CHAPTER 5

EFFECTIVE SIGNCRYPTION APPROACH FOR SECURE CONVENTION FOR MULTILAYER CONSENSUS USING ECC

The used algorithm in cryptography represents the facts for computation and/or computation costs in general. The motivation for any problem is the primitive generators that make the protocol a big advantage over the technology augmentation. This chapter presents a methodological approach on session specific challenge-response protocol for a better, improved and stronger security on reduced costs. The basic primitives are applied on Diffie-Hellman and Elliptic Curve Cryptography. The purpose is providing the security properties for protocol compositional logic that focuses on privacy rights in information assessment in multidisciplinary obligations. In addition, we portrait a signcryption approach for password authenticated key exchange protocol for multilayer consensus, which logically combines individual signature and encryption cost in the form of reduced computational cost and communications cost in single stride of operation. The overall computation time potentially is reduced for the proposed methodology on signature and key generation. The results for ECC based multilayer consensus of key generation approach are tested on Automated Validation of Internet Security Protocol Architecture (AVISPA) tool and SPAN tool. Further, by preserving the definition of signcryption, we enhanced the same scheme in comparison to the other proposed schemes.

5.1 INTRODUCTION

The challenge-response (C-R) protocols are one the recent research trends in cryptography; the mathematical modeling is moved around the process calculus. It is included the actions to generate new random numbers, perform encryption and signature, send or/and receive messages, finally performing decryption with verification with matched digital signature. The security proofs allow the protocols using combining of their independent proofs in parts. Secure composition as it designs in such a way that may not degrade and does not affect its own existing security so it has considered itself as a difficult security problem. This philosophy is more amenable for automation of security protocol analysis, where the cryptographic assumption considered being perfect in speedup in computation costs and communication costs, energy
minimization, respective applicability for applications, etc. The major thing is in protocol derivational logic is to develop further derivational system approach on behalf of logical methods, where the protocol analysis concerns to the soundness theorems. Datta [93] has presented an innovative framework for secure composition on its formal methods such as: Protocol Composition Logic (PCL) and Protocol Derivation System (PDS). PDS is syntactic support approach in derivations to start from basic components make complex protocols and combines or extends in a sequence of operations over the refinements, transformations and compositions. Floyd-Hoare logic is a foundation of PCL that supports axiomatic proofs for protocol properties [94]. The PCL objective is to form a proof method for every applicable derivation for PDS. Therefore it may also be enabling its security proofs, and may also being applicable in parallel development for others protocols [95]. Any protocol execution contains as assertions associated with the same. The powerful possible observation offers reason to leaving all sprints of the protocol without any logics. The basic operation for ISO-9798-3 based on Diffie-Hellman exponential as (CR) protocol considers, as shown in Figure 5.1, that represents to show the messages how are sent by one and may be received by other. The basis of execution consists of protocol on initiator role and responder role, respectively. The initiator principle role is executing to generate a fresh random number, send the message with its generators to peer; the Responder receive message of its peer (Sender) with source address; verify the same message that contains the signature in anticipated format, and at the end both should be ready to send a subsequent another messages with signature of initiator and responder [96].

![Figure 5.1: ISO-9798-3 Protocol](image)

The backbone of security protocol is the foundational basis that is making certain to forward correctness in many distributed systems to error prone. In relation to presented protocols that contain security redundancies or flaws in the literature of subsequent sections. A simple logic is
described as a consequence of communication and its progression towards the authenticated trustworthy parties involved in authentication protocols. Further, we consequently explained our proposed work formally for a variety of protocol families that ascertain the errors and nuances, and instead of that suggest improvements in them.

We have given a brief idea with intensification of Elliptic Curve Cryptography (ECC) in the next section, which considers multilayer consensus key generations using the same. Afterword’s signcryption based approach is applied which reduces the computation cost as well as communications cost.

5.2 ELLIPTIC CURVE CRYPTOGRAPHY

The cryptography heart is fame for its Discrete Logarithmic Problem (DLP), it acts as pivotal role on fundamental basis in security systems. At a lower cost, high speed computational algorithm and an incorporation of the exciting feature that keep with greater significance and is always a demanding issue. For efficient implementation of cryptographic protocols are playing a central role for the same. Cryptographic algorithms used in the approach which are slow in running approach impinged the customer dissatisfaction and inconvenience. It is very clear now that computation and communication security with faster algorithms run are leading in high performance and high speed.

Elliptic Curve Cryptography (ECC) [12] was proposed in 1985 by Neal Koblitz and Victor Miller. With rapid growth as in the recent state of affairs using the ECC algorithm is playing an important role and assumed to be secure according to NIST guidelines released in 2012. These types of contribution are also possible by except of ECC but they require higher key length. Due to this increased length, the computation costs and/or communication costs involved in the method are not longer suitable for short-memory devices. ECC is one of the techniques that are being used to provide the same level of security on shorter sizes key. But, for the research point of view still huge improvement is possible. The possible cost minimization is available in the literatures in overall consideration [13], is one of our motivation.

The core building block of the public key cryptography for DLP on ECC is presented on two $P$ and $Q$ points on the elliptic curve for the secret key $k$, such that $Q = kP$ [14]. This procedure restrains itself as a repeated point doubling (DBL) and point addition (ADD) operations.
It has been considered as an effective alternative approach relative to RSA algorithm of an already established due to following reasons- low entropy random numbers, lack of forward secrecy, chosen cipher-text attacks, higher complexity, the mathematical attacks etc. It also guarantees services on shorter keys for ECC security. Less arithmetic cost, time saving and less space for key storage are the special benefits, when keys are transmitting the same. These characteristics make ECC is one the right choice in security to incorporate the feature in mobile devices, online banking, smart cards, routers, consumer electronics items, printers, bridges, automotive, network devices, and many more are still be possible. Increased ECC evidence can be evinced by its inclusions in the most credited standards organizations for National Institute of Standards and Technology (NIST), International Standards Organization (ISO), American National Standards Institute (ANSI), and Institute of Electrical and Electronic Engineers (IEEE).

Hardware encryption devices are being used for the cryptographic algorithms that are running on a physical and general purpose security for the most operating systems. These devices are providing high speeds, high performance and at most security services. ECC is mostly best used due to exhaustive where limitation in resources for devices is and prompting for feasibility for high speed demands.

In cryptographic field, ECC has attracted the most attention of researchers in last two-and-half decades and dominating DSA/RSA system approaches. The extension of ECC is providing better performance as it is being used in the cryptosystems because of an improved version of algorithm, the uses of special functionalities and use of specialized curves. The major thought being it works on less memory and has much faster computation. The requirements of memory size execution and code sizes are also smaller. ECC is an appropriate algorithm that works on smaller key length and makes for an efficient practical application. These are based on two fields such as prime and binary fields for ECC. A large amount of published research is offering an interesting standard prime field based alternative in cryptosystems, the reason being that for the same security level, it requires less memory and provide much faster performance. The elliptic curve scalar multiplication is evaluating (on general) by (27) equation-

\[
\begin{align*}
    y^2 = (x^3 + ax + b) \mod p \text{ where } (x, y) \in \mathbb{Z}_p \\
    (\text{for large prime field } ) p > 3 \text{ a, b } \in \mathbb{Z} \text{ and } 4a^3 + 27b^2 \neq 0 \mod p 
\end{align*}
\]  

(27)

The scalar multiplication used in the ECC is the backbone for every algorithms used in ECC primitives. These are principally based on three main approaches. The first one is based on prime
or binary field operations in the underlying finite-field, and these are en-routing itself as alternative replacements for better solutions. The second approach is the computation of scalar multiplication on behalf of the applied algorithms that decides its complexity cost [97].

The National Institute of Standard and Technology (NIST) [98] document specifies with an objective as an authentication and integrity in cryptography. It means the discrete logarithmic problem works as key role in the medium and assumptions in the form of unpredictable for almost all applications.

This chapter is presenting the contents into four (04) sections. A protocol derivational approach is presented, in section 5.3. In section 5.4, Multilayer Consensus ECC-based password authenticated key exchange (PAKE) as an auxiliary model is presented for standard ECC protocol. Section 5.5, a formal verification approach is depicted on AVISPA & SPAN tool. Finally in section 5.6 is presented applicability of signcryption with the enrichment to the applications.

5.3 MOTIVATION TOWARDS DERIVATION OF SECURE PROTOCOL COMPOSITION

Protocol Derivation System (PDS) and Secure Composition of Protocol are addressed the two central specification under the security framework [94]. The aspiration is in developing methods for their complex protocols as security aspects by identifying and/or independent combination of their independent proofs. PDS supports logical approach in derivations which initiate from the basic component and extend or combine a component sequence of compositions, transformation and refinements operation. They consider a list of elementary building block elements, encryption set operations which replaces the same with an encrypted nonce for plaintexts, then transfer the same into specific channel and at the end it should be recovered as unintelligible, respectively. It consist a set of roles involment like to be a server, an initiator and a responder, where each plays a role of actions on desired protocol on a sequence of input, output operations. A common shared secrete key is formed on behalf of the executing roles played by their own private signing keys on generated nonces. A component $C_1$ for Diffie-Hellman is an example that is sharing a key on $g^{lr} \mod q$ and it gives a sign for communication in between two parties where mostly the passive attacker usually difficult to be discovered. Component $C_2$ is considered as
signature-based authenticator at the other end as challenge-response signature on its generated nonce [99]. The standard authentication mechanism is shown below-

$$ I \rightarrow R: m; \ R \rightarrow I: SIG_R(m) $$

(28)

In the cryptography, it is assumed that m is a nonce or fresh value. Public key certificate possessed by responder R to verify it’s signature using the transformation and refinements operations. The refinement shows the message component instances replaced by unidentifiable means, such type replacement gives guaranteed freshness, guarantees from internet key exchanges, protection in identity enclosed against from passive attackers, forward secrecy etc. A basic thought of the existing refinements have presented here. For protection in identity, the first refinement $ R_1, SIG_x(m) = E_k(SIG_x(m)) \}$ works, the second refinement $ R_2, SIG_x(m) = SIG_x(HMAC_x(m,ID_x)) \}$, proves that the signature term is itself generated from x and in addition key hash proves that x possesses key K. For Internet Key Exchange (IKE) is an important property for mutual authentication. The refinement $ R_3, SIG_x(m) = SIG_x(m), HMAC_k(m, ID_x), $ same purpose is served like $ R_2 $ but instead of the same it is used for Just Fast Key protocol as a derivation. Refinement $ R_4, SIG_x(m) = SIG_x(m, ID_x), $ is assumed for x possess the required information Y’s is identified as public key certificate. The refinement $ R_5, g^x = g^n, n_x \}$ where $ n_x $ is a spanking new to serve the two purposes (i) to provide guarantee as fresh value for each in order to prevent in replay attack and (ii) to derive the secret key. Refinement $ R_6, SIG_x(m) = SIG_x(m), ID_x, $ where Id_x refers to public key certificate for x and the verification keys don’t possess them for others. The public key certificate is used in the session key establishment. Further, Refinement $ R_7, SIG_x(m) = E_{k'}(m), HMAC_k(role,E_{k}(m)) \}$ where $ k' \}$ and $ k \}$ identify the protocol shared among initiator and responder. Just Fast Key formulation is used using this refinement. A hashed key includes encrypted signature.

Transformations in the other way are classified in three (03) parts such as- Message Content Move $ T_1, $ Binding $ T_2 $ and Cookie $ T_3. $ $ T_1 $ message is a move from one state to another but any freshly generated data is not contained by the same. Transformation $ T_2 $ adds (in general) binding information from one the protocol to different in some significant form and should be unpredictable state as:

$$ I \rightarrow R: m \quad I \rightarrow R: m $$

$$ R \rightarrow I: n, SIG_R(m) \quad R \rightarrow I: n, SIG_I(m,n) $$

(29)
The Cookie transformation is a freshly generated data stored in small that makes a protocol resistant to DOS (blind) attack. When each time user logs in to browser it adds cookies back to server that can also are considered on website as previous activity information.

\[ R \rightarrow I: n, SIG_R(m) \quad I \rightarrow R: SIG_I(m, n) \]

For protocols derivation of an innovative approach is proposed, as depicted in Figure 5.2, by Datta. We are presenting a broad aspect towards the protocol derivation using compositional logics:

**Protocol P₁** obtains from two symmetric component \( C_2 \) as sequential composition as:

\[
I \rightarrow R: m; R \rightarrow I: SIG_R(m); I \rightarrow R: n; R \rightarrow I: SIG_I(n)
\]

(30)

The \( m \) and \( n \) values assumed as fresh nonce and public key certificates for \( I \) and \( R \) possessing each other’s for verifying the signatures.

**Protocol P₂**: Transformation \( T_1 \) applies on protocol \( P_1 \) to get this protocol:

\[
I \rightarrow R: m; R \rightarrow I: n, SIG_R(m); R \rightarrow I: SIG_I(n)
\]

(31)
This is a way to reduce the messages complexity length from 4 to 3.

**Protocol P₃**: This protocol is achieved from protocol P₂ by T₂ by binding operation:

\[ I \rightarrow R: m; R \rightarrow I: n, \text{SIG}_R(m,n); R \rightarrow I: \text{SIG}_I(m,n) \]  (32)

**Protocol P₄**: This is one of the standard challenge-response protocol for the alternative derivation of ISO-9798-3 protocol, obtained from refinement R₄ is applied over the protocol P₃.

\[ I \rightarrow R: m; R \rightarrow I: n, \text{SIG}_R(n,m,\text{ID}_I); R \rightarrow I: \text{SIG}_I(m,n,\text{ID}_R) \]  (33)

In such regard protocol P₃ is refined that included inside of the peer’s identity signature, from man-in-middle attack, in case of wrong identities verification. Thus it provides the mutual authentication.

**Protocol P₅**: Component C₃ composes with protocol P₃, here C₃ is Diffie-Hellman component for getting this protocol. ECC or RSA component also works according to the needs and is also applicable provided all remaining things are same.

\[ I \rightarrow R: g^i; R \rightarrow I: g^r, \text{SIG}_R(g^r,g^i); R \rightarrow I: \text{SIG}_I(g^i,g^r) \]  (34)

If responder R is honest, initiator I completes its session with R, then a secret key \( g^{ir} \) shares with them. Here man-in-the-middle attack is still possible. But, to overcome this situation four alternative paths for the same are:

**Protocol P₆**: The protocol is obtained from protocol P₅ by applying the refinement R₁. This is also known as secure-transmission-system (STS) protocol, it keeps all the properties of P₅ and in addition it doesn’t provide any identity protection from passive attackers and man-in-middle attacks.

\[ I \rightarrow R: g^i; R \rightarrow I: g^r, E_K \left( \text{SIG}_R \left( g^r, g^i \right) \right); R \rightarrow I: E_K \left( \text{SIG}_I \left( g^i, g^r \right) \right) \]  (35)

However, possible man-in-middle attack is still proved by Lowe here but he was not able to say anything for mutual authentication breaks remain the same or not.

**Protocol P₇**: On protocol P₆ refinement R₅ is applied for this protocol:

\[ I \rightarrow R: g^i, n_i; R \rightarrow I: g^r, n_r, E_K \left( \text{SIG}_R \left( g^r, n_r, g^i, n_i \right) \right); I \rightarrow R : E_K \left( \text{SIG}_I \left( g^i, n_i, g^r, n_r \right) \right) \]  (36)
The exponentials reuse across multiple sessions is computationally more efficient in this protocol is a motivation issue that enables it for the forward secrecy and it doesn’t compromise with its secrecy in the long run.

**Protocol P8:** It is obtained by using refinements $R_6$ to protocol $P_7$, such as

$$I \rightarrow R: g^i, n_i; \; R \rightarrow I: g^r, n_r, E_K(SIG_R(g^i, n_i, g^r, n_r), ID_R); \; I \rightarrow R: E_K(SIG_R(g^r, n_r, g^i, n_i), ID_I)$$ (37)

After the application of this refinement, the protocol assumption is possessed by a public key certificate that discharges it from exchanging certificates alongside the signature identifiers, and by the other means that no new properties are introduced.

**Protocol P9:** This protocol is attained from the cookie transformation $T_3$ on protocol $P_8$:

$$\begin{align*}
I & \rightarrow R: g^i, n_i; R \rightarrow I: g^r, n_r, HMAC_{HK}(g^r, n_r, g^i, n_i), \\
I & \rightarrow R: g^i, n_i, g^r, n_r, HMAC_{HK}(g^r, n_r, g^i, n_i), E_K(SIG_I(g^i, n_i, g^r, n_r), ID_I), \\
R & \rightarrow I: E_K(SIG_R(g^r, n_r, g^i, n_i), ID_R)
\end{align*}$$ (38)

In addition to protocol properties of $P_8$, this ensures that additional property is resistant to the blind Denial-of-Service (DoS) attacks.

At this juncture, the derived protocol provides the DoS protection, mutual authentication, key secrecy, computational efficiency and identity protection from initiator & responder, respectively.

Further, the Just Fast Key (JFK) for initiator (I) and responder (R) is obtained from protocol $P_9$, with the only difference that they offer only identity protection.

**Protocol P10:** The $JFK_R$ is obtained by applying refinement $R_7$ to protocol $P_9$. Instead of that, the protocol has added two more refinements.

$$\begin{align*}
I & \rightarrow R: g^i, n_i; R \rightarrow I: g^i, n_i, HMAC_{HK}(g^r, n_r, g^i, n_i), \\
I & \rightarrow R: g^i, n_i, g^r, n_r, HMAC_{HK}(g^r, n_r, g^i, n_i), E_K(SIG_I(g^i, n_i, g^r, n_r), ID_I), \\
R & \rightarrow I: E_K(SIG_R(g^r, n_r, g^i, n_i), ID_R), HMAC_K(R, E_K(SIG_I(g^i, n_i, g^r, n_r), ID_R))
\end{align*}$$ (39)

During computation the keys $K$ and $K'$ require knowledge of $g^{ir}$ that guarantees it to initiate from the Man-in-the-middle attack and it can’t be computed by the hashed encrypted signature.

**Protocol P11:** The $JFK_I$ obtained by applying transformation $T_1$ to protocol $P_9$. Instead of same, this protocol has added modifications.
\[
\begin{align*}
I \rightarrow R : g^i, n_i; & \quad R \rightarrow I : g^r, n_r, ID_R, HMAC_{HK_r}(g^r, n_r, g^i, n_i) \\
I \rightarrow R : g^i, n_i, g^r, n_r, HMAC_{HK_r}(g^r, n_r, g^i, n_i), & \quad E_k(SIG_I(g^i, n_i, g^r, n_r) ID_I), \\
\intertext{Equation (40)}
E_k(SIG_I(g^i, n_i, g^r, n_r)) HMAC_{HK_r}(g^r, n_r, g^i, n_i), & \quad E_k(SIG_I(g^i, n_i, g^r, n_r) ID_I) \quad I, SIG_I, ID_I; \\
R \rightarrow I : E_k(SIG_R(g^r, n_r, g^i, n_i) ID_I).
\end{align*}
\]

Here, the \( ID_R \) message component is shifted, to reason for applying transformation to include the peer’s identity inside the signature. In this regards, I’s signature possesses the R’s identity before it sends the message in the protocol. This also retains all the properties contained in \( P_9 \) is different except for identity protection. But, the major drawback is the responder’s identity protection.

**Protocol \( P_{12} \):** The protocol \( P_{12} \) is obtained from the protocol \( P_{11} \) by applying the refinement \( R_4 \). This is equivalent to \( JFK_t \) except for one additional signature that is added by using the one more transformation in message and for other end the core security property is ignored.

\[
\begin{align*}
I \rightarrow R : g^i, n_i; & \quad R \rightarrow I : g^r, n_r, ID_R, HMAC_{HK_r}(g^r, n_r, g^i, n_i), \\
I \rightarrow R : g^i, n_i, g^r, n_r, HMAC_{HK_r}(g^r, n_r, g^i, n_i)(SIG_I(g^i, n_i, g^r, n_r) ID_I), \\
\intertext{Equation (41)}
E_k HMAC_{HK_r}(g^i, n_i, g^r, n_r), & \quad E_k(SIG_I(g^i, n_i, g^r, n_r) ID_I) \quad I, SIG_I, ID_I; \\
R \rightarrow I : E_k(SIG_R(g^r, n_r, g^i, n_i) ID_I).
\end{align*}
\]

The peer’s identities refinement adds are of \( ID_I \) and \( ID_R \) inside the signatures, respectively. This prevents the attacks, and retains all the properties of protocol \( P_{11} \).

**Protocol \( P_{13} \):** The Internet Key Exchange (IKE) is one of the protocol that is obtained by applying refinement \( R_2 \) to protocol \( P_5 \). This is described as the core for IKE as

\[
I \rightarrow R : g^i; \quad R \rightarrow I : g^r, SIG_R\left(HMAC_K(g^r, g^i, ID_R)\right); \quad I \rightarrow R : SIG_I\left(HMAC_K(g^i, g^r, ID_I)\right) \quad (42)
\]

Each principal used in the exponentials signs a keyed hash and their own identities. The adversary can’t attack on the used hashed key from the secret \( g^{ir} \) which is only known to \( I \) and \( R \). So, this provides for both the mutual authentication and a shared secret between them.

**Protocol \( P_{14} \):** This protocol derivation is achieved by using the refinement \( R_5 \) when applied to protocol \( P_{13} \). This sensibly parallels the steps for \( JFK_r \) and \( JFK_t \) where exponential nonces are exchanged.

\[
\begin{align*}
I \rightarrow R : g^i, n_i; & \quad R \rightarrow I : g^r, n_r, SIG_R\left(HMAC_{HK_r}(g^r, n_r, g^i, n_i)\right), \\
I \rightarrow SIG_I\left(HMAC_{HK_r}(g^r, n_r, g^i, n_i) ID_I\right).
\end{align*}
\]

\( (43) \)
The purpose is to allow and reuse the exponential in a more efficient protocol for multiple sessions. Although, it contains one of the tradeoff during the processing, in the loss of perfect forward secrecy.

**Protocol P**\textsubscript{15}: It is one of alternative paths for protocol P\textsubscript{5} that consists of the core for JFK\textsubscript{r} and JFK – SIGMA. This protocol is obtained by using the refinement \( R_{3} \) to protocol \( P_{5} \).

\[
I \rightarrow R: g^{i}; R \rightarrow I: g^{r}, SIG_{R}(g^{r}, g^{i}), HMAC_{K}(g^{r}, g^{i}, ID_{R}); \\
I \rightarrow R: SIG_{I}(g^{i}, g^{r}), HMAC_{K}(g^{i}, g^{r}, ID_{I})
\] (44)

This is very similar to protocol like \( P_{13} \), and it also possesses the same properties of shared secret and mutual authentication. One of differences observed is in signing the keyed hash and the principals to send the hash separately. So, for adversary can’t launch the MAN-IN-MIDDLE ATTACK attack because the computation of the hash requires key only known to I and R.

**Protocol P**\textsubscript{16}: This protocol is obtained from protocol \( P_{5} \) by using refinement \( R_{4} \), and it is also known as ISO-9783-3 protocol:

\[
I \rightarrow R: g^{i}; R \rightarrow I: g^{r}, SIG_{R}(g^{r}, g^{i}, ID_{I}); I \rightarrow R: SIG_{I}(g^{i}, g^{r}, ID_{I})
\] (45)

This protocol set-up is a secret mutual authentication scheme and refers to man-in-middle attack which is not possible, because of its intended identity recipient’s signature, attacker doesn’t forward identity either to I or to R.

Now, we are applying the recent protocol of ECC-Based Password Authenticated Key Exchange Protocol on Multilayer Consensus as a key part, the next section elaborates the same and is also one of the motivating issues.

### 5.4 RELATED WORK AND BACKGROUND

ECC-Based Password Authenticated Key Exchange Protocol on Multilayer Consensus is a key of our work and its related idea has been presented in [100]. Password authenticated key exchange protocol (PAKE) is an elementary protocol derivation based on two-steps and it is also known as Simple Authenticated Key Exchange Protocol (SAKA), presented in [101], shown in Figure 5.3. Further, X. Ding et al. presented PAKE on three step to resist password compromise impersonation, compromise on ephemeral key, forward secrecy, and dictionary attack. The IEEE standard 1063.2 was released in 2009; this specifies secrets on shorter key as a strong security transactions and to show a proficiently utilizing password [102]. The basic idea is presented here,
A group generator $G$ is available, where each party randomly selects its secret keys (as a number) and multiplies the same with $G$, which shares using the ECC as depicted in X.1035 standard that is resistant against to guess the password attack.

**Figure 5.3:** Working of ECC-based PAKE (EPAK) protocol in between Alice & Bob

**Step I:** Assume Alice as an initiator chooses a secret random key $d_A$, multiplies with group generator $G$ that is public key $P_A$ and represented the same in Elliptic point $(x_a, y_a)$. It further computed hash $H_{(p)}$ with it encrypts $P_A$ as $X$ and is sent it to Bob (20):

$$P_A = d_A \cdot G; \ (x_a, y_a) = P_A; \ X = E_{H_{(p)}}(P)$$  \hspace{1cm} (46)

A packet received $X$, Bob decrypted the same and represented in elliptic point $(x_a, y_a)$ (47):

$$P_A = D_{H_{(p)}}(X); \ (x_a, y_a) = P_A$$  \hspace{1cm} (47)

**Step II:** At other end Bob picks up a secret random key $d_B$ as private key and obtained public key $P_B$ it is multiplied it to group generator $G$, also appropriates it to Elliptic point in (48):

$$P_B = d_B \cdot G; \ (x_a, y_a) = P_B$$  \hspace{1cm} (48)

Again, it is multiplied private key with Alice public key to obtained a shared key $Q_{AB}$ and finds its appropriate EC points $(x_{ab}, y_{ab}) = P_{AB}$ then computes $S_B$ having $Q_A, Q_B$ and $Q_{AB}$ and finally uses $H_{(p)}$ to encrypt:

$$Q_{AB} = d_B \cdot P_A = d_B \cdot d_A \cdot G; \ (x_{ab}, y_{ab}) = P_{AB}; \ S_B = H(P \mid y_a | y_b | y_{ab}); \ Y = E_{H_{(p)}}(P_B)$$  \hspace{1cm} (49)
To decrypt $Y$, Alice used $H(p)$ and obtained $Q_B$ and also the converted elliptic point $(x_a, y_a)$ aligned to $Q_B$. Again, the Alice private key multiplies with public key sent by Bob $Q_B$ and shares a common shared key $Q_{AB}$ followed by points $(x_{ab}, y_{ab})$. Finally computed $S_A$ having for verification of $Q_A$, $Q_B$ and $Q_{AB}$. Alice is now assured the verification on received the requited values as (50):

$$P_B = D_{H(p)}(Y) \quad Q_{AB} = d_A, P_B = d_A, d_B, G; S_A = H(P|y_a|y_b|y_{ab})$$ (50)

**Step III:** Bob needed to assure Alice that she has required values as well. So, she needs to performs $T_A$ out of $Q_A$, $Q_B$ and $Q_{AB}$ and send it to Bob as (51):

$$T_A = (H(P|x_a|x_b|x_{ab})$$ (51)

On the other side Bob calculates $T_B$ and compares it with $T_A$. If the verification holds good Bob also assures Alice that she also has the required values as well (52):

$$T_B = H(P|x_a|x_b|x_{ab})$$ (52)

**Step IV:** Therefore, on the generated parameters both parties are verified with each other and calculated the secret shared key as (53):

$$K_{AB} = H(P|x_a|x_b|x_{ab}|y_a|y_b|y_{ab})$$ (53)

Further, here Multilayer Consensus Password Authenticated Key Protocol Exchange (ECPAKE) protocol for key exchange is considered. A key agreement for mutual authentication among (an initiator) appliance network $A_N$, Home Area Network $H_c$, Building Area Network $B_c$, Neighbor Area Network $N_c$ and Central Controller $C_c$ is considered, where all controllers are resulted in individual operations on them and are reported correctly working for all. In this we considered the same approach which is taken the ECC advantage for key generation, as shown in Figure 5.4. If required, it can be extended to a larger layer of security; it is also adopted by the standard X.1035.

For PAKE using ECC approach, first iteration in between $A_N$ and $H_c$. Further, the same philosophy is applied for second, third and fourth consequent layers. These have also been available for ECC based password authenticated key exchange protocol for multilayer consensus. Since the MCEPAK proposed protocol [100] is established on ECC, and X.1035 that contains similar benefits like the Diffie-Hellman procedure. In the proposed work, the security and different attacks have been analyzed and modeled on the same, where an adversary (internal or external) is capable of re-scheduling, re-playing, re-ordering, re-routing, deleting and recording
of the messages is considered. We have done the formal verification of the proposed protocol underneath under the same adversary conditions.

![Multilayer Consensus Key-Generation Approach](image)

**Figure 5.4:** Multilayer Consensus Key-Generation Approach

### 5.5 FORMAL VALIDATION USING SPAN AND AVISPA TOOL

AVISPA [103] is one of the automatic verification and validation tool that used in the cryptography. It is widely used for Internet security applications and its protocols verification. It offers a significant expressive formal language for specifying protocols with their safety measures that is modularized into different four back-ends under the perimeter, the structure is shown in figure 5.5. Its accomplishment is based on the automatic analysis techniques. The High Level Protocol Specification Language (HLPSL) is described to formally validate the security protocols and it specifies the intended security properties. The HLPSL specification is first translated into Intermediate Format (IF) through translator HLPSL2IF. The IF is a lower-level language that is used for directly interpretation for back-ends tool. The IF objective has formulated for developers with the implication to use it for their input language analysis. This happens automatically and is transparent to the user [104].
Now, the IF specification is analyzed at the back-ends for the satisfied or violated security goals. The AVISPA Tool comprises of four back-ends such as: On-the-fly Model Checker (OFMC) [105], Constraint Logic-based Attack Searcher (CL-AtSe) [106], SAT-based Model Checker (SATMC) [107]-[108], and Tree-Automata Based Protocol Analyzer (TA4SP) [109]. The definition of OFMC says it is a useful debugging tool for protocol specification that allows agents to execute all the required steps for honest run of the protocol and is specific for manual check if needed. CL-ATSE is set of constraints, used to find attacks on protocols where translation and checking are fully automatic that are internally performed by the same i.e. no external tool is used. Its back-end is slightly different format of trace in some aspects of attack other than what OFMC does, it writes an interpretation in the intermediate format (IF) as tests. SATMC’s is used to check the executability that includes functionality to confirm the on HLPSL specification. SATMC is strict in particularly for the proper specification; this feature is useful in finding errors. The TA4SP proves secrecy properties with an unbounded number of sessions. From the practical point of view, this works completely automatic and is supported by two (2) tools such as Timbuk and its extensional part. The analysis of four back-ends are harmonized with each other in a sense for some common back-ends procedure, but these are not equivalent.
that should return with different results. The proposed MCEPAK protocol running on the tool, as shown in Table 5.1, at back ends of OFMC and CL-AtSe, with safety measures.

An impressive SPAN tool comes with simple editing protocol specifications of web graphical interfaces of AVISPA, and in addition to this it contains honest agents for protocol simulation, intruder simulation for honest agents and an attack simulation. Attack simulation in this is like the same layout in intruder simulation, but attacks are automatically built by using OFMC/CL-AtSe facilities.

### Table 5.1: OFMC and CL-AtSe Back end results on AVISPA

<table>
<thead>
<tr>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
</table>
| A@ubuntu:~/avispa-1.1$avispa BasicMainHlPsl.hlpsl --ofmc | SUMMARY SAFE  
DETAILS BOUNDED_NUMBER_OF_SESSIONS  
PROTOCOL /home/A/avispa-1.1/testsuite/results/BasicMainHlPsl.if  
GOAL as_specified  
BACKEND OFMC  
COMMENTS STATISTICS parseTime: 0.00s  
searchTime: 0.05s  
visitedNodes: 6 nodes  
depth: 2 plies |
| A@ubuntu:~/avispa-1.1$avispa BasicMainHlPsl.hlpsl --cl-atse | SUMMARY SAFE  
DETAILS BOUNDED_NUMBER_OF_SESSIONS  
TYPED_MODEL  
PROTOCOL /home/A/avispa-1.1/testsuite/results/BasicMainHlPsl.if  
GOAL As Specified  
BACKEND CL-AtSe  
STATISTICS Analysed : 1 states  
Reachable : 0 states  
Translation: 0.02 seconds  
Computation: 0.00 seconds |

The security protocol analysis is the major idea possible through the two specifications such as High Level Protocol Specification Language (HLPSL) and CAS+. HLPSL is a language used for specifying the cryptographic protocols for AVISPA toolset and CAS+ is a light evolution of CASRUL language. The Figure 5.6, depicts the operation on ECC through CAS language and shows the principles of sender pattern as the tool dictates it like the same.
This specification translates a CAS+ language from Alice-to-Bob for simple and fast specification of security protocols; interactively building a Message Sequence Chart (MSC) \cite{100}-\cite{111} of protocol execution; MSC build attacks automatically on either of HLPSL and CAS+ specifications; and for intruder interactively builds a specific possible attacks.

\textbf{Figure 5.7:} Intruder Simulation on ECC in between sender and receiver

Figure 5.7 is shown simulation approach on sender-receiver with an inclusion to intruder pattern generation and is observed as real messages transmission.
In figure 5.8, we have shown the intruder formal verification for multilayer consensus. The MCEPAKE proposed scheme is an enhancement for multilayer security among the layers of networks. Its percent improvement encryption and decryption time between layers are presented below in Table 5.2.

**Table 5.2:** Execution Time: Encryption and Decryption

<table>
<thead>
<tr>
<th>$A_N \leftrightarrow H_C$</th>
<th>$A_N \leftrightarrow B_C$</th>
<th>$A_N \leftrightarrow N_C$</th>
<th>$A_N \leftrightarrow C_C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2t_0$</td>
<td>$4t_0$</td>
<td>$6t_0$</td>
<td>$8t_0$</td>
</tr>
<tr>
<td>$2t_0$</td>
<td>$2t_0$</td>
<td>$2t_0$</td>
<td>$2t_0$</td>
</tr>
<tr>
<td>0%</td>
<td>50%</td>
<td>66.69%</td>
<td>75%</td>
</tr>
</tbody>
</table>

Using ECC algorithm for encryption and decryption of desired message $m$ with the shared secret key $K$ is used in MCEPAKE protocol to form the ciphertext $C_m = \{KG, m + K.P_B\}$, where $P_B$ is the responder public key used by the initiator. Now, to decrypt ciphertext $C_m$ to find plain texts message as $m = \{m + K.P_B - n_B(K.G)\}$, where $n_B$ is a responder private key [112], is used during the whole procedure.

In the next section, our proposed Signcryption technique is presented in relation to the basics of this technique. This is one of the efficient and modernized techniques that serves two purposes such as digital signature and encryption of transmitted message with reduced computation and communication costs.
5.6 Signcryption

In 1997 Zheng [113] first proposed the Signcryption primitive of cryptography. It logically combines digital signature and encryption scheme in a single step in less computational and communication cost. Using the Signcryption, he proposed 58% of less computational cost and 70% less communication cost only when in general it is compared with the individually signature–then-encryption schemes. The parameters used in the schemes contain the respective sizes that decide its cost such as |p| = 512 bits, |q| = 144 bits and |hash(.)| = |KH(.)| ≡ |q|/2. Here we have presented in a Table 5.3 that shows the above enrichments in relation to basic signature-then-encryption on Schnoor Signature plus ElGamal Encryption versus Zheng signcryption. Except the exponential (EXP) function, the other functions used in the explanation are equalized with each other so its cost is considered to be negligible. The computational cost represents a reduction in \{((5.17 - 2.17)/(5.17)) = 58\%. Whereas this table concludes \{(\mid\text{hash(.)}\mid + |p| + |q|) - (\mid\text{KH(.)}\mid + |p|)/(|\text{hash(.)}| + |p| + |q|)\} = 70\%\) saving in communicational cost.

<table>
<thead>
<tr>
<th>Schemes</th>
<th>Computational Cost</th>
<th>Communicational Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signature-then-encryption based on “Schnoor Signature + ElGamal Encryption”</td>
<td>EXP=3, MUL=1, DIV=1, HASH=1, ENC=1</td>
<td>\mid\text{hash(.)}\mid +</td>
</tr>
<tr>
<td></td>
<td>{ EXP=2.17, MUL=0, DIV=1, HASH=2, ENC=1 } Total Modular Reduction=5.17</td>
<td></td>
</tr>
<tr>
<td>Signcryption</td>
<td>EXP=1, MUL=1, DIV=1, HASH=1, ENC=1</td>
<td>\mid\text{KH(.)}\mid +</td>
</tr>
<tr>
<td></td>
<td>{ EXP=1.17, MUL=0, DIV=1, HASH=2, ENC=1 } Total Modular Reduction=2.17</td>
<td></td>
</tr>
</tbody>
</table>

This is a huge saving for applications in computation and communication cost in the form of secure & authenticated message delivery. There are other applications also that are brought to notice such as authenticated electronic secure transactions, non-repudiated key transportation, video conferencing inclusive of through secure and authenticated multicast services, unforgeable messages fast & compact services.
For the last so many years, many variations of this scheme have been proposed which are having their own problems and limitations, that are offering optimized computational costs and different levels of security. Baek, in 2002, gave the formal proofs of Signcryption in [114]. The real life application of Signcryption is based on old adage “killing two birds with one stone”. Confidentiality is achieved through encryption, whereas authentication is provided by the integrity of this scheme. Authentications on private key and digital signatures on public key are authentication scheme and are playing an important role.

In general, the objective of Signcryption states that the cost of Signature and Encryption achieved through approach is always be less than the individual cost of Signature and individual cost of Encryption [113]. Further, these are interpreted in a number of ways:

- A combination of digital signatures and encryption scheme, signcryption should be more efficient (computationally).
- A naive combination of digital signatures and ciphertext encryption, signcryption should produce shorter cipher text.
- A naive combination of digital signatures and public-key encryption, signcryption should be endowed with better safety measures and/or bigger functionality when compared.

The signcryption scheme works in five phases such as: Setup phase, Sender Key Generation phase, Responder Key Generation, Signcrypt phase, and in Unsigncrypt phase.

**Phase 1: Setup Phase**

The setup factor is based on the security and common key generation parameters. The overall parameters factors are made public for all as the summary contains like: $p$ is a large prime; $q$ is a prime factor of $p - 1$; $g$ is an integer with order $q$ modulo $p$ in $[1, ..., p - 1]$; $KH$ is a key hashed one way function of hash($k, m$); and $(E, D)$ are used for encryption $E$ and decryption $D$ respectively.

**Phase 2: Sender Key Generated phase**

Alice has the pair of keys ($X_a$, $Y_a$), where $X_a$ Alice’s private key randomly chosen from $[1, ..., q - 1]$; $Y_a$ is a generated public key for Alice to modulo prime $p$; Alice is now ready to send a message to Bob.

**Phase 3: Responder Key Generation phase**
Bob keeps a pair of keys \((X_b,Y_b)\), as private key \(X_b\) randomly selected from \([1, \ldots, q-1]\) and his public key \(Y_b\) is generated on the prime modulo \(p\). Bob is now ready to send a message to Alice

**Phase 4: Signcrypt phase**

The initiator and responder accomplish the following operations to message signcrypt:

The key \(k\) splits in \(k_1\) and \(k_2\) of equal length parts; Calculate \(s = hash(k_2,m)\); Calculate \(c = E_{k_1}(m)\) is message \(m\) encryption with the key \(k_1\); then Alice send it to Bob as \((r, s, c)\)

**Phase 5: Unsigncrypt phase**

Finally, to unsigncrypt the signcrypted message, responder accomplishes the following operations:

Calculates \(k\) using \(r, s, g, p, Y_a\) and \(X_b\); \(k = hash(Y_a \ast g^r)^{s \ast X_b} \mod p\); now again Split \(k\) in \(k_1\) and \(k_2\) for the verification of original message in the form of appropriate lengths; the message \(m\) evaluates it by performing decryption \(m = Dk_1(c)\); A valid message \(m\) accepted only if \(KHk_2(m) = r\) is satisfied.

### Table 5.4: Comparison of different algorithm schemes based operations

<table>
<thead>
<tr>
<th>Schemes</th>
<th>Participant</th>
<th>ECPM</th>
<th>ECPA</th>
<th>DIV</th>
<th>MUL</th>
<th>ADD</th>
<th>HASH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zheng</td>
<td>Sender</td>
<td>1</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Receiver</td>
<td>2</td>
<td>1</td>
<td>-</td>
<td>2</td>
<td>-</td>
<td>2</td>
</tr>
<tr>
<td>Hwang</td>
<td>Sender</td>
<td>2</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Receiver</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>Zhou</td>
<td>Sender</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Receiver</td>
<td>4</td>
<td>4</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Basu</td>
<td>Sender</td>
<td>2</td>
<td>-</td>
<td>-</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Receiver</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Proposed</td>
<td>Sender</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>-</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Scheme</td>
<td>Receiver</td>
<td>1</td>
<td>-</td>
<td>2</td>
<td>2</td>
<td>-</td>
<td>1</td>
</tr>
</tbody>
</table>

This approach contains the many features as: - it requires much smaller overhead than the conventional sign-then-encrypt schemes, security against unforgeability, unsigncryptability to verify message. The Table 5.4 is our proposed scheme which shows the improvement over Basu et al. [115] and its related proposed schemes on point multiplication on elliptic curve (ECPM), addition ADD, and Multiplication MUL.
The correctness definition of the scheme is secure, if it satisfies the following conditions:

(i) Unforgeabilty: For an adaptive attacker, it is computationally infeasible for the dishonest Bob and then allows querying for Alice signcryption to masquerade in creating authentic text messages.

(ii) Non-repudiation: For a third party, it is computationally feasible to settle the dispute between the two events.

(iii) Confidentiality: For an attacker it is infeasible to gain an access from signcrypted text. The other party involved may be anyone other than Alice/Bob.

Further, the scheme is generalized into the forms of requirements specifications. It is not only necessary for all messages to require integrity and confidentially, whereas some messages require sign only, while others need to be encrypted. Later on the two cases may provide one of the specific parties to them, despite the fact that conventional signcryption requires both of them. As a result the applications must implement the three individual primitives that include signature, encryption, and signcryption. This scheme has been generalized so that it provides the dual functions with more practicability and flexibility, when simultaneously it requires authenticity and confidentiality. Also, it is endowed with solitary signature or encryption function when authenticity/confidentiality is required without any additional computation and amendments [116].

In the recently scenario, there are many applications that are in light due to their various abilities such as— decreased computation cost, reduced bandwidth, easy applicability to tiny digital phone, handshake on transport layer security, and the connect internet ability. Unforgeable key establishment is the second major application over ATM networks.

5.7 SUMMARY

This chapter contains a secure composition approach that adds and/or makes a way for secure computing techniques. These approaches are widely contributing significantly to the cryptographic applications. Instead of the same our focus is on relative advantages over the signcrypted multilayer consensus based approaches for secure composition. It has been showing in information security, the proposed approach makes scientifically strong security mechanisms in applied cryptography. Our proposed approach has considered the protocol derivational system
and protocol compositional logic approach. The abstract idea presented here is to derive the use of basic components in the formation of Diffie-Hellman, and applicability for secure composition that can be also applied for ECC with its reduced relative cost. In addition to addressing, security concerns without any compromise. Thereafter, by using signcryption primitive that is applied on multilayer consensus ECC based, password-authenticated key exchange protocol approach that significantly reduced both computational and communicational cost. Whereas, new paradigm of signcryption is applied for cost effectiveness, high performance and is favorable for short-memory devices applications and there are many more are the possible advantages of the proposed approaches. Moreover the protocol is formally validated on AVISPA and SPAN tools.