
2	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	secp256r1	secp256r1	YES	TLS 1.2
3	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	Curve25519	Ed25519	YES	TLS 1.2
4	TLS_AES_128_GCM_SHA256	Curve25519	Ed25519	YES	TLS1.3

The evaluation was carried out using the new openssl and libsodium test beds, as well as the various elliptic curves used in TLS 1.2 and TLS 1.3 cipher suites, as shown in Tab. 3, on a platform Intel® CoreTM i7-8850H Processor at 2.60 GHz running Ubuntu 18.04.

The goal of the research was to determine the amount of energy used by the client and server during initial data transmission during elliptic curve key exchange and signature verification. The effects of various elliptic curve handshake methods, as well as their power consumption, are shown in Fig. 5. Curve 25519/Ed25519 outperforms other elliptic curves used in TLS 1.2 cipher suites, according to the results. Curve25519/Ed25519 also outperforms in TLS 1.3 cipher suites during the handshake process between the client browser and the web server, according to the results. The above result shows that curve25519/Ed25519, which uses elliptic curves, outperforms secp256r1 and RSA on both the client and server sides, as well as achieving security 2^{128}.

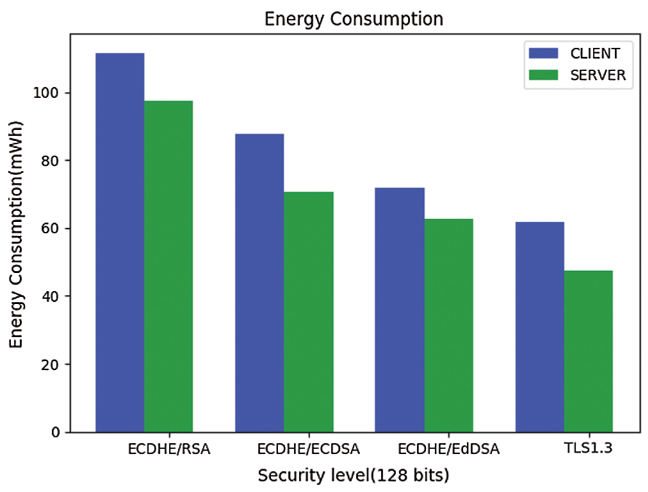

Figure 5: Energy consumption for various Elliptic curves

The number of CPU cycles spent on different elliptic curves used to exchange the key between the web browser and the web server are shown in Fig. 6. The curve25519 used in ECDHE/EdDSA performs well in contrast to other elliptic curves used in ECDHE/ECDSA and ECDHE/RSA in TLS 1.2 and TLS 1.3 cipher suites based on different elliptic curve operations. When comparing TLS 1.3 cipher suites to TLS 1.2 cipher suites, the curve25519/Ed25519 provides better CPU cycles operations.

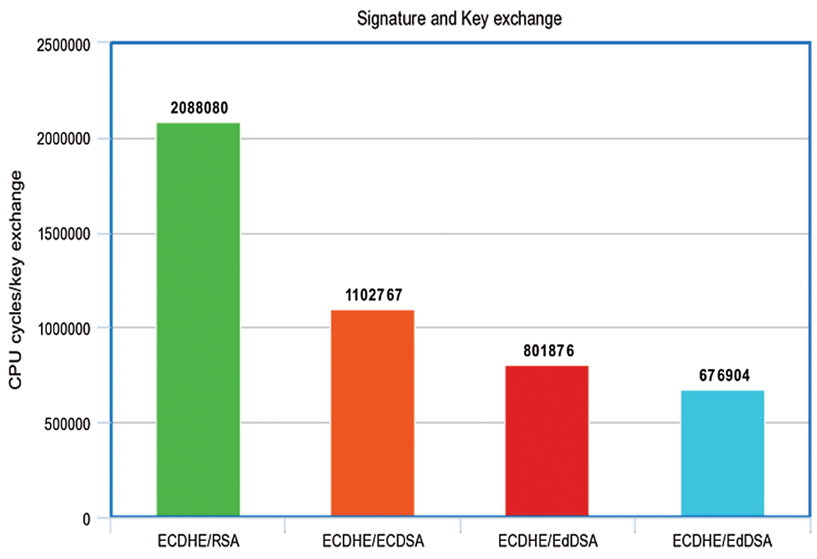

Figure 6: Signature and key exchange CPU cycles/operations

The product of handshake completion time of various elliptic curves used in TLS 1.2 and TLS 1.3 cipher suites is shown in Fig. 7. Curve25519/Ed25519 and secp256r1 are elliptic curves that are used for different elliptic curve equations and perform the handshake process between the web browser and the web server. As compared to other elliptic curves used in TLS 1.2 cipher suites that also achieve the security of 2^{128}, curve25519/Ed25519 performs better. ECDHE/EdDSA also performs well in both TLS 1.2 and TLS 1.3 cipher suites, according to the results.

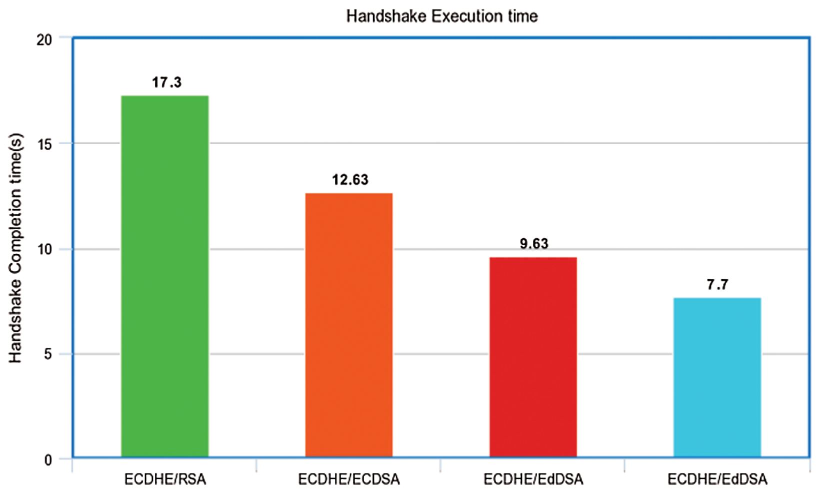

Figure 7: Handshake execution time for various elliptic curves

For various elliptic curves used in TLS 1.2 and TLS 1.3 cipher suites, Fig. 8 shows the performance of ECDSA signing, ECDSA verify, EdDSA signing, EdDSA verify, and ECDHE compute key. Higher is better for key compute, signature signing, and verification of all elliptic curves in TLS 1.2 and TLS 1.3 cipher suites, according to the results of various elliptic curves. The above findings show that in TLS 1.2 and TLS 1.3 cipher suites, curve25519/Ed25519 using key compute and signature signing and verification outperforms secp256r1and RSA key compute and signature verification.

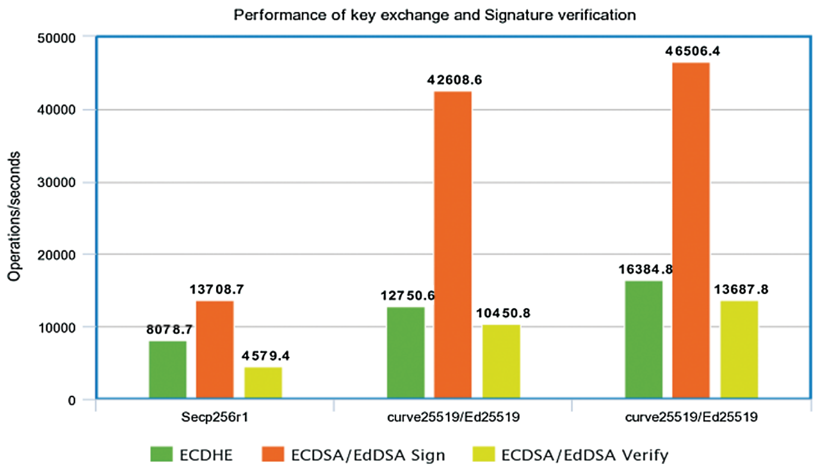

Figure 8: Performance of key exchange and verification on various elliptic curves

5 Conclusion

The best type of elliptic curve to choose, computations used for the curves, and base point to fix on curves for better security results in e-commerce applications were all investigated in this paper. Standard Elliptic curves are used in e-commerce applications, such as 1) curves over prime fields GF(P) –P-192, P-224,P-256,P-384,P-521. 3) Curve25519/Ed25519 2) Secp256r1. Since bulk encryption (symmetric cipher) currently operates only on AES 128 and the bit values specify better application protection, Secp256r1 and Curve25519 is the prime curve that is commonly used in most e-commerce and online banking. Also, at 2^{128}, the curves Secp256r1 and Curve25519 will provide better security against

Pollard's Rho method, but bulk encryption (symmetric cipher) will use AES 128. When it comes to curves, since prime curves are faster on general-purpose CPUs and use a Giant integer multiplier circuit, the Montgomery curve is the most recently used curve for SSL/TLS, and it provides better security 2^{128}. The Montgomery curves X25519 and secp256r1 are considered the fastest curves in ECC since they compute the points on the EC using the Montgomery ladder (constant time computation) rather than the point multiplication method used in the short Weierstrass curve. As it takes the form of a Montgomery curve, like Montgomery ladder, the Twisted Edwards curve, such as Ed25519, can be considered one of the fastest curves (mixed addition and mixed differential addition). After evaluating the results based on the above elliptic curves, it was determined that the curve25519/Ed25519 outperforms all other curves used in most E-commerce and online banking servers in TLS 1.2 and TLS 1.3 cipher suites in the performed and evaluated results. As a result, curve25519/Ed25519 is recommended as one of the best elliptic curves in the TLS 1.2 and TLS 1.3 cipher suites used in E-commerce and online banking servers.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] S. Blake-Wilson, N. Bolyard, V. Gupta, C. Hawk and B. Moeller, "Elliptic curve cryptography (ECC) cipher suites for transport layer security (TLS)," 2006.

[2] T. Dierks and C. Allen, "The TLS protocol version 1.0," *RFC2246,* 1999.

[3] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) protocol version 1.1," *RFC 4346,* 2006.

[4] T. Dierks and E. Rescorla, "The transport layer security (TLS) protocol version 1.2," *RFC 5246,* 2008.

[5] E. Rescorla and T. Dierks, "The transport layer security (TLS) protocol version 1.3," 2018.

[6] A. Freier, P. Karlton and P. Kocher, "The secure sockets layer (SSL) protocol version 3.0," *RFC 6101,* vol. 11, 2011.

[7] B. Möller, T. Duong and K. Kotowicz, "This POODLE bites: Exploiting the SSL 3.0 fallback," *Security Advisory*, vol. 21, pp. 34–58, 2014.

[8] S. Lucks, "BEAST: A fast block cipher for arbitrary blocksizes," in *Communications and Multimedia Security II*, Boston, MA: Springer, pp. 144–153, 1996.

[9] D. J. Bernstein and T. Lange, "SafeCurves: Choosing safe curves for elliptic-curve cryptography," 2014. [Online]. Available at: https://safecurves.cr.yp.to.

[10] A. Langley, W. Chang, N. Mavrogiannopoulos, J. Strombergson and S. Josefsson, "ChaCha20-Poly1305 cipher suites for transport layer security (TLS)," *RFC 7905,* pp. 1–7, 2016.

[11] S. Josefsson and I. Liusvaara, "Edwards-curve digital signature algorithm (eddsa)," *Internet Research Task Force, Crypto Forum Research Group, RFC*, vol. 8032, pp. 257–260, 2017.

[12] D. J. Bernstein, "Curve25519: New Diffie-Hellman speed records," *International Workshop on Public Key Cryptography*, Berlin, Heidelberg: Springer, pp. 207–228, 2006.

[13] Y. Gluck, N. Harris and A. Prado, "BREACH: Reviving the CRIME attack," *Unpublished manuscript,* 2013.

[14] M. Carvalho, J. DeMott, R. Ford and D. A. Wheeler, "Heartbleed 101," *IEEE Security & Privacy*, vol. 12, no. 4, pp. 63–67, 2014.

[15] S. Lee, Y. Shin and J. Hur, "Return of version downgrade attack in the era of TLS 1.3," in *Proc. of the 16th Int. Conf. on emerging Networking experiments and Technologies*, pp. 157–168, 2020.

[16] G. Razoqui, M. S. Inci, T. Eisenbarth and B. Sunar, "Lucky 13 strikes back," in *Proc. of the 10th ACM sym. on information, computer and communications security*, pp. 85–96, 2015.

[17] A. Klein, "Attacks on the RC4 stream cipher," *Designs, Codes and Cryptography*, vol. 48, no. 3, pp. 269–286, 2008.

[18] P. Sirohi, A. Agarwal and S. Tyagi, "A comprehensive study on security attacks on SSL/TLS protocol," in *2nd Int. Conf. on Next Generation Computing Technologies*, IEEE, pp. 893–898, 2016.

[19] J. Jonsson and B. S. Kaliski, “On the security of RSA encryption in TLS,” in *Annual Int. Cryptology Conf.*, Berlin, Heidelberg: Springer, pp. 127–142, 2002.

[20] D. Bleichenbacher, “Chosen cipher text attacks against protocols based on the RSA encryption standard,” in *Annual Int. Cryptology Conf.*, Berlin, Heidelberg: Springer, pp. 1–12, 1998.

[21] E. Bresson, O. Chevassut and D. Pointcheval, “Provably secure authenticated group Diffie-Hellman key exchange,” *ACM Transactions on Information and System Security (TISSEC)*, vol. 10, no. 3, pp. 10, 2007.

[22] F. Kohlar, S. Schäge and J. Schwenk, “On the Security of TLS-DH and TLS-RSA in the Standard Model,” *IACR Cryptol. ePrint Arch*, pp. 367, 2013.

[23] P. C. Kocher, “Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems,” in *Annual Int. Cryptology Conf.*, Berlin, Heidelberg: Springer, pp. 104–113, 1996.

[24] N. P. Smart, “The discrete logarithm problem on elliptic curves of traces one,” *Journal of Cryptology*, vol. 12, no. 3, pp. 193–196, 1999.

[25] T. Ager, F. Kohlar, S. Schäge and J. Schwenk, “On the security of TLS-DHE in the standard model,” in *Annual Cryptology Conference*, Berlin, Heidelberg: Springer, pp. 273–293, 2012.

[26] T. Jager, J. Schwenk and J. Somorovsky, “Practical invalid curve attacks on TLS-ECDH,” in *European Symposium on research in computer security*, Cham: Springer, pp. 407–425, 2015.

[27] S. Vaudenay, “The Security of DSA and ECDSA,” in *International workshop on public key cryptography*, Berlin, Heidelberg: Springer, pp. 309–323, 2003.

[28] P. L. Montgomery, “Speeding the Pollard and elliptic curve methods of factorization,” *Mathematics of Computation*, vol. 48, no. 177, pp. 243–264, 1987.

[29] I. Blake, G. Seroussi and N. Smart, “Elliptic curves in cryptography,” Cambridge university press, vol. 265, 1999.