Key Exchange Protocol Using Encryption Scheme Provably Secure Against CCA

Jalaj Kumar Upadhyay

Abstract*—In past, considerable work has been done in the field of password authenticated key exchange since Bellovin and Merritt [3]. In this paper, we present a new idea using an assymmetric encryption scheme that do not require the knowledge of public key of the other party. We encrypt the password along with the session key to be exchanged. The encryption scheme provides the security and the encrypted password provides the authentication. The first protocol we present is trivally secure against the** *key compromise impersonation attack* **and is also secure against the** *dictionary attack, impersonation attack and known session key attack* **under** *random oracle model. We also give a slight modification to the protocol to make it secure against the** *exhaustive offline dictionary attack***.**

***Index Terms*—Key Exchange, Commutative Encryption.**

I. Introduction

PROTOCOL that allows two or more party to agree on a common shared key is commonly known as key exchange protocol. The key exchange protocols can be categorized in to two categories: unauthenticated and authenticated. In an unauthenticated key exchange, the parties involved in key exchange do not authenticate each other. As a result, the adversary can launch a *man-in-middle attack* to agree upon a different key with each of the party involved in key exchange. This situation is resolved by using authenticated key exchange. The protocol is authenticated if every party verifies the authenticity of all the other parties involved in the key exchange. The authentication can be carried out in different ways such as digital signature, password based authentcation or basic challenge-response method. When the means of authentication is a password that is shared by all parties, then it is called password authenticated key exchange (PAKE). Informally speaking, PAKE is a protocol in which two or more parties share a low entropy password and communicate over an insecure channel, authenticate each other and agree on a large session key that is cryptographically secure.

Past Work. Diffie and Hellmann [6] proposed the first key exchange protocol. Their protocol was based on DDH assumption, but lacked authentication. Later on, several works have been done to include authentication. Bellovin and Meritt [3] were the first to suggest use of password to provide authentication and thereby introduced the idea of password authenticated key exchange in two party setting. Their work is another instance of the Diffie Hellmann key exchange. The basic idea is to encrypt the key with symmetric encryption scheme. An attacker can decrypt the encryption by guessing the password, but she cannot be certain whether the decryption results in correct message or not. Jablon [7] proposed a protocol in which he used the password to compute the generator for the Diffie Hellmann key exchange. Bellare et al [1] presented a formal model that is provably secure in the ideal cipher model. The security model was defined for the flow of information in EKE protocol. Boyko *et al* [5] and Mackenzie [11] defined a similar construction called the PAK suite. The security analysis for the forward secrecy is done under the CDH assumption. However, they do not provide enough guarantee in favour of its use in practice. The protocol by Yung *et al* [8] also provide the forward security but is quite inefficient. MacKenzie and Swaminathan [12] proposed a protocol called SNAPI, that is provably secure against the active adversaries. They based their security on the RSA in the *random oracle model*.

Security Aspects. The attack on any key exchange protocol can be viewed in two ways depending on the type of adversary: passive adversary and active adversary. In the former case, the adversary can just eavesdrop the communication and try to gain some information about the password or the key. The adversary only has the information exchange during the course of protocol as the input. In the later case, the adversary can impersonate, inject, delete or replay the communication in order to get some information about either the password or the key. However, if we assume that the password shared by the two parties is of low entropy, then there is another type of attack, which is called *offline dictionary attack*. In this attack, the adversary can act passively or actively to gain some information. From the information gained, the adversary can try a brute force attack on all possible candidates for the password. If the information gained from the active (or passive) interaction with the communicating party is enough to precisely determine the password, then we say that the adversary has a successful attack. Lomas *et al* [9] presented protocols which are resistant to the*offline dictionary attacks*. The protocols assumes that the client has the server's public key. The protocols are therefore not strictly password only protocols. Later Boyarsky [4] also presented a protocol for protection against *offline dictionary attacks*. Recently there have been new security threats known as *key compromise impersonation*, in which compromise of private key of one user implies the compromise of other user's private key. Another widely applicable attack is *known session key attack*, in which the adversary gains information about some session key of the past and using this information, she tries to gain the password or the present session key.

Our Contribution. Most of the protocols that have been

J. Upadhyay is student in the Department of Computer Science and Engineering, Institute of Technology, Banaras Hindu University, Varanasi-221005, INDIA e-mail: (jalaj.upadhyay@gmail.com).

proposed are based on *key encryption*. Key encryption can be done using the symmetric or assymmetric encryption logic. If *symmetric encryption logic* is used, then the password is either used as a key or is used to generate the key for the encryption. In case of public key cryptography (PKC), an extra information (the public key of the the other party) is also required. As a result, an extra step of authentication is required before the key exchange actually starts.

In this paper, we do away with this method. We use a family of encryption scheme that is provably secure against both *chosen plain text* and *chosen cipher text attack* (which will be defined later) and can be easily modelled to act as a *commutative encryption scheme*, which is the actual requirement for the working of the protocol. We also provide an illustration using an encrption scheme that is efficient enough to be used in application. We do not require that the communicating parties have the information about the public key of the other parties, as opposed to the case of PKC. Both the password and the key is encrypted using this encryption scheme. The basic idea is that the encryption protects the password and the password provides the authentication. Since, the password is encrypted by an encryption scheme which is secure, the authentication is well guaranteed. The original protocol is modelled to resist all type of attack and the most an adversary can perform is *brute force dictionary attack*. However, we provide a variation of protocol that resist even the *dictionary attack*. The switching between the both the variant can be done by altering a single step.

In next section, we will give the notations used in this paper and define the security model. In the next section, we will give a brief overview of the protocol. In the fourth section, we define the encryption scheme which will be used in the protocol and present the protocol showing how we model the encryption steps to make it look like commutative encryption scheme. Finally, we give the security proof.

II. Notation and Security Model

We briefly go through the notation used through out the paper.

- The length of the string x is denoted by $|x|$, and the concatenation of the two strings x and y is denoted by $x\|y$. The *bitwise exclusive or* of two string x and y is denoted by $x \oplus y$. The i^{th} bit of x is denoted by x_i and a substring of x from x_i to x_j is written as $x_{i..j}$.
- The plain text message is denoted by m and $\mathcal{H}$ denotes the collision resistent hash function. We will write $\mathbf{R}_l[X]$ ($\mathbf{L}_l[X]$ respectively) as the rightmost (leftmost respectively) l-$bits$.
- G will be a cryptographically secure pseudo random generator and g will denote the generator of a multiplicative group $GF(p)^*$, where p is a prime number.

A. Security Model

The capabilities of adversary are modelled as oracle queries. One aim of the protocol is that no adversary should be able to get any information about the password or the session key generated by the password. Equally important is the mutual authentication of the communicating party. We assume that the adversary has access to three oracles. The queries that he can make are listed below:

1) *Hash*(m): This query models the adversary $\mathcal{A}$ hashing a message m. The oracle $\mathcal{O}_H$ maintains a list of the hash value and the corresponding messages. On any new query, it first check if the similar request has been already made. If any entry matches the list file, the oracle simply returns the hash corresponding to the entry, otherwise it outputs a random number from the output domain, and stores the value in its list.
2) *Pseudo Number Generator*(s): This query models the adversary generating a pseudo-random number from the seed s. The oracle $\mathcal{O}_G$ maintains the list of the seeds and the corresponding random number generated. On new query, it first check its data base. If there is any entry matching the present query, the oracle output the corresponding pseudo random number for that seed, else it flips an unbiased coin the required number of time, outputs the result and prepares an entry for this new query.
3) *Decrypt*(C): This models the decryption of the cipher text C. If the oracle $\mathcal{O}_D$ has already received the cipher text, it returns the decryption, else the oracle first generates a secret key and decrypts the cipher text. It also stores the ciphertext and the decrypted message in its list.

To prove the security of the protocol we have to prove the security of the encryption scheme. The most common attack on the encryption scheme are *known plain text attack*, *cipher text only attack*, *chosen plain text* and *chosen cipher text attack*. The first two are weaker attacks while the later two are stronger attacks. Hence, we only define the notion of polynomial security against the *chosen plain text* and *chosen cipher text attack*. We adapt the notion of polynomial security in the random oracle model [10]. We use the following notation in the definition:

- R denotes the random oracle and 2^∞ denotes the set of all oracles.
- 1^k denotes the output of the *triple algorithms* f, f^{-1}, d, where f is an *one-way function*, f^{-1} is its inverse and d is the uniform distribution over 1^k.

Definition 1: A Chosen Plain text adversary $\mathcal{A}$ (CP-adversary) is a pair of non-uniform polynomial time algorithms (F, A_1), each with access to an oracle. F comes up with two message m_0 and m_1, such that if A_1 is given the encryption α (of any of them randomy choosen), A_1 will not be able to distinguish which one is encrypted. Formally, an encryption scheme $\mathcal{G}$ to be secure in the random oracle model we require that for any CP-adversary $\mathcal{A}$ = (F, A_1),

$$\mathbf{Pr}[R \leftarrow 2^\infty; (\mathcal{E}, \mathcal{D}) \leftarrow \mathcal{G}(1^k); (m_0; m_1) \leftarrow F^R(\mathcal{E}); b \leftarrow 1; \alpha \leftarrow \mathcal{E}^R(m_b) : A_1(E, m_0, m_1, \alpha) = b] \leq \frac{1}{2} + (poly(n))^{-1} \quad (1)$$

Definition 2: A chosen ciphertext adversary $\mathcal{A}$ (CC-adversary) is a pair of non-uniform polynomial time algorithms (F, A_1), each with access to an oracle and a black

box implementation of decryption oracle D^R. The job of f is to come up with a pair of (equal length) messages m_0 and m_1 such that if A_1 is given the encryption α of a random one of these, A_1 will not be able to guess well which one as long as A_1 is not allowed to ask α of the decryption oracle. Formally, A_1 is forbidden from asking an oracle query equal to its final argument. Formally, encryption scheme $\mathcal{G} = (\mathcal{E}, \mathcal{D})$ is secure against CC-attack if for each $\mathcal{A} = (F, A_1)$,

$$\begin{aligned}\mathbf{Pr}[R \leftarrow 2^{\infty}; (\mathcal{E}, \mathcal{D}) \leftarrow \mathcal{G}(1^k); (m_0; m_1) \leftarrow F^{R,D^R}(\mathcal{E});\\ b \leftarrow 1; \alpha \leftarrow \mathcal{E}^{R,D^R}(m_b) : A_1(E, m_0, m_1, \alpha) = b]\\ \leq \frac{1}{2} + (poly(n))^{-1} \quad (2)\end{aligned}$$

III. Overview of the Protocol

This protocol is the cryptographic analogy of the game in which Alice has to send some contents (in a box) to Bob, without anyone else finding out the contents. The game starts by Alice putting the contents in a box, locking the box with her lock and sending the box to Bob. Bob locks the box again with his own lock and sends it back to Alice. Alice removes her lock and sends to Bob. Bob removes his lock to find the content. To apply this for password authenticated key exchange, we use encryption scheme as the lock, and to authenticate we use password. But the encryption scheme that we use should carry the commutative property of the lock analogy.

Now, the direct urge is to use any Commutative encryption scheme. If E and D are the encryption and the decryption algorithm and K_1 and $K_2 \in \mathcal{K}$, where $\mathcal{K}$ denotes the set of all possible key, then a *commutative encryption scheme* is an encryption scheme having the following property: $\mathrm{E}_{K_1}[E_{K_2}] = E_{K_2}[E_{K_1}]$ Hence, $D_{K_1}[D_{K_2}[E_{K_1}[E_{K_2}[M]]]] = M$.

But, we cannot prove that the commutative encryption scheme is IND-CCA secure [13]. Now as the security of the key depends on the encryption scheme by which it is encrypted, so IND-CCA is the basic requirement. Hence we do not use any standard commutative encryption scheme. The encryption scheme that we use should be secure as per equation 1 and 2. Therefore, we propose a variation of encryption scheme given by Bellare and Rogaway [2] and use an encryption Scheme given by Seberry and Zheng [14] as the practical application. Both the scheme are provably secure against the CCA, and can be easily modelled to act as a commutative encryption scheme. The password provides the authentication and it is secured from the attack by the adversary, by preimage resistant property of hash function (in step 2 of the protocol defined below) and the encryption scheme (in all the other steps). We assume that both the parties (we recognize them to be Alice and Bob, and the adversary as Eve) involved in key exchange can construct a public key/private key pair over a pre-defined group. We emphasize that the group should be same, and it is in the public knowledge. Apart from the knowledge of group, both Alice and Bob share a common low entropy password, which only they know.

IV. Encryption Scheme and the Protocol

We use an Encryption scheme which is a slight variation of [2][1]. We also assume that we have *one-way function* (h_1 and h_2) and a cryptographically secure *pseudo-random number generator*, G. For a fixed length plaintext *m*, we chose a random number *r*. The encrypted text is:

$$\mathrm{E} = \mathcal{E}(r, m) = f(r) \| h_1(r) \oplus h_2(m) \| G(r) \oplus m$$

The decryption is done as:

$$\mathrm{D} = \mathcal{D}(E, r) = E \oplus G(r)$$

A. The Protocol

The message that Alice sends at the start is $m = K \| \mathcal{H}(K, P)$, where P is the low entropy password that only Alice and Bob share, and $K \in \mathbf{K}$ is the key. The password is taken from a small dictionary Π.

1) **Encryption by Alice:** Alice chooses a random number $r_A \in \{1, \ldots, p\}$, calculates $\mathcal{E}_A(r_A, m)$ and sends the calculated value.
2) **Bob's encryption:** Bob chooses a random number $r_B \in \{1, \ldots, p\}$. Taking some definite bits of the password as $seed$, Bob finds a pseudo-random number r, such that $|r| = |m|$. He calculates $m_1 = \mathbf{R}_{|m|}[E_A] \oplus r$ and $E_B = \mathcal{E}_B(r_B, m_1)$. He sends E_B and $H_{Bob} = \mathcal{H}(P, E_B)$ to Alice.
3) **Alice's decryption and Bob's authentication:** Alice authenticates the hash value H_{Bob} by calculating the hash of $P \| E_B$. Now Alice decrypts E_B by calculating $\mathcal{D}(E_B, r_A)$

$$\begin{aligned} w &= G(r_A) \oplus E_B \\ &= f(r_B) \| \big(h_1(r_B) \oplus h_2(m_1)\big) \| \big(G(r_B) \oplus m \oplus r\big)\end{aligned}$$

Alice parse it as $a \| b \| c$. He further computes r as Bob does and then calculate :
$D_b = b \oplus h_2(m_1) \oplus h_2(m)$
$D_c = c \oplus r$
Alice sends $D_A = a \| D_b \| D_c$ to Bob.
4) **Bob's decryption and Alice's authentication:** Bob decrypt by calculating $\mathcal{D}(D_A, r_B)$

$$\begin{aligned} D_B &= D_A \oplus G(r_B) \\ m &= \mathbf{R}_{|m|}[D_B]\end{aligned}$$

Bob checks the authentication of the message by hashing m and checking with $D_b \oplus h_1(r_B)$. Bob authenticates Alice by checking the hash of password concatanated with key with the value he recieved.

The protocol is insecure against the *offline dictionary attack*, as we will note later. The insecure step is the second step, where Bob sends E_B and $\mathcal{H}(P, E_B)$ to Alice. An adversary can use this information to launch an *offline dictionary attack*. Once she gets a match of the hash, she can use this password to check the rest of the information exchange. In this way

[1]The security of the scheme can be proved by noting that $h(r\|m) = h_1(r) \oplus h_2(m)$. If one can invert $h(r\|m)$, then one can invert both h_1 and h_2

she can learn the password. We can stop the adversary from launching this attack by using r instead of P during hashing in the second step. The adversary can no longer use *exhaustive dictionary attack*, but has to be dependent on *hash collision*. In the security analysis we will consider the case when the offline dictionary attack is infeasible. However, we will also show if offline dictionary attack is feasible, how we can use this variant of the protocol to resist the offline dictionary attack. Moreover, the protocol provides a way for Alice to get information about $\mathbf{G}(r_B)$. However, in the practical application of the protocol, as in the case shown below, there is no way for Alice to gain even this knowledge.

B. Illustration of the Protocol using Zheng and Seberry Scheme

A word on the Encryption Scheme. The scheme can be viewed as an analogue to the encryption scheme of Bellare and Rogaway [2] which is proved to be secure against CCA. The one-way trapdoor function is the DHP, $\mathbf{R}_{|m|}[G(y_A^x)]$ acting as $G'(x)$, and the hash function, $\mathcal{H}$ concatanated with $\mathbf{L}_{|\mathcal{H}(m)|}[\mathrm{G}(y_A^x)]$ can be seen as a variation of $h(rx)$ of [2].

Alice's private key is x_A and the public key is $y_A = g^{x_A}$. Similarly, Bob's private key is x_B and the public key is $y_B = g^{x_B}$

1) **Alice's encryption:** Alice chooses a random number $x \in \{1, \ldots p\}$ and calculates $c_{1A} = g^x$ and $c_{2A} = G(y_A^x) \oplus (\mathcal{H}(m) \| m)$. She sends c_{2A} to Bob.
2) **Bob's encryption:** Bob chooses a random number $y \in \{1, \ldots, p\}$ and calculate $c_{1B} = g^y$. Taking some definite bits of the password as $seed$, Bob finds a pseudo-random number r, such that $|r| = |m|$. He calculates $m_1 = \mathbf{R}_{|m|}[c_{2A}] \oplus \mathrm{r}$ and $c_{2B} = \mathrm{G}(y_B^y) \oplus (\mathcal{H}(m_1) \| m_1)$. Bob sends c_{2B} and $H_{Bob} = \mathcal{H}(P, c_{2B})$ to Alice.
3) **Alice's decryption and Bob's authentication:** Alice authenticates the hash value H_{Bob} by calculating the hash of $P \| c_{2B}$. Now Alice decrypt c_{2B}. The steps are :

$$\begin{aligned} w &= G\big(c_{1A}^{x_A}\big) \oplus c_{2B} \\ &= G(c_{1A}^{x_A}) \oplus G(y_B^y) \\ &\oplus \big(\mathcal{H}(m_1) \| (\mathbf{R}_{|m|}[c_{2A}] \oplus r)\big) \\ &= G(y_B^y) \oplus G(y_A^x) \\ &\oplus \big(\mathcal{H}(m_1) \| (\mathbf{R}_{|m|}[G(y_A^x)] \oplus m \oplus r)\big) \\ &= G(y_B^y) \oplus \big((\mathbf{L}_{|\mathcal{H}(m)|}[G(y_A^x)] \oplus \mathcal{H}(m_1)) \| (m \oplus r)\big) \end{aligned}$$

Alice parse this as $a \| b$. She also computes r as Bob computes and then calculates $D_a = a \oplus \mathbf{L}_{|\mathcal{H}(m)|}[\mathrm{G}(y_A^x)] \oplus \mathcal{H}(\mathbf{R}_{|m|}[c_{2A}] \oplus r\ ) \oplus \mathcal{H}(m)$, and $D_b = b \oplus r$. Note that Alice has send $G(y_B^y) \oplus (m \| \mathcal{H}(m))$.
4) **Bob's decryption and Alice's authentication:** Bob decrypt using his private key by the decryption logic

$$\begin{aligned} z &= G(c_{1B}^{x_B}) = G(y_B^y) \\ w &= z \oplus D_A \\ &= G(y_B^y) \oplus G(y_B^y) \oplus (\mathcal{H}(m) \| m) \\ m &= \mathbf{R}_{|m|}[w] \end{aligned}$$

Bob checks the authentication of the message by hashing m and checking with $R_{|m|}[w]$. Now $m = K \| \mathcal{H}(K, P)$, so he can authenticate Alice by hashing $P \| \ m_{\{1,\ldots,|K|\}}$

V. Formal Security Analysis

Few of the security aspects are trivial to see. For example, No-Key Compromise Impersonation security is fulfilled trivially. However, the other security aspect are not so trivial. For the formal discussion of security in case of other form of attack, we will state the assumption and the claim on which we base the security of the protocol.

Assumption 1: The pseudo-random generator *G* is cryptographically secure and the hash function *H* is collision resistant.

Claim 1: The Encryption Scheme used in the protocol is secure against both CPA and CCA.

A. Impersonation Attack

Suppose Eve is the adversary who also construct a public/private key pair compatible with the encryption scheme. Eve tries to gain information about either the password or the session key. He has an oracle access to the Hash Oracle $\mathcal{O}_H$, a Pseudo Random Number Generating Oracle $\mathcal{O}_G$ and a decryption oracle $\mathcal{O}_D$. We consider the sitation in which dictionary size is not small to have a feasible

1) Eve as Alice: If there is an adversary which can find any bias on password or key during any course of protocol, he can invert a *pseudo-random generator* with non-negligible advantage.

Proof: Suppose Eve has some non-negligible advantage, (say $\frac{1}{2}$ + λ(n)) in gaining the information about the key or the password by impersonating Alice. He can gain information in the following stages of the protocol:

- In the step 1, by choosing a key $K_{Adv} \in K$ and a password $P_{Adv} \in \Pi$. He encrypts the message by his own defined public key. This is equivalent to the *chosen plaintext attack*. If a successful attack in this step is $Event_1$, then
$$\mathbf{Pr}[Event_1] \leq \frac{1}{2} + (poly(n))^{-1} \qquad (3)$$
- After he receives $\mathcal{H}(P, E_B)$ from Bob, Eve flips a coin. If it is 1, then he ask for a query $P_{Adv} \parallel E_B$ to $\mathcal{O}_H$. The probability of Eve gaining knowledge of the password is same as that for $P_{Adv} \parallel E_B$ satisfying
$$\mathcal{H}\big(P_{Adv} \parallel E_B\big) = \mathcal{H}\big(P \parallel E_B\big),$$
which equals the advantage in having a *Hash Collision*. If the coin flip is 0, Eve ask for the $\mathcal{O}_D$ the decryption corresponding to E_B. In the case of successful attack, this is equivalent to the *chosen cipher-text attack*, and will give him m_1. Since, he know $R_{|m|}[E_A]$, he can compute r. Now he can try to gain (partial) information of the password by using $\mathcal{O}_G$. He provides some random bit sampled from P_{Adv} as input to $\mathcal{O}_H$. The advantage in winning this game equals probability of finding two seed that gives same random number when given as input to Pseudo Random Generator. This probability we denote by Adv_A^{PSG}. If we denote the event of successful attack in this step as $Event_2$, then

$$\mathbf{Pr}[Event_2] = \tfrac{1}{2}Adv_A^{Coll} + \tfrac{1}{2}\big(Adv_A^{CCA}\big)\big(Adv_A^{PSG}\big)$$

Now,

$$\begin{aligned} Adv_A^{Alice} &= \mathbf{Pr}[Event_1] \\ &+ \mathbf{Pr}[\neg Event_1]\mathbf{Pr}[Event_2|\neg Event_1] \\ &\leq \frac{1}{2} + \frac{1}{poly(n)} \\ &+ \mathbf{Pr}[\neg Event_1]\mathbf{Pr}[Event_2|\neg Event_1] \\ &\leq \frac{1}{2} + \frac{1}{poly(n)} + \mathbf{Pr}[Event_2] \\ But, Adv_A^{Alice} &= \frac{1}{2} + \lambda(n) \\ \Rightarrow \frac{1}{2} + \lambda(n) &\leq \frac{1}{2} + \frac{1}{poly(n)} + \frac{1}{2}Adv_A^{Coll} \\ &+ \frac{1}{2}\big(Adv_A^{CCA}\big)\big(Adv_A^{PSG}\big) \end{aligned}$$

so, Adv_A^{PSG} is non-negligible.

Now, this is contradictory to the fact that the *pseudo-random generator* is cryptographically secure. Hence our original proposition that Eve has a non-negligible advantage in gaining information either about key or the password is wrong. ■

2) Eve as Bob: The only step at that Eve can win against the protocol is in the second step. This we say because if Eve fails in this step, Alice stops carrying out the later stages of the protocol. Eve receives E_A and has to send $\mathcal{H}(P, E_B)$ and E_A. He flips a coin. If the coin is 1, then Eve try to decrypt the E_A and gain information about the message. This is an instance of *chosen cipher-text attack* and let the advantage is Adv_A^{CCA}. Otherwise, he try to have a hash collision so that he can authenticate himself as Bob and hence gain the message at the last step of the protocol. The advantage of Bob is therefore, $\text{Adv}_A^{Bob} = \frac{1}{2}Adv_A^{CCA} + \frac{1}{2}Adv_A^{Coll}$ Now, we have assumed that the hash function is *collision Resistant*. Also, the *encryption scheme* is secure against CCA. Hence, Eve does not gain any information about the $password$ or the key.

B. Passive Attack

Eve can also launch passive attack. He can just eavesdrop to find the message exchange and then try to launch *offline dictionary attack* or some other form of cryptanalysis. The information that Eve has by eavesdropping are

E_A = $f(r_A)$ $\|$ G(r_A) $\oplus$ (m $\|$ $\mathcal{H}(m)$)
E_B = $f(r_B)$ $\|$ G(r_B) $\oplus$ (m_1 $\|$ $\mathcal{H}(m_1)$
D_A = $f(r_B)$ $\|$ G(r_B) $\oplus$ (m $\|$ $\mathcal{H}(m)$)
H_1 = $\mathcal{H}(P$,$E_B)$

He can easily calculate $\mathbf{R}_{|m|}[E_A]$. Now, by all the permutation of this three information and $\mathbf{R}_{|m|}[E_A]$, Eve can get just $m \oplus r$. Since r is a result of *pseudo-random generator*, the advantage by which Eve can derive message m equals the advantage of winning against *pseudo-random generator*. Hence, Eve cannot gain any (partial) information by *passive attack*.'However, if the password is of low entropy, then offline dictionary attack is possible. So, depending on the dictionary size, we can alternate between $\mathcal{H}(P, E_B)$ and $\mathcal{H}(r, E_B)$.

C. Dictionary Attack

In all the steps, password is either hashed or is sent encrypted by a scheme which is secure against CCA and CPA. Hence, all entries of the dictionary are the possible password. Hence, all that Eve can do is to follow the protocol by every entry of the dictionary. But, in most of the cases we have limited number of chances available to enter the password

D. Known Session Key Security

We will represent the session key in the i^{th} session as K_i, and add the superscript i to denote all the information exchanged by Alice and Bob during the exchange of K_i. Inorder to prove forward secrecy, we assume that we are at present communicating to share session key K_j, and session key K_i has been compromised for some $i < j$. The adversary has following informations:

$$\begin{aligned} E_A^i &= G(r_A) \oplus (m^i \| \mathcal{H}(m^i)) \\ E_B^i &= G(r_B) \oplus (m_1^i \| \mathcal{H}(m_1^i) \\ D_A^i &= G(r_B) \oplus (m^i \| \mathcal{H}(m^i)) \\ \mathcal{H}_1^i &= \mathcal{H}(P, E_B^i) \\ E_A^j &= G(r_A) \oplus (m^j \| \mathcal{H}(m^j)) \\ E_B^j &= G(r_B) \oplus (m_1^j \| \mathcal{H}(m_1^j) \\ D_A^j &= G(r_B) \oplus (m^j \| \mathcal{H}(m^j)) \\ \mathcal{H}_1^j &= \mathcal{H}(P, E_B^j) \end{aligned}$$

Now, the adversary can try for the *offline dictionary attack* to get the *password* from the hash $\mathcal{H}_1^i$, and then he can construct the message m^i. This allow him to get $G(r_B)$ and $G(r_A)$. This he can use to find the message and hence the key at any later stages. So, if Bob sends $\mathcal{H}(P, E_B^i)$, the protocol has no Known Session Key Security because it has no security against the offline dictionary attack. But if we use the variant of the protocol where Bob sends $\mathcal{H}(r, E_B^i)$ as the authentication, then we have security against the dictionary attack and consequently Known Session Key Security.

VI. Conclusion

In this paper, we provided a protocol that uses a different approach from the classical methods of key exchange. It takes the goods of both the variation of key agreement, the one using Symmetric Encryption and one involving Assymmetric Encryption. It encrypts the password (or some function of the password) by the Encryption Scheme as most of the Assymetric version do. However, we do not require to publicize our encryption key as in Symmetric Encryption scheme. The case where the brute force attack on the dictionary is feasible or infeasible in analysed concurrently, ammending the protocol only in one step.

A rather interesting possibilty to work upon is to construct a protocol that uses a Commutative Encryption Scheme, as for the lock-key analogy. This will reduce the complexity of the protocol by reducing the amount of work Alice has to do in the decryption stage. This, however require the construction to be in such a manner so as to make it Encryption independent.

Another interesting way to think is the use of Homomorphic Encryption Scheme. Alice encrypts the message by some homomorphic Encryption Scheme, and Bob further encrypts by multiplying by some random number, or some function that act as random number. Alice then decrypts by his decryption algorithm. Bob can then retrieve the key by multiplying by inverse of his random number.

ACKNOWLEDGMENT

The author wish to thanks Yassine Lakhnech and Abhishek Jain for the proof check.

REFERENCES

[1] M. Bellare, D. Pointcheval, and P. Rogaway, *"Authenticated Key Exchange Secure Against Dictionary Attacks",* Eurocrypt-2000: LNCS-1807, pages 139-155..

[2] M. Bellare and P. Rogaway, *"Random Oracles are Practical: A Paradigm for Designing Efficient Protocols",* ACM Conference on Computer and Communications Security 1993, pages 62-73

[3] S. Bellovin and M. Merritt, *"Encrypted Key Exchange: Password Based Protocols Secure Against Dictionary Attack",* Proceedings of Symposium on Security and Privacy, IEEE 1992, pages 72-84.

[4] M. Boyarsky, *"Public-Key Cryptography and Password Protocols: The Multi-User Case",* Proceedings of the 6th Annual Conference on Computer and Communications Security, ACM, 1999

[5] V. Boyko, P. MacKenzie, and S. Patel, *"Provably Secure Password-Authenticated Key Exchange Using Diffie-Hellman",* Proceedings of EUROCRYPT,2000, pages 156-171.

[6] W. Diffie and M. Hellman, *"New Direction in Cryptography",* IEEE Transaction on Information Theory, volume IT-22(6), November 1976, pages 644-654.

[7] D. Jablon, *"Strong Password only authenticated key exchange",* ACM Communication Review, 26(5):5-20, 1996.

[8] J. Katz, R. Ostrovsky, and M. Yung, *"Forward secrecy in password-only key exchange protocols",* In Proceedings of SCN-2002, LNCS 2576, Springer-Verlag, Berlin, pages 29-44.

[9] T. M. A. Lomas, L. Gong, J. H. Saltzer, and R. M. Needham, *"Reducing risks from poorly chosen keys",* ACM Operating Systems Review, 23(5), Pages 14-18. [Proceedings of the 12th ACM Symposium on Operating System Principles,Dec. 1989].

[10] S. Goldwasser and S. Micali, *"Probabilistic encryption"*, Journal of Computer and System Sciences vol 28, April 1984. Pages 270-299.

[11] P. D. MacKenzie, *"The PAK suite: Protocols for password-authenticated key exchange",* Technical Report 2002-46, DIMACS, 2002..

[12] P. MacKenzie and R. Swaminathan, *"Secure Network Authentication with Password Identifcation",* Submission to IEEE P1363a. August 1999.

[13] S. Weis, *"New Foundations for Efficient Authentication, Commutative Cryptography, and Private Disjointness Testing",* Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science.

[14] Y. Zheng and J. Serberry, *"Immunizing Public Key Cytosystems against Chosen Ciphertext Attacks",* IEEE Journal on Selected Areas in Communications 11(5) , pages 715-724 (1993).

Jalaj Kumar Upadhyay cleared the joint entrance examination of Indian Institute of Technology with a percentile of 99 and is at present pusuing his B-Tech course in Computer Science and Engineering, Institute of Technology, Banaras Hinidu University.