Security Analysis of Password-based Authenticated Key Exchange Protocols

Ashutosh Mishra, Sweta Mishra and Sunil Panday*

***Abstract*—Password-based cryptosystems commonly suffer from dictionary attacks because their security depends on low-entropy passwords. It is ever challenging to design a password-based cryptosystem secure against this attack. Password-based authenticated key exchange (PAKE) protocols allow two or more interacting parties to establish cryptographic keys based on their knowledge of a password. The PAKE protocols commonly use password-based encryption and are therefore susceptible to dictionary attacks. Many existing PAKE protocols are claimed to be secure against these dictionary attacks, but there is no easy method to verify their claim. In this work, we focus on evaluating the security of two-party PAKE protocols under possible attack scenarios. We first consider all possible combinations of participants of an attack scenario, which turn out to be 5 in number. This gives rise to 25 possible attack scenarios among the participants. We find that 11 out of these 25 scenarios are valid. We then analyze the security of 5 PAKE protocols under the attack scenarios developed by us. Namely, we analyze EKE, SPEKE, SRP, KOY and IdBP protocols, which are considered significant protocols over the time. We also provide some suggestions on improving existing PAKE designs.**

***Index Terms*—Password, Password-based authentication, PAKE protocols, Key exchange protocol, Dictionary attack.**

I. Introduction

PASSWORD-based cryptosystems are widely used because they are considered convenient by the users, apart from being the least expensive technique to provide authentication. They suffer from the inherent weakness of low-entropy passwords, which forces the overall system to be susceptible to dictionary attacks against password guessing. It is ever challenging to derive a strongly secure system from an insecure source such as passwords. Therefore, the new paradigm towards the PAKE protocols is to provide security beyond dictionary attacks. This is made possible by introducing randomness independent of the secret password (thus enlarging the entropy pool) and by providing no way to verify the correctness of the guess for the secret password (thus preventing the attacker from knowing if her guess is correct). Another important requirement of PAKE protocols is "forward secrecy" [1]. This ensures that the previously used session key remains secure even if the long-term secret (password) is compromised. Moreover, a PAKE protocol should be able to prove the authenticity of the communicating parties while sharing a session key, messages should not be tampered with without detection, and each instance of the protocol should be uniquely identified.

Manuscript received November 26, 2024; revised April 8, 2025.

Ashutosh Mishra is a research scholar in the Department of Mathematics, National Institute of Technology, Manipur, India-795004 (e-mail:iitd.ashu@gmail.com).

Sweta Mishra is an Assistant Professor in the Department of Computer Science and Engineering, Shiv Nadar Institution of Eminence, Delhi NCR, India-201314 (e-mail:sweta.mishra@snu.edu.in).

Sunil Panday is an Associate Professor in the Department of Mathematics, National Institute of Technology, Manipur, India-795004 (Corresponding Author, e-mail: sunilpanday@hotmail.co.in).

A. Motivation

There exist many PAKE protocols, but most of the protocols which have attracted practical interest have only some heuristic security analysis. Encrypted Key Exchange (EKE) [2] is the first PAKE protocol to provide security beyond the dictionary attack but without proof of security. Protocols such as AEKE [3], REKE [4], SPEKE [1] etc. were later designed following the same approach as EKE. The security proof of a variant of EKE, termed 'EKE2', appeared in Eurocrypt '2000 in [5]. However, this proof was made in the ideal cipher model, an attack model that does not guarantee real-world security [6]. The scheme proposed in [7] is the first provably secure protocol in the standard model. In this work, a large number of parameters are considered to provide proof of the security of the scheme. This constraint is due to the challenge posed by PAKE protocols in providing strong security derived from insecure sources such as passwords.

The security analysis of the PAKE protocols has been heuristic in some cases and following well-defined models (ROM [5] or Standard model [7]) in some others. However, existing models of provable security are not very easy to follow for industry implementers of PAKE protocols. More importantly, existing proof models are difficult to apply to some of the proposed protocols.

This has led to some deployed PAKE protocols without well-accepted proof of their security, such as SPEKE [1], SRP [8] etc. Further, new attack approaches are emerging with changing technology and hence there is a need to consider stronger adversarial models for analysis. For example, Chang and Yung, in a rump-session presentation in Crypto '2012, [9] introduced a new attack model which they termed the 'Midgame Attack'. In this attack model, a powerful adversary can have complete (but for a short time) access to a system at some point in the middle of a cryptographic computation. The model was motivated with respect to its applicability to hash functions, but this appears to be a practical approach considering PAKE protocols as well.

In [10], Bellare and Rogaway define an adversarial model considering stronger adversarial approaches to analyze the security of session key distribution under a three-party scenario. The idea was easily implementable in other protocol settings and in [5] it was used for the two-party scenario. The communication model has been formulated considering the situation in which the communication between two parties is entirely controlled by the adversary. The adversary's interaction with the participants is modeled through queries to different oracles as follows.

TABLE I
ANALYSIS OF THE PROTOCOLS UNDER PROPOSED ATTACK SCENARIOS. A : SECURITY AT SYSTEM A, B: SECURITY AT SYSTEM B, IN: INSIDER, OUT: OUTSIDER, ON: ONLINE, OF: OFFLINE, DLP: DISCRETE LOG PROBLEM, $|k|$: THE LENGTH OF THE SESSION KEY k, FA: FULLGAME-ACCESS, MA: MIDGAME-ACCESS, μ: PASSWORD ENTROPY, μ_A: ENTROPY OF PASSWORD OF USER A, μ_B: ENTROPY OF PASSWORD OF USER B.

2*Attack Scenarios		Protocol				
		EKE [2]	**SPEKE** [1]	**SRP** [8]	**KOY** [7]	**IdBP** [15]
Case I		Secure	Secure	Secure(A)/2^{μ}(of, B)	Secure(A)/2^{μ}(of,B)	Secure
2***Case II**	In	Insecure	Insecure	2^{μ}(of, B)	2^{μ} (of, B)	Secure
	Out	2^{μ} (on)	2^{μ} (on)	2^{μ} (on)	2^{μ} (on)	Secure
Case III		2^{μ} (on)	2^{μ} (on)	2^{μ}(on,A)/2^{μ}(of,B)	2^{μ} (on)/2^{μ}(of)	2^{μ_A} (on)
Case IV		$2^{\lvert k\rvert}$	DLP	DLP	DLP	Secure
2***Case V**	FA	Insecure	Insecure	Insecure	Insecure	Insecure
	MA	2^{μ} (of)	DLP	2^{μ} (of)	DLP(A)/2^{μ}(of,B)	DLP
2***Case VI**	In	Insecure	Insecure	2^{μ} (of, B)	2^{μ} (of, B)	2^{μ_A} (of)
	Out	2^{μ} (on)	2^{μ} (on)	2^{μ}(on,A)/2^{μ}(of,B)	2^{μ}(on,A)/2^{μ}(of,B)	2^{μ_A} (on)
Case VII		2^{μ} (on)	2^{μ} (on)	2^{μ} (of)	2^{μ} (of)	$2^{\mu_A+\mu_B}$ (of)
Case VIII		$2^{\lvert k\rvert}$	DLP	DLP(A)/2^{μ}(B) (of)	DLP(A)/2^{μ} (of,B)	DLP
2***Case IX**	FA	Insecure	Insecure	Insecure	Insecure	Insecure
	MA	2^{μ} (of)	DLP	2^{e} (of)	DLP(A)/2^{μ}(of,B)	$2^{\mu_A+\mu_B}$ (of)
Case X		2^{μ} (on)	2^{μ} (on)	2^{μ}(on,A)/2^{μ}(of,B)	2^{μ} (on)/ 2^{μ} (of)	2^{μ_A} (on)
2***Case XI**	FA	Insecure	Insecure	Insecure	Insecure	Insecure
	MA	2^{μ} (of)	DLP	DLP	DLP(A)/2^{μ}(of,B)	DLP or 2^{μ_A}(on)

The 'Send' oracle accepts messages fabricated by the adversary and responds following the protocol description.

The 'Reveal' oracle provides the session key. The 'Corrupt' oracle provides the secret password. For proof of security, considering these oracle queries under the standard model, which is the most realistic model, it is not always possible to apply to some most widely recognized protocols like EKE [2], SRP [1] etc.

This restriction is again due to the security dependency of PAKE on inherently insecure passwords.

This is the reason why we provide the general model for analyzing security considering possible attack scenarios. Our midgame approach as described in Section 2 provides a similar concept as the 'Reveal' oracle. We use the term midgame approach to better explain the adversarial approach in the scenarios considered.

We also analyze the security of some existing systems under our proposed scenarios. Specifically, we performed an analysis on 5 PAKE protocols which are commonly regarded as significant constructions. EKE [2] is considered as a seminal work, SPEKE [1] (BlackBerry inter-process protocol uses SPEKE protocol to generate the session key [11]) and SRP [8] (SSL/TLS supports SRP ciphersuites [12]) are practically implemented protocols, KOY [7] is the first provably secure design under standard model. The IdBP [15] protocol which is not actually considered as a PAKE protocol, is included to show the possible implementation and analysis on the design which implements Id-based encryption.

Our contribution: The contributions are as follows.

- We characterize possible attack scenarios for two-party PAKE protocols, including the new attack model such as the 'midgame attack'. In all, we describe 11 attack scenarios.
- We consider all possible combinations of participants during an attack scenario. For example, for a particular instance of the protocol during the communications between two involved parties A and B, a powerful attacker can actively check all internal computation of system A or B.
- We analyze 5 well-known PAKE protocols in our proposed scenarios to verify the security claims of these designs.
- We provide suggestions for improving existing designs in light of our findings.

The results of our analysis of the 5 PAKE protocols are given in Table I. We consider 11 possible attack scenarios, the details of which are explained in section II. To consider possible attacker approaches, we include different modes of the attacker such as: insider (legitimate party acting as an attacker), outsider (third party acting as an attacker), eavesdropper, attacker with idle-system access, attacker with midgame-access (access of the system at some point in the middle of the computations) and attacker with fullgame access from the beginning of the computation of the protocol. For our analysis μ is the entropy of the passwords and therefore the dictionary attack complexity is 2^{μ} (online or offline).

Recently, we have some strong proposals like OPAQUE [14], an Asymmetric Password Authenticated Key Exchange Protocol (aPAKE) which defends against some vulnerabilities of password-over-TLS [13]; however, analysis of aPAKE is outside the scope of this paper and can be considered as a future work.

The rest of the document is organised as follows. In Section 2 we describe our proposed possible attack scenarios for PAKE protocols. A brief description of 5 PAKE protocols and their detailed security analysis based on our proposed attack scenarios is presented in Sections 3 and 4 respectively. Subsequently, suggestions for improvement of existing designs are provided in Section 5. We conclude this work in Section 6.

II. POSSIBLE ATTACK SCENARIOS TAKING EXHAUSTIVE APPROACH

The main goal behind the design of PAKE protocols is to increase the entropy of the overall system from the traditional password entropy by preventing dictionary attacks against password guessing. Existing techniques to analyze the security of PAKE protocols are not applicable

TABLE II
NOTATIONS

A, B	The system principles, user A and user B respectively.
pwd_x	The password (pwd) generated by user x.
μ	Entropy of user generated passwords.
r_i	Random number where $i \in \mathbb{N}$.
$r \xleftarrow{\$} \mathbb{Z}$	x randomly generated from $\mathbb{Z}$.
$E(.)$	Encryption function.
$D(.)$	Decryption function.
$E_{pwd_x}(m)$	Symmetric encryption of m with password pwd of user x.
E_{pk_x}	Encryption with public key (pk) of user x.
E_{sk_x}	Encryption with secret key (sk) of user x.
k	The session key.
$Challenge_x$	Random challenge selected by x.
E	Attacker.

to most of the practically accepted protocols as it is hard to prove security under a standard model when the secret information (i.e., password) is inherently insecure. Therefore, it is difficult to verify the security of such protocols. In this work, we propose possible attack scenarios to analyze any two-party PAKE protocol. Throughout the paper, the scenarios shown for A (user/system) are equally applicable to B (user/system). For our work, we have taken the assumption that attacker E cannot have simultaneous access to the internal states of two different systems. The following section describes possible approaches taken by the attacker. The notations are listed in Table II.

A. Different Modes of Attack

To describe our work, we incorporate possible approaches taken by an attacker which is shown in Fig. 1. This categorization depends on the time and mode of access.

- **Eavesdropping:** This is the mode of attack where the attacker gets unauthorized access to data in communication during the protocol execution.
- **Idle-system access:** This is the mode of attack where the attacker gets access to internal states of the system of a legitimate user of the protocol when the system is in an idle state, i.e., not involved in the execution of the protocol.
- **Insider:** This is the mode of attack where a legitimate participant of a protocol acts as an attacker to compromise unknown secrets.
- **Outsider:**This is the mode of attack where any third party acting as an attacker impersonates a legitimate participant in communication with another legitimate participant.
- **Fullgame-access:** This is the mode of attack depending on the time of access of the internal computations inside a system of a legitimate user by an attacker during the protocol execution. This includes the scenario where an attacker of a PAKE protocol can have access to all information which is involved in a computation from the beginning of the protocol execution.
- **Midgame-access:** This is the mode of attack depending on the time of access of the internal computations inside a system of a legitimate user by an attacker during protocol execution. For PAKE protocols, the key establishment computations usually take a very short time, and therefore it is difficult to get access to the computations of key establishment. We define the midgame-access as the scenario where attacker gets access to the system of a user after key establishment, i.e, he gets the access of session key and all computations afterward. The idea of 'midgame-attack' is introduced by Chang and Yung in [9] and we are following the same idea for this attack mode.

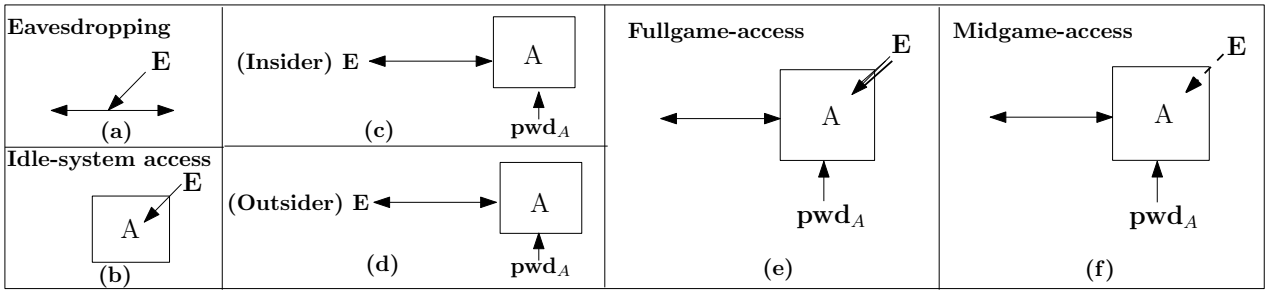
B. Detailed Description of Possible Attack Scenarios

To reach our conclusions, we take the following approach. We first check all possible participants to describe an attack which contributes the 5 attack scenarios listed below.

1. System of A and attacker E, where E accesses internal states of the system.
2. System of A, user A and attacker E, where E acting as an insider or outsider attacker.
3. System of A, user A, attacker E and system of B, where E impersonates user B.
4. System of A, user A, attacker E, system of B and user B, where E eavesdrops the communications between users A and B.
5. System of A, user A, attacker E, system of B and user B, where E being more powerful eavesdrops the communications between users A and B and also access the internal computations of system of A or B.

Taking all permutations of the above 5 attack scenarios, we find 11 valid cases among 25 cases which exhaust all possibilities. The resulting 11 cases of possible attack scenarios are described below, and the graphical overview is provided in Fig. 2.

- **Case I: E with system of A.** This is the case when attacker E has access to internal states of system of A, i.e., idle-system access of system of A.
- **Case II: E interacts with user A.** This is the case when insider or outsider E (see Fig. 1(c) & (d)) interacts with user A to extract her secret password.
- **Case III: E with system of B, interacts with user A.** This is the case when E with idle-system access of system of B and without his involvement tries to impersonate as B in a communication with user A.
- **Case IV: E intercepts communications between users A and B.** This is the case when E can only intercept the communications between users A and B during protocol execution and tries to get the secret password.
- **Case V: E intercepts communications between users A and B with access to system B.** This includes the cases which provide E (i) the fullgame-access or (ii) the midgame-access (see Fig. 1(e) & (f)) of system of B to access the internal states and also the ability to intercept the communications between users A and B.
- **Case VI: E with system of A, interacts with user A.** This is the case when E either be an insider or outsider (see Fig. 1(c) & (d)) has the idle-state access of system of A and executes the protocol interacting with user A.
- **Case VII: E with system of A and B, interacts with user A impersonating B.** This is the case when E has access (not simultaneously) to both the internal states (idle-state access) of systems A and B and tries to



E is the attacker and A is a legitimate user of a protocol. pwd_A is the password used at system A. (a) E eavesdrops communication channel. (b) E with idle-system access the system of A. (c) E an insider attacker communicates with A (d) E an outsider attacker communicates with A (e) E with fullgame-access over system A from the beginning of the protocol execution. (f) E with midgame-accessl of system A, i.e., access of the system after key establishment computation of the protocol execution

Fig. 1. Different modes of attack by attacker E

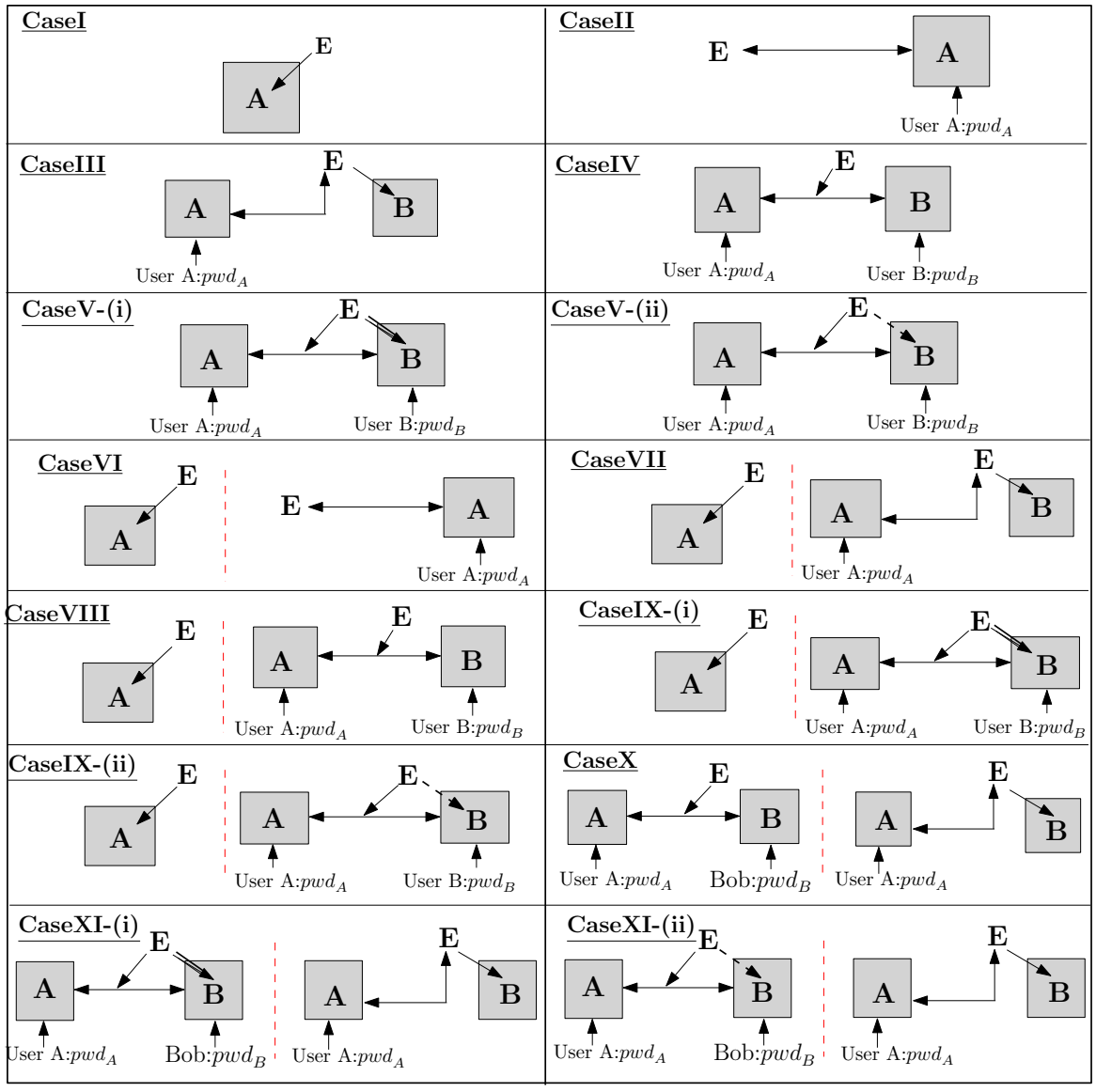


Fig. 2. Overview of possible attack scenarios, pwd_X is the password at system X.

impersonate B during the execution of protocol with user A.

- **Case VIII: E with system of A, intercepts communications between users A and B.** This is the case when E has the idle-state access of system of A and tries to intercept messages exchanged between users A and B during protocol execution.
- **Case IX: E with system of A, intercepts communications between users A and B with access to system of B** This is the case when E with idle-state access of system of A, intercepts the communications between users A and B with (i) the fullgame-access or (ii) the midgame-access (see Fig. 1(e) & (f)) of system of B.
- **Case X: E intercepts communications of users A and B and further with system of B, interacts with user A.** This is the case when E intercepts messages of communications between users A and B and tries to impersonate as B in further communication with user A.
- **Case XI: E intercepts communications between users A and B with access of system of B and then impersonates B to communicate with user A.** This is the case when E intercepts the communications between users A and B with (i) the fullgame-access or (ii) the midgame-access (see Fig. 1(e) & (f)) of system of B and then tries to impersonate as B in further communication with user A.

In the following sections, we provide an overview and detailed analysis of some significant PAKE protocols under the above attack scenarios.

III. Overview of Some Significant PAKE Protocols

In this work, we analyze 5 significant PAKE protocols. An overview of these 5 PAKE protocols is discussed in this section.

A. Encrypted Key Exchange Protocol [2]

This work is considered as the seminal paper in the field of PAKE protocols with claimed security resistance to dictionary attack. In this protocol, system principles: user A and user B, share a common password pwd offline and exchange a session key using both symmetric and asymmetric encryptions as shown in Fig. 3, enumerated in the sequence of the flow of the protocol. A generates ephemeral asymmetric key pairs (pk_A, sk_A) and sends pk_A to B encrypted with pwd. B then generates a session key, k and sends it to A, double encrypted first with pk_A and then with pwd. At the end, both parties verify the correctness of k by exchanging individual challenges. Overall, the protocol requires five communication rounds and six encryptions.

1) Security analysis: Use of a shared password prevents the individual freedom from changing the password and is insecure even if any of the users become malicious. In its usual implementation, if the session key k is leaked, it can be verified at steps 4 and 5. To guess password with compromised k, the protocol provides verifiable texts at step 2, which is explained in section IV-A. Therefore, it does not provide forward secrecy, as a compromised password can reveal all previously generated session keys.

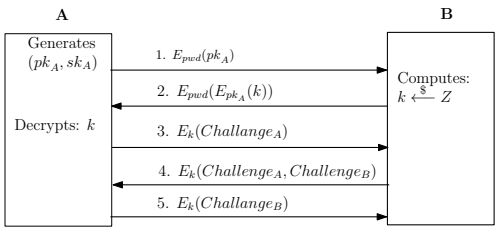


Fig. 3. Block diagram of Encrypted Key Exchange protocol

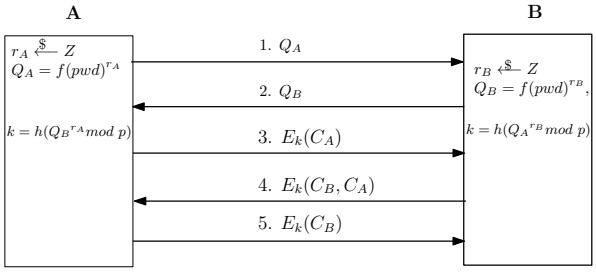


Fig. 4. Block diagram of SPEKE protocol

B. Simple Password Exponential Key Exchange Protocol (SPEKE) [1]

This is a protocol which uses Deffie-Hellman (DH) key exchange [16] approach but the base of the DH is derived from common password pwd agreed between users A and B implemented on a function f. Specifically the computation $f(pwd)$ outputs a value or base of a large prime order. After the base computation A generates a random number r_A and then $Q_A = f(pwd)^{r_A}$. A sends Q_A to B. Similarly B computes r_B and $Q_B = f(pwd)^{r_B}$ and sends Q_B to A. The session key k is computed as $k = H({Q_B}^{r_A} mod p) = H({Q_A}^{r_B} mod p)$ where ${Q_B}^{r_A} = {Q_A}^{r_B}$. Next A and B prove the possession of the same k in next subsequent message exchanges following challenge-response technique. Different choices for function f is discussed in [1], but a fully constraint SPEKE method is defined to use $f = (pwd)^2$. The overview of the protocol is depicted in Fig. 4.

1) Security analysis: Password pwd is the memorized secret shared between two parties A and B

and prone to insider attack. In [17], it is shown that an adversary can test multiple possible passwords using a single impersonation attempt. The password space can be divided into equivalence classes and in [17] it is termed as exponential equivalence classes (as password is used as exponent). This result contradicts the claim of SPEKE that it is as strong as EKE [2]. Another analysis [18] on SPEKE shows with an easy description that when user A initiates communication with user B, E can easily intercept the message and can start impersonating B by initiating another session showing from B (but actually from E) to A and replaying messages from first session with little modification. With two such parallel sessions, E can easily be authenticated in both the sessions without his knowledge of the secret pwd. This shows impersonation is possible without detection. The paper [18] also shows that SPEKE is susceptible to key malleability attack, i.e., exchanged messages can be tampered with and used to generate session key without detection of the tampering.

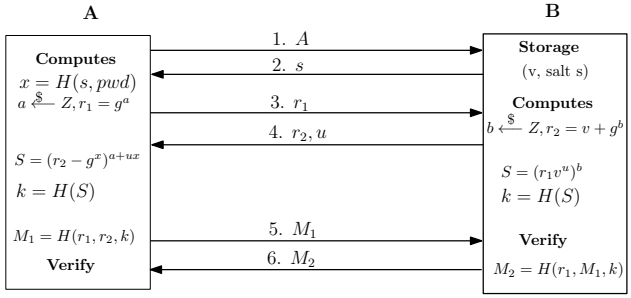
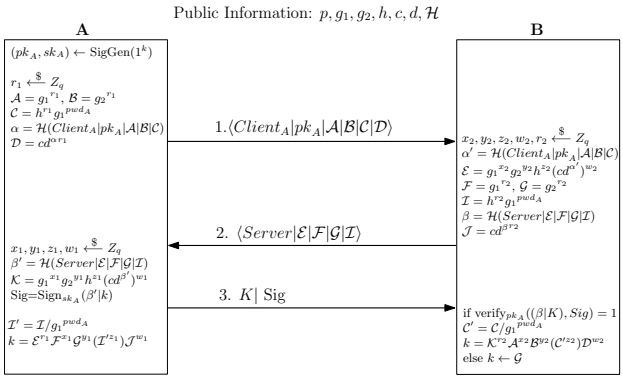


Fig. 5. Block diagram of SRP protocol



The public parameters are generated either by a trusted third party or a source of randomness which generates p, q and generators g_1, g_2, h, c, $d \in \mathcal{G}$ along with a universal one-way hash function $\mathcal{H}$, where p, q are primes such that $q|(p-1)$ and $\mathcal{G}$ be a subgroup of $Z_p{}^*$ of order q

Fig. 6. Overview of key-exchange of KOY protocol

C. Secure Remote Password Protocol (SRP) [8]

This protocol is designed for the client-server model and claims to prevent active online attacks mainly by never transmitting the client (A)'s password pwd over the wire. The server (B) stores the verifier v and the salt s where $x = H(pwd, s)$, H is the hash function and $v = g^x$. To initiate the protocol, client A sends her identity to server B. B responds by sending salt s to A. A computes $x = H(s, pwd)$, and $v = g^x$. A generates a random number a to compute $r_1 = g^a$. A sends r_1 to B. B generates random numbers b, u and computes $r_2 = v + g^b$, $S = (r_1 v^u)^b$ and session key $k = H(S)$. B sends to A the values r_2 and u. Upon receiving the values, A also computes $S = (r_2 - g^x)^{a+ux}$ to generate the session key $k = H(S)$. After key generation, both A and B verify the possession of k following challenge-response technique. It requires six communication rounds but no encryption of communicated messages. The graphical overview of the protocol is presented in Fig. 5.

1) Security analysis: Server B stores the verifier corresponding to client's secret password and if the value is compromised, then it needs the effort of offline dictionary attack, i.e., 2^{μ} (μ: password entropy) to disclose pwd. Impersonation attack is possible by the exposure of verifier v. The protocol claims to protect password from network eavesdroppers by never sending it over the wire.

D. Katz, Ostrovsky, Yung Protocol (KOY) [7]

This protocol is the first povably secure protocol under the standard cryptographic model, and it is based on Cramer-Shoup [19] public key cryptosystem with an extension which introduces two different encryptions named client-encryption and server-encryption. The assumption is that the public parameters are generated either by a trusted third party or a source of randomness which generates p, q and generators g_1, g_2, h, c, $d \in \mathcal{G}$ along with a universal one-way hash function $\mathcal{H}$, where p, q are primes such that $q|(p-1)$ and $\mathcal{G}$ be a subgroup of $\mathbb{Z}_p{}^*$ of order q in which DDH assumption holds [7]. The protocol follows the client-server model where client A generates password pwd_A for authentication. The server stores a vector $PW_s = \langle pwd_A \rangle_{A \in Client}$ where Client is the set of all clients of the protocol. When client A interacts with server B, it first generates public key pk_A and secret key sk_A, for a one-time signature scheme. Then client computes $\mathcal{A} = g_1{}^{r_1}, \mathcal{B} = g_2{}^{r_1}, \mathcal{C} = h^{r_1} g_1{}^{pwd_A}, \alpha = \mathcal{H}(Client_A|pk_A|\mathcal{A}|\mathcal{B}|\mathcal{C}), \mathcal{D} = (cd^{\alpha})^{r_1}$ where r_1 is a random value, and sends at step 1 the value $\langle Client_A|pk_A|\mathcal{A}|\mathcal{B}|\mathcal{C}|\mathcal{D}\rangle$ to B. The server B then selects random values x_2, y_2, z_2, w_2, r_2 and computes $\alpha' = \mathcal{H}(Client_A|pk_A|\mathcal{A}|\mathcal{B}|\mathcal{C}), \mathcal{E} = g_1{}^{x_2} g_2{}^{y_2} h^{z_2} (cd^{\alpha'})^{w_2}, \mathcal{F} = g_1{}^{r_2}, \mathcal{G} = g_2{}^{r_2}, \mathcal{I} = h^{r_2} g_1{}^{pwd_c}, \beta = \mathcal{H}(Server|\mathcal{E}|\mathcal{F}|\mathcal{G}|\mathcal{I}), \mathcal{J} = cd^{\beta r_2}$. At step 2, B sends A the value $\langle Server|\mathcal{E}|\mathcal{F}|\mathcal{G}|\mathcal{I}\rangle$ to A. The client then generates randomly x_1, y_1, z_1, w_1 and computes $\beta' = \mathcal{H}(Server|\mathcal{E}|\mathcal{F}|\mathcal{G}|\mathcal{I})$ then $\mathcal{K} = g_1{}^{x_1} g_2{}^{y_1} h^{z_1} (cd^{\beta'})^{w_1}$. Finally, client computes $\text{Sig}=\text{Sign}_{sk_A}(\beta'|\mathcal{K})$ and at step 3, sends $\mathcal{K}|Sig$ to B. Session key k then computed by client and server as $k = \mathcal{E}^{r_1} \mathcal{F}^{x_1} \mathcal{G}^{y_1} (\mathcal{I}'^{z_1}) \mathcal{J}^{w_1} = \mathcal{K}^{r_2} \mathcal{A}^{x_2} \mathcal{B}^{y_2} (\mathcal{C}'^{z_2}) \mathcal{D}^{w_2}$. The overview of the key exchange protocol is provided in Fig. 6.

1) Security analysis: The server stores the client password pwd_A in a vector $PW_s = \langle pwd_A \rangle_{A \in Client}$ where Client is the set of all clients of the protocol. As a practical consideration, the paper [7] suggests to store $g_1{}^{pwd_A}$ instead of pwd_A where generator g_1 is a public parameter. To get pwd_A, the attacker can try all possible choices of pwd_A as pwd'_A and check if $g_1{}^{pwd_A} = g_1{}^{pwd'_A}$ when server data is compromised. This requires the complexity of offline dictionary attack. The authors also claim that the protocol is secure if dictionary attack is the best approach to compromise the password. In [20] it is demonstrated that the KOY protocol does not achieve forward secrecy and with little modification the modified protocol satisfies the requirement of forward secrecy.

E. Authenticated ID-based key Exchange and remote log-in with simple token and PIN [15] (IdBP)

This protocol is based on an Id-based public key cryptosystem where the involved parties A and B get their secret keys from a trusted authority (TA). The TA computes a random master secret s. When users A and B request their key pairs providing identities Id_A, Id_B respectively, TA computes the hash of corresponding identities and generates points $\mathcal{A}$ and $\mathcal{B}$ on the elliptic curve. Both users A and B receive their key pairs as $(\mathcal{A}, s\mathcal{A})$ and $(\mathcal{B}, s\mathcal{B})$ from TA respectively. A generates her password α(a memorized value) and computes $s\mathcal{A} - \alpha\mathcal{A} = (s - \alpha)\mathcal{A}$ and stores $(s - \alpha)A$ in her system. Similarly, B stores $(s - \beta)\mathcal{B}$ where β is the password of B. The value s is unknown to both A and B as

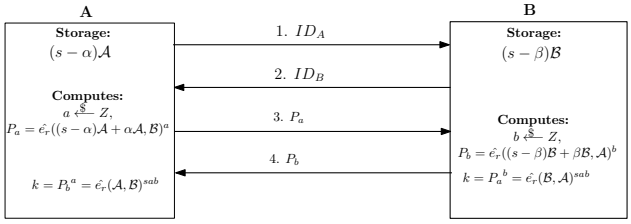


Fig. 7. Block diagram of Scott's ID-based Key Exchange protocol

the difficulty of getting the value depends on the difficulty in solving discrete log over elliptic curve which is assumed to be difficult. The computations are based on a bilinear mapping called distorted Tate pairing [15] and the protocol uses the concept of Deffie-Hellman key exchange [16] to derive the session key k. A and B choses their random values a and b and computes $P_a = \hat{e_r}((s-\alpha)\mathcal{A} + \alpha\mathcal{A}, \mathcal{B})^a$, $P_b = \hat{e_r}((s-\beta)\mathcal{B} + \beta\mathcal{B}, \mathcal{A})^b$ respectively. The session key k is computed as $(P_a)^b$ or $(P_b)^a$ and the overview of the protocol is shown in Fig. 7.

1) Security analysis: In this protocol, both users A and B have their own secret passwords which they use to protect their individual secret keys. The protocol is not secure against insider attack because of the inherent weakness of the bilinear mapping called distorted Tate-pairing [15]. Therefore, if A gets access to system of B and then the value $(s-\beta)\mathcal{B}$, it is easy for A to get the password β of B by applying a dictionary attack listing all possible choices. Therefore, the protocol is not resistant to identity theft, i.e., it is easy to impersonate B after A compromises the password of B. It is claimed in the paper [15] that determining session key k is equivalent of solving Bilinear Deffie-Hellman problem [21].

IV. Detailed Analysis of Protocols under Proposed Attack Scenarios

The detailed analysis under the proposed attack scenarios of the 3 protocols, discussed in section III, is provided below and analysis of other 2, i.e, SPEKE [1] and SRP [8] protocols are omitted as the analysis of SPEKE is similar to EKE and the analysis of SRP is similar with KOY protocol. We assume that the applied asymmetric or symmetric cryptosystems do not allow any partial information leakage. Our analysis aims to show the complexity of compromising the secret password. The resulting analysis is summarized in Table I. Unless specified, the analysis of A (user/system) is equally applicable to B (user/system).

A. Encrypted Key Exchange Protocol [2]

In this protocol users A and B share the password pwd, hence $pwd_A = pwd_B = pwd$. The protocol uses both symmetric and asymmetric encryptions. In a communication between users A and B, first the ephemeral asymmetric key pairs (pk_A, sk_A) are generated by A for each protocol execution. Then at step 1, A sends B, $E_{pwd}(pk_A)$, i.e., pk_A encrypted with pwd. Then B generates a random session key k and at step 2 sends k to A as a double encrypted value $E_{pwd}(E_{pk_A}(k))$. A gets k after decryption first using pwd as key and then with sk_A. Finally both A and B verifies the knowledge of k following challenge-response technique as shown in 3. For the below analysis, we assume μ as the entropy of the password pwd. All equivalent cases are discussed together.

Case I: This is the case when E gets idle-system access to system of A. E tries to compromise the password pwd which is a memorized entity. Therefore, E gets no information to execute the attack and hence protocol is secure.

Cases II and VI: This is the case when E, being an insider or outsider, communicates with user A. As idle-system access of system of A does not provide any information (discussed for case I), cases II and VI can be analyzed equivalently. Now for the insider E, i.e., when B acts as E, E knows the secret pwd and can behave in any unwanted way to make the protocol totally insecure. For outsider E, without knowledge of pwd, E can initiate the protocol with a guess of pwd'. E generates asymmetric key pairs (pk_E, sk_E), encrypts with pwd' as $E_{pwd}(pk_E)$ and sends to A at step 1. Then A decrypts with pwd and gets ${pk_E}'$, generates session key k and sends the double encrypted value $E_{pwd'}(E_{pk_E}(k))$ to E at step 2. All the communicated values are random. Therefore, it is difficult to verify the correctness of pwd' till step 2. If ($pwd' \neq pwd$), it is only detectable at the verification stage after generation of session key k (steps 3 to 5 of Fig. 3) which terminates the instance of the protocol. Therefore, with maximum of 2^{μ} online efforts, E can get the pwd by checking possible guesses and the guess can be verified when the protocol executes successfully.

Cases III, VII and X: The idle-system access of system A provides no information and compromising pwd from intercepted communications of users A and B is the most challenging scenario (see analysis for case IV). Therefore, cases III, VII and X follow the same analysis. The scenario covered for analysis provides E with the idle-system access of system of B and E tries to impersonate B in communication with A. Without the knowledge of pwd, E initiates the protocol with a guess for pwd as pwd'. E generates the ephemeral asymetric key pairs (pk_E, sk_E) and encrypts pwd' as $E_{pwd'}(pk_E)$ and sends to A. A receiving the value and after decryption may get some other random value ${pk_E}'$ if $pwd \neq pwd'$. Next, A generates a random session key, k and encrypts it first with ${pk_E}'$ and then with pwd which produces $E_{pwd'}(E_{pk_E}(k))$ which is again random. Therefore, the initial three steps of the protocol are difficult to verify for correctness. Remaining steps from 3 to 5 verify the correctness of k through challenge-response technique as shown in Fig. 3 which detects if k is incorrect. This way, k can be correctly verified when $pwd = pwd'$. This requires maximum 2^{μ} online efforts of E to get the correct pwd.

Cases IV and VIII: Case VIII includes the scenario of idle-system access of system A (provides no information) along with the scenario of case IV. Therefore, both the cases are equivalent. Specifically, in this case, E intercepts the communication between users A and B i.e., steps 1-5 (see Fig. 3). E gets all the encrypted values $E_{pwd}(pk_A)$, $E_{pwd}(E_{pk_A}(k))$, $E_k(challenge_A)$, $E_k(challenge_A, challenge_B)$ and $E_k(challenge_B)$. Let E guess pwd' and gets $pk'_A = D_{pwd'}(E_{pwd}(pk_A))$ but without knowing k it is difficult to compute

further. If k is known, then the guess for pwd can be verified as the guess is pwd'. E decrypts the 1st message and gets $pk'_A = D_{pwd'}(E_{pwd}(pk_A))$ and checks if $E_{pwd}(E_{pk_A}(k)) = E_{pwd'}(E'_{pk_A}(k))$ which gives $pwd = pwd'$. To guess the value k, it requires the effort of $2^{|k|}$ ($|k|$ =length of k), an expensive task. Therefore, the complexity is $2^{|k|}$ to get pwd.

Cases V, IX, XI: These are the cases where E gets fullgame-access or midgame-access to system B. As idle-system access to system A or B does not provide any information, therefore all these cases are equivalent. When E gets fullgame-access to system B, he gets all the values involved in protocol computation including password, hence protocol becomes insecure. For midgame-access we take the assumption that the computations after key establishment are available to E, i.e., the session key along with the intercepted messages of communication between A and B. Therefore, E has the values $E_{pwd}(pk_A)$, $E_{pwd}(E_{pk_A}(k))$, $E_k(challenge_A)$, $E_k(challenge_A, challenge_B)$ and $E_k(challenge_B)$ from interception and the value k from midgame-access, where pwd is the common password and pk_A is the public key generated by A. With k the guess for unknown pwd can easily be verified with 2^μ offline efforts. Specifically, E guesses pwd', computes $pk'_A = D_{pwd'}(E_{pwd}(pk_A))$ and checks if $E_{pwd}(E_{pk_A}(k)) = E_{pwd'}(E'_{pk_A}(k))$ then $pwd = pwd'$.

B. Katz, Ostrovsky, Yung Protocol (KOY) [7]

In this protocol, client A uses the password pwd_A to authenticate herself to server B. Server B stores a vector $PW_s = \langle pwd_A \rangle_{A \in Client}$ where Client is the set of all clients of the protocol. It is provided in [7] as an option to store g^{pwd_A} instead of pwd_A, and we consider this option for our analysis. The primes p, q, the generators g_1, g_2, h, c, d and hash function $\mathcal{H}$ (see Fig. 6) are the public values.

To execute the protocol, A first generates pk_A and sk_A, the one-time signature keys. Then client computes $\mathcal{A} = {g_1}^{r_1}, \mathcal{B} = {g_2}^{r_1}, \mathcal{C} = h^{r_1}{g_1}^{pwd_A}, \alpha = \mathcal{H}(Client_A|pk_A|\mathcal{A}|\mathcal{B}|\mathcal{C}), \mathcal{D} = (cd^\alpha)^{r_1}$ where r_1 is a random value and sends '$Client_A|pk_A|\mathcal{A}|\mathcal{B}|\mathcal{C}|\mathcal{D}$' to B at step 1. Server B then selects random values x_2, y_2, z_2, w_2, r_2 and computes $\alpha' = \mathcal{H}(Client_A|pk_A|\mathcal{A}|\mathcal{B}|\mathcal{C}), \mathcal{E} = {g_1}^{x_2}{g_2}^{y_2}h^{z_2}(cd^{\alpha'})^{w_2}, \mathcal{F} = {g_1}^{r_2}, \mathcal{G} = {g_2}^{r_2}, \mathcal{I} = h^{r_2}{g_1}^{pwd_c}, \beta = \mathcal{H}(Server|\mathcal{E}|\mathcal{F}|\mathcal{G}|\mathcal{I}), \mathcal{J} = cd^{\beta r_2}$. B sends A, $'(Server|\mathcal{E}|\mathcal{F}|\mathcal{G}|\mathcal{I})'$ at step 2. The client then generates randomly x_1, y_1, z_1, w_1 and computes $\beta' = \mathcal{H}(Server|\mathcal{E}|\mathcal{F}|\mathcal{G}|\mathcal{I})$ then $\mathcal{K} = {g_1}^{x_1}{g_2}^{y_1}h^{z_1}(cd^{\beta'})^{w_1}$. Finally, client computes Sig=$\text{Sign}_{sk_A}(\beta'|\mathcal{K})$ and sends $\mathcal{K}|Sig$ to B. Session key k then computed by client and server as $k = \mathcal{E}^{r_1}\mathcal{F}^{x_1}\mathcal{G}^{y_1}(\mathcal{I}'^{z_1})\mathcal{J}^{w_1} = \mathcal{K}^{r_2}\mathcal{A}^{x_2}\mathcal{B}^{y_2}(\mathcal{C}'^{z_2})\mathcal{D}^{w_2}$.

In the following analysis we consider μ as the entropy of pwd_A and equivalent cases are discussed together.

Case I: This is the case when attacker E gets idle-system access to system A or B and tries to compromise the password pwd_A. As the password is a memorized entity for A, the system of A provides no information. E with access to system of B gets the vector PW_s which stores g^{pwd_A}. E tries to guess for pwd as ${pwd_A}'$ and checks if $g^{pwd_A} = g^{{pwd_A}'}$ then $pwd_A = {pwd_A}'$. This requires 2^μ offline efforts.

Case II: In this case, E communicates with A or B, behaving as an insider or outsider. When E is an insider and B behaves as E, then with the stored value g^{pwd_A}, E can guess the password as pwd' and check if $g^{pwd_A} = g^{{pwd_A}'}$ which gives $pwd = pwd'$ with 2^μ offline efforts by guessing all possible values for password. When E acts as A, then E already knows the password pwd and there is nothing to compromise for her.

When E is an outsider and behaves as B, then he does not have the value g^{pwd_A}. E can guess pwd_A as ${pwd_A}'$. At step 1, the value $\mathcal{C} = h^{r_1}{g_1}^{pwd_A}$ uses pwd_A at system of A, which is used to compute session key k at system of B. Similarly at step 2 the value $\mathcal{I} = h^{r_2}{g_1}^{pwd_A}$ generated at system of B and used to compute k at system A. Incorrect guess for pwd_A will be detected at the verification steps for the value k. Therefore, E needs 2^μ online efforts to get the password pwd_A.

Cases III, X: To compromise password pwd_A from exchanged messages is the most challenging scenario, therefore cases III and X are analyzed together. It includes the scenario where E tries to impersonate B, by getting idle-system access to A or B with the aim of compromising pwd_A. As pwd_A is a memorized entity for A, therefore access to system A provides no information. E can get pwd_A with 2^μ online efforts and the guess for pwd_A is verified as correct when protocol executes successfully. With access to system of B, E gets the value g^{pwd_A}. E can guess pwd_A as ${pwd_A}'$ and if $g^{pwd_A} = g^{{pwd_A}'}$, then $pwd_A = {pwd_A}'$. This requires 2^μ offline efforts to get pwd_A.

Case IV: In this case, E intercepts the exchanged communications which provides the values $\langle Client_A|pk_A|\mathcal{A}|\mathcal{B}|\mathcal{C}|\mathcal{D}\rangle$, $\langle Server|\mathcal{E}|\mathcal{F}|\mathcal{G}|\mathcal{I}\rangle$ and $K|Sig$ where $r_1, r_2, x_2, y_2, z_2, w_2 \xleftarrow{\$} \mathbb{Z}_q$ and $\mathcal{A} = {g_1}^{r_1}, \mathcal{B} = {g_2}^{r_1}, \mathcal{C} = h^{r_1}{g_1}^{pwd_A}, \mathcal{D} = cd^{\alpha r_1}, \mathcal{E} = {g_1}^{x_2}{g_2}^{y_2}h^{z_2}(cd^{\alpha'})^{w_2}, \mathcal{F} = {g_1}^{r_2}, \mathcal{G} = {g_2}^{r_2}, \mathcal{I} = h^{r_2}{g_1}^{pwd_A}, \mathcal{J} = cd^{\beta r_2}$. Among the values, the password dependent value generated at system of A is $\mathcal{C} = h^{r_1}{g_1}^{pwd_A}$ where h, g_1 are generators, r_1 is a random value. To get pwd_A from $\mathcal{C}$, it is required to solve for the random value r_1. The difficulty solving r_1 is equivalent to solving discrete log problem over ${\mathbb{Z}_q}^*$. Hence, protocol is secure under this scenario.

Cases V, IX and XI: The idle-system access of system of A provides no information and idle-system access of system of B is a subset of the scenario of case V. Therefore, cases V, IX and XI follow the same analysis. The analysis covers the case where E intercepts communications between A and B with fullgame-access or midgame-access to system A or B. E with fullgame-access to system A or B gets all values involved in computations including, password, hence the protocol becomes insecure. E with midgame-access of system B gets the session key k and g^{pwd_A} at system B and $\langle Client_A|pk_A|\mathcal{A}|\mathcal{B}|\mathcal{C}|\mathcal{D}\rangle$, $\langle Server|\mathcal{E}|\mathcal{F}|\mathcal{G}|\mathcal{I}\rangle$ and $K|Sig$ from interception. From the value g^{pwd_A}, it is easy to get pwd_A with maximum 2^μ offline efforts by checking if $g^{pwd_A} = g^{{pwd_A}'}$. The midgame-access of system A follows the analysis of case IV, hence security is equivalent to solving DLP over ${\mathbb{Z}_q}^*$.

Case VI: In this case, with idle-system access of A or B, E communicates with A or B, behaving as an insider

or outsider. E tries to compromise the password pwd_A. When E is an insider and B acts as E, E has the value g^{pwd_A} from where E can get the password with 2^{μ} offline efforts by checking if $g^{pwd_A} = g^{pwd_A{}'}$. As A knows her password pwd_A, she does not need to behave as an insider to compromise it. In the outsider case, when E gets idle-system access of A (provides no information of pwd_A) and tries to communicate with A to impersonate as B, E needs 2^{μ} online efforts as discussed for case II above. When outsider E with idle-system access to system B tries to communicate with B, E gets the value g^{pwd_A}. E can guess pwd_A as $pwd_A{}'$ and check if $g^{pwd_A} = g^{pwd_A{}'}$ then $pwd_A = pwd_A{}'$. Therefore, it requires 2^{μ} offline efforts to get the correct password pwd_A.

Cases VII: For this case, E gets access to both the systems of A and B (not simultaneously). Hence, E always gains the value g^{pwd_A} on system B. With 2^{μ} offline efforts, E gets the password pwd_A by guessing $pwd_A{}'$ and verifying when $g^{pwd_A} = g^{pwd_A{}'}$.

Case VIII: This is the case where E with idle-system access of system A or B, intercepts the communication between A and B. E tries to compromise the password pwd_A. When E gets access to system A, the analysis is similar to case IV. When E gets access to system B along with the intercepted messages, E gets the values g^{pwd_A} from system B and values $\langle Client_A|pk_A|\mathcal{A}|\mathcal{B}|\mathcal{C}|\mathcal{D}\rangle$, $\langle Server|\mathcal{E}|\mathcal{F}|\mathcal{G}|\mathcal{I}\rangle$ and $K|Sig$ from interception. E can easily get pwd_A from g^{pwd_A} with 2^{μ} offline efforts.

C. Authenticated ID-based Key Exchange and Remote log-in with Simple Token and PIN (IdBP) [15]

This is a key exchange protocol based on an Id-based public key cryptosystem. It involves a trusted authority (TA) to generate the asymmetric key pairs for each user of the protocol. The TA first computes a random master secret s. When user A requests her key pairs, TA computes the hash of user A's identity Id_A (say) and outputs a point $\mathcal{A}$ on the elliptic curve. Then TA computes $s\mathcal{A}$ and sends user A the key pairs $(\mathcal{A}, s\mathcal{A})$ through secure channel. User A stores in her system the value $(s-\alpha)\mathcal{A}$ which is derived from $s\mathcal{A}$ as $s\mathcal{A} - \alpha\mathcal{A} = (s-\alpha)\mathcal{A}$ where α is the password of A. This s remains secret to A as the hardness to compute for the value of s is equivalent to solve the discrete log problem over elliptic curve which is assumed to be difficult. Similarly, user B gets key pairs $(\mathcal{B}, s\mathcal{B})$ and stores in his system the value $(s-\beta)\mathcal{B}$ protected with password β. To establish communication between users A and B, the protocol follows the Deffie-Hellman key exchange [16] approach implementing a bilinear mapping known as distorted Tate pairing [15] as shown in Fig. 7. Due to the properties of this bilinear mapping $\hat{e_r}(s\mathcal{A}, \mathcal{B}) = \hat{e_r}(s\mathcal{B}, \mathcal{A}) \neq 1$ [15] holds where $\mathcal{A}, \mathcal{B}, s$ are as defined above.

We assume μ_A and μ_B are the entropy of passwords α and β respectively. Equivalent cases are discussed together.

Case I: Attacker E with idle-system access of system A. Therefore E gets the value $(s-\alpha)\mathcal{A}$ and then tries to guess the password α. The value $(s-\alpha)\mathcal{A}$ is random, which includes the random secret s and unknown password α. From the value $(s-\alpha)\mathcal{A}$, solving discrete log gives the value $(s-\alpha)$ from which getting α is still difficult, as both s and α are unknown. Therefore, E can not verify the correctness of his guess for α by applying dictionary attack. Hence, protocol is secure under this scenario.

Case II: This is the case when E interacts with user A and E can behave as an insider or outsider. For the insider E, i.e., when B acts as E, E has the value $(s-\beta)\mathcal{B}$ and can generate sB from it by adding $\beta\mathcal{B}$. E tries to get α, the password of A. To generate session key k the protocol proceeds as follows. E computes $P_b = \hat{e_r}(s\mathcal{B}, \mathcal{A})^b$ where b is a random number generated by E and sends P_b to A while A generates $P_a = \hat{e_r}(s\mathcal{A}, \mathcal{B})^a$ where a is a random number generated by A and sends P_a to B. Then session key $k = (P_a)^b = (P_b)^a$ as the relation $\hat{e_r}(s\mathcal{A}, \mathcal{B}) = \hat{e_r}(s\mathcal{B}, \mathcal{A})$ holds for applied bilinear mapping. Therefore, from exchanged messages P_a, P_b, E receives no information related to α as P_a includes the value $s\mathcal{A}$ not $(s-\alpha)\mathcal{A}$. To get α E needs to have the value $(s-\alpha)\mathcal{A}$ which is not available. Hence, protocol is secure without having enough information. In the outsider's case, E tries to communicate with A, impersonating B and to compromise the password of B, i.e., β. However, being an outsider, E does not know the values $(s-\beta)\mathcal{B}$ (stored at system B) or $s\mathcal{B}$ to execute the attack. To know the master secret s the probability of success is $2^{-|s|}$ where $|s|$ is the length of s, 160 bits number over elliptic curve, but this effort does not provide β. Therefore, due to lack of information, E can not compromise the password of B.

Case III, X: This is the case when E gets the system of B and tries to impersonate B in communication with user A. Case X also provides E with intercepted messages from communications between A and B, but compromising the password of B from exchanged messages P_a and P_b is difficult to achieve (as shown for case II, insider). Therefore, cases III and X can equivalently be analyzed. From idle-access of system of B, E gets the value $(s-\beta)\mathcal{B}$. E aims to get the password β. He applies guesses for β as β', computes $(s-\beta)\mathcal{B} + \beta'\mathcal{B}$ to get $s\mathcal{B}$ from it and initiates the protocol. Wrong guess will not satisfy the equation $\hat{e_r}(s\mathcal{A}, \mathcal{B}) = \hat{e_r}(s\mathcal{B}, \mathcal{A})$ (if correct $s\mathcal{B}$ is not obtained) and will terminate the protocol at the time of verification for the possession of the session key k. Therefore, E tries all possible values of β to satisfy the above equation, which needs 2^{μ_B} online efforts to get correct β.

Cases IV: In this case, E intercepts the communication and gets the values P_a, P_b where $P_a = (s\mathcal{A}, \mathcal{B})^a$ and $P_b = (s\mathcal{B}, \mathcal{A})^b$, a and b are random secret numbers, generated by A and B respectively and $s\mathcal{A}, s\mathcal{B}$ are their respective secret keys. E wants to reveal the password α or β but,, it does not get corresponding values $(s-\alpha)\mathcal{A}$ or $(s-\beta)\mathcal{B}$ which are the only values to incorporate α or β of A or B. Therefore, the protocol is secure as E does not get enough information to compromise passwords.

Cases V: In this case, attacker E gets fullgame-access or midgame-access to system of B along with interception of communicated messages between A and B at the time of protocol execution. When attacker E gets **fullgame-access** of system of A or B, he gets all the values involved in protocol computations including password, hence, the protocol is insecure.

For **midgame-access** the assumption is that the computations after key establishment are available to E. Specifically E gets the session key k and tries to compromise β with

midgame-access to system of B.

Therefore, E gets the values $k = {P_a}^b$, $(s-\beta)\mathcal{B}$ from midgame-access, $P_a = \hat{e_r}((s-\alpha)\mathcal{A}, \mathcal{A})^a$, $P_b = \hat{e_r}((s-\beta)\mathcal{B}, \mathcal{B})^b$ from interception of messages

where P_a is generated by user A. From the above information the unknown values are a (random value) and password α generated by A, b (random value) and password β generated by B and s, the master secret (160 bits) generated by TA. To get β, E can only use the values P_b and $(s-\beta)\mathcal{B}$. Specifically, E can apply all choices for β as β' adding $\beta'\mathcal{B}$ to the value $(s-\beta)\mathcal{B}$. To verify the correctness of the guess, E then needs to know b from P_b to check if $P_b = \hat{e_r}((s-\beta)\mathcal{B} + \beta'\mathcal{B}, \mathcal{A})^b$. This gives $\beta = \beta'$. However, it requires to solve discrete log problem over elliptic curve to get b which is assumed to be a difficult problem.

Case VI: In this case, E being an insider or outsider, communicates with A after gaining idle-system access to system A. For the insider E when B acts as E, E gets the value $(s-\alpha)\mathcal{A}$ by accessing system of A where α is password of A. E has the value $(s-\beta)\mathcal{B}$ at his system from which he gets $s\mathcal{B}$ by adding $\beta\mathcal{B}$. E then computes the value $\hat{e_r}(s\mathcal{B}, \mathcal{A})$. To reveal the password α, E solves the equation if $\hat{e_r}(s\mathcal{B}, \mathcal{A}) = \hat{e_r}((s-\alpha)\mathcal{A} + \alpha'\mathcal{A}, \mathcal{B})$ then $\alpha = \alpha'$ (as $\hat{e_r}(s\mathcal{B}, \mathcal{A}) = \hat{e_r}(s\mathcal{A}, \mathcal{B})$ for distorted Tate pairing [15]). Therefore, E needs 2^{μ_A} offline effort to try all possible choices to get the correct α.

For outsider E, he accesses the system A and gets the value $(s-\alpha)\mathcal{A}$ but being outsider E does not have the value $(s-\beta)\mathcal{B}$ to impersonate B. E can use identity of B but secret of A to impersonate B. This is possible as the relation $\hat{e_r}(s\mathcal{A}, \mathcal{B}) = \hat{e_r}(s\mathcal{B}, \mathcal{A})$ holds for successful protocol execution. However, in this scenario, E can use $(s\mathcal{A}, \mathcal{B})$ instead of $(s\mathcal{B}, \mathcal{A})$. E sends the identity of B at step 1 and then, using A's data, impersonates B as explained below. At every attempt, E tries to guess the password as α' and his aim is to obtain $s\mathcal{A}$ from $(s-\alpha)\mathcal{A}$ by adding $\alpha\mathcal{A}'$. Let E get $s\mathcal{A}'$ from his guess to α'. Then he computes $P_b = \hat{e_r}(s\mathcal{A}', \mathcal{B})^b$, where b randomly generated and sends P_b to A. Similarly A sends to B, $P_a = \hat{e_r}(s\mathcal{A}, \mathcal{B})^a$ where a randomly generated by A. Then computed session key k should satisfy $k = (P_b)^a = (P_a)^b$ which is only possible when $\hat{e_r}(s\mathcal{A}, \mathcal{B}) = \hat{e_r}(s'\mathcal{A}, \mathcal{B})$, i.e., when E gets correct guess for α. Therefore, it needs 2^{μ_A} online efforts.

Case VII: This is the case when E accesses both systems A and B (not simultaneously). With access to both the systems, attacker E gets the values $(s-\alpha)\mathcal{A}$ and $(s-\beta)\mathcal{B}$ where α and β are secrets of A and B respectively. E then solves if $\hat{e_r}((s-\alpha)\mathcal{A} + \alpha'\mathcal{A}, \mathcal{B}) = \hat{e_r}((s-\beta)\mathcal{B} + \beta'\mathcal{B}, \mathcal{A})$ to get $\alpha = \alpha'$ and $\beta = \beta'$ which needs total $2^{\mu_A} \times 2^{\mu_B} = 2^{\mu_A + \mu_B}$ efforts by applying dictionary attack for the guesses of α and β respectively.

Cases VIII: For this case, E accesses the idle system of A and also intercepts the communication between A and B. Therefore, E gets the value $(s-\alpha)\mathcal{A}$ by accessing system of A, and the values $P_a = (s\mathcal{A}, \mathcal{B})^a, P_b = (s-\beta)\mathcal{B}$ where a, b random numbers generated by A and B respectively, through interception. To verify the guess for secret α, E needs to know the random value a which computes the value $P_a = \hat{e_r}((s-\alpha)\mathcal{A} + \alpha\mathcal{A}, \mathcal{B})^a$ (see Fig. 7). This requires solving a discrete log problem (dlp) over elliptic curve which is supposed to be difficult.

Cases IX: In this case, E gets idle-access to system of A and fullgame or midgame-access of system of B with interception of communication between A and B. When attacker E gets fullgame-access of system of A or B, he gets all values involved in protocol computations including password, hence the protocol is insecure.

With midgame-access E gets the following information. $(s-\alpha)\mathcal{A}$ from system of A, P_a, P_b from interception and $k = (P_a)^b$, $(s-\beta)\mathcal{B}$ from midgame-access of system of B. The complexity to compromise β from the values $(s-\beta)\mathcal{B}$ and P_b is disscussed for case V (security of solving DLP). Therefore, E tries the approach as discussed for case VII. With access to both the systems, attacker E gets the values $(s-\alpha)\mathcal{A}$ and $(s-\beta)\mathcal{B}$ where α and β are secrets of A and B respectively. E then solves if $\hat{e_r}(((s-\alpha)\mathcal{A} + \alpha'\mathcal{A}, \mathcal{B}) = \hat{e_r}((s-\beta)\mathcal{B} + \beta'\mathcal{B}, \mathcal{A})$ to get $\alpha = \alpha'$ and $\beta = \beta'$ which needs total $2^{\mu_A} \times 2^{\mu_B} = 2^{\mu_A + \mu_B}$ efforts by applying dictionary attack for the guesses of α and β respectively.

Cases XI: In this case, attacker E gets fullgame-access or midgame-access to system of B along with interception of exchanged messages between A and B during protocol execution. E then tries to impersonate B in further communication with A. When attacker E gets fullgame-access to system A or B, he gets all the values involved in protocol computations including, password, hence the protocol is insecure.

For midgame-access the assumption is that the computations after key establishment are available to E. E gets the values P_a, P_b from interception, values $(s-\beta)\mathcal{B}$, $k = (P_a)^b$ from midgame-access of system of B and again value $(s-\beta)\mathcal{B}$ while trying to impersonate as B with system of B. As explained for case V, with values P_b and $(s-\beta)\mathcal{B}$ the security is equivalent to solve discrete log problem which is assumed to be hard. As explained for case III to impersonate B, E tries to apply guesses for β as β' to the value of $(s-\beta)\mathcal{B} + \beta'\mathcal{B}$, which should provide $s\mathcal{B}$. With an incorrect guess, E can initiate the protocol which will be detectable at the time of verification of established session key k. Specifically, wrong guess does not satisfy the equation $\hat{e_r}(s\mathcal{A}, \mathcal{B}) = \hat{e_r}(s\mathcal{B}, \mathcal{A})$ [15].

Therefore, E tries all possible values of β to satisfy the above equation, which needs 2^{μ_B} online efforts to achieve success. This online trial is not very practical as any such attempt is easily detectable in practice. So both the complexities discussed above are difficult to achieve.

V. Suggested Improvements

In this section we provide some suggestions which may improve the security of the 5 PAKE protocols discussed in section III. There may exist several possible approaches to improve the security to prevent dictionary attack. We propose the following changes which may overcome the general security issues discussed in section IV and provide atleast the security of the actual protocol. Designing a secure PAKE is always challenging, and any design modification may introduce some unknown threats. Therefore, we include possible improvements for individual steps which need modification. For every PAKE protocol, one of the necessary requirements is that there should not be the possibility of a

TABLE III
SUMMARY OF SECURITY ISSUES & SUGGESTED IMPROVEMENTS
MITM: MAN-IN-THE-MIDDLE, ZKP: ZERO-KNOWLEDGE PROOF, KE: KEY EXCHANGE.

Protocol	Uniqueness	Security Issues	Suggested Improvements
EKE [2]	- Seminal work - Simplicity of design	- Forward Secrecy - Reply & MitM	- Use of nonce for freshness, Mutual authentication - k to be chosen by both participants
SPEKE [1]	- First commercial implementation - Simple and efficient	- Forward Secrecy - Replay, MitM, Key malleability	- Use of some session specific randomness - careful selection of group parameters
SRP [8]	- Supported by SSL, prevents phishing - Mutual authentication of participants - ZKP: Authentication without revealing password	- Offline dictionary attack - Impersonation & Replay attack	- Use of some session specific randomness - Secure storage of verifier - Change in basic design approach
KOY [7]	- First provably secure under standard model - Mutual authentication of participants	- Offline dictionary attack - Forward Secrecy	- Secure storage of verifier - Change in basic design approach for efficiency
IdBP [15]	- Not a PAKE but Authenticated ID-Based KE - Two-Factor Authentication	- Insider threat if Token/PIN leaked - Forward Secrecy, Replay & MitM	- Secure storage of verifier - Change of the considered Id-based primitive

known plain text attack by knowing the predictable message of the protocol after successful key establishment. This may lead to an easy compromise of the session key and hence the compromise of the user privacy. The concept of All-Or-Nothing encryption (for block cipher) introduced by Rivest in [22] can be an approach to resist such attack, but it is efficient when a message is long (in terms of the number of blocks) and introduces computational overhead at the time of decryption. A good design should be able to balance the trade-off in an efficient way.

- **Encrypted Key Exchange Protocol [2]** Following changes may overcome the general security issues discussed in Section IV-A. See Fig. 3 for an overview of the protocol. The protocol should incorporate a nonce to define the session id at step 1 to avoid replay attack (specially for the case of insider). The value k should be generated by both the parties (also discussed in [2]). At step 5, the placement of challenge$_B$ and challenge$_A$ should be swapped as it provides verifiable text [8]. There should be some independent randomness (from the password) in the computation of the session key k in step 2 so that k remains secret even if the password is compromised.
- **Simple Password Exponential Key Exchange Protocol (SPEKE) [1]** The fully constraint SPEKE defines the function $f = (pwd)^2 \bmod p$ where p is a safe prime. This leads to the attack of password guessing forming exponential equivalence classes as explained in [17]. To overcome this problem, f should be redefined. There exist numerous analysis and suggestions, as in [23] for safe implementation of the Deffie-Hellman approach which may be considered while designing f. The key malleability problem can be restricted by the use of a similar concept from KOY [7] which forces some validation check at both participant's side. Without the knowledge of password, successful authentication with two parallel sessions as shown in [18] can be restricted by breaking the symmetry in the exchange messages by associating some session specific randomness.
- **Secure Remote Password Protocol (SRP) [8]** The verifier corresponding client's password makes the dictionary attack feasible to reveal the secret password. Other than the storage of verifiable data on the server, client never transfers password related data during protocol execution. Attackers with internal access to systems A and B at the time of protocol execution would compromise the secret, which is quite challenging to restrict. Therefore, further changes to improve the security of this protocol may change the basic design approach.
- **Katz, Ostrovsky, Yung Protocol (KOY) [7]** The security of this protocol is bounded by dictionary attacks as claimed by the authors [7]. The protocol does not provide forward secrecy and both the weakness and suggested modifications to incorporate it are provided in [20]. Apart from that, the main concern is the practicality of the design, which requires thorough research.
- **Authenticated ID-based key Exchange and remote log-in with simple token and PIN [15]** The protocol uses Id-based encryption following bilinear mapping named distorted Tate pairing. This bilinear mapping has the property that if user A obtains the stored secret (secret value of B that combines the secret key and password of B) of user B, then A can easily obtain the password of B. This shows an insider attack scenario. To overcome this attack, the protocol should use an Id-based encryption scheme which avoids this inherent weakness of the presented bilinear mapping [15].

Table III summaries the results of the reported analysis and our analysis with suggested improvements for each of the considered protocols. In this work, we consider PAKE protocols, including variations in the design approach. Our analysis aligns with the existing analysis which proves the efficacy of the proposed analysis tool. A good overview of PAKE is also avaialble at [24].

VI. CONCLUDING REMARKS

In this work, we evaluate the security of 5 PAKE protocols against our proposed attack scenarios. We have found that dictionary attack is the best attack to compromise a secret password in most cases. Other reported weaknesses of the protocols are also summaried in Table III. A natural research extension to this work would be to apply the ideas discussed in this paper to design an efficient PAKE in terms of security and feasibility and also to work for an encryption technique that efficiently resists known plain-text attack. Future analysis can also include recent asymmetric PAKE protocols (aPAKE) such as OPAQUE [14] which can be implemented in TLS [13].

REFERENCES

[1] D. P. Jablon, "Strong password-only authenticated key exchange," *Computer Communication Review*, vol. 26, no. 5, pp. 5–26, 1996.

[2] S. M. Bellovin and M. Merritt, "Encrypted key exchange: password-based protocols secure against dictionary attacks," in *Proc. IEEE Computer Society Symposium on Research in Security and Privacy, Oakland, CA, USA*, 4-6, May 1992.

[3] S. M. Bellovin and M. Merritt, "Augmented encrypted key exchange: A password-based protocol secure against dictionary attacks and password file compromise," in *Proc. 1st ACM Conf. Computer and Communications Security (CCS '93), Fairfax, VA, USA*, pp. 244–250, Nov. 1993.

[4] M. Steiner, G. Tsudik, and M. Waidner, "Refinement and extension of encrypted key exchange," *Operating Systems Review*, vol. 29, no. 3, 1995.

[5] M. Bellare, D. Pointcheval, and P. Rogaway, "Authenticated key exchange secure against dictionary attacks," in *Advances in Cryptology – EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium*, pp. 139–155, May 2000.

[6] R. Canetti, O. Goldreich, and S. Halevi, "The random oracle methodology, revisited (preliminary version)," in *Proc. 30th Annual ACM Symp. Theory of Computing (STOC '98), Dallas, TX, USA 23-26 May 1998*, pp. 209–218, 1998.

[7] J. Katz, R. Ostrovsky, and M. Yung, "Efficient password-authenticated key exchange using human memorable passwords," in *Advances in Cryptology – EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding* pp. 475–494, 2001.

[8] T. D. Wu, "The secure remote password protocol," in *Proceedings of the Network and Distributed System Security Symposium, NDSS 1998, San Diego, California, USA* 1998.

[9] D. Chang and M. Yung, "Midgame attacks (and their consequences)," *Crypto 2012 Rump Session, Available: http://crypto.2012.rump.cr.yp.to/008b781ca9928f2c0d20b91f768047fc.pdf.*

[10] M. Bellare and P. Rogaway, "Provably secure session key distribution: the three party case," in *Proc. 27th ACM Symp. Theory of Computing 29 May-1 June 1995, Las Vegas, NV, USA*, pp. 57–66, 1995.

[11] F. Hao and S. F. Shahandashti, "The SPEKE protocol revisited," in *Chen, L., Mitchell, C. (eds) Security Standardisation Research (SSR 2014), Lecture Notes in Computer Science, vol. 8893, Springer, Cham, https://doi.org/10.1007/978-3-319-14054-4_2*.

[12] "Secure Remote Password protocol," *Wikipedia*. [Online]. Available: https://en.wikipedia.org/wiki/Secure_Remote_Password_protocol. [Accessed: May 14, 2024].

[13] J. Hesse, S. Jarecki, H. Krawczyk, and C. Wood, "Password-authenticated TLS via OPAQUE and post-handshake authentication," in *Advances in Cryptology – EUROCRYPT 2023, C. Hazay and M. Stam, Eds., Lecture Notes in Computer Science, vol. 14008, Springer, Cham, https://doi.org/10.1007/978-3-031-30589-4_4*.

[14] S. Jarecki, H. Krawczyk, and J. Xu, "OPAQUE: An asymmetric PAKE protocol secure against pre-computation attacks," in *Advances in Cryptology – EUROCRYPT 2018, J. Nielsen and V. Rijmen, Eds., Lecture Notes in Computer Science, vol. 10822, pp 456-486, 2018, Springer, Cham, https://doi.org/10.1007/978-3-319-78372-7_15*.

[15] M. Scott, "Authenticated ID-based key exchange and remote log-in with simple token and PIN number," *IACR Cryptology ePrint Archive*, vol. 2002, no. 164, 2002.

[16] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[17] M. Zhang, "Analysis of the SPEKE password-authenticated key exchange protocol," *IEEE Communications Letters*, vol. 8, no. 1, pp. 63–65, 2004.

[18] F. Hao and S. F. Shahandashti, "The SPEKE protocol revisited," *IACR Cryptology ePrint Archive*, vol. 2014, no. 585, 2014.

[19] R. Cramer and V. Shoup, "A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack," in *Advances in Cryptology – CRYPTO '98, Proceedings of 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998*, Santa Barbara, CA, USA, Aug. 1998, pp. 13–25, 1998.

[20] J. Katz, R. Ostrovsky, and M. Yung, "Forward secrecy in password-only key exchange protocols," in *Security in Communication Networks, Third International Conference, SCN 2002, Amalfi, Italy, September 11-13, 2002. Revised Papers*, pp. 29–44, 2002.

[21] D. Boneh and M. K. Franklin, "Identity-based encryption from the Weil pairing," *SIAM J. Comput.*, vol. 32, no. 3, pp. 586–615, 2003.

[22] R. L. Rivest, "All-or-nothing encryption and the package transform," in *Fast Software Encryption, 4th International Workshop, FSE '97, Haifa, Israel, January 20-22, 1997, Proceedings*, pp. 210–218, 1997.

[23] J. F. Raymond and A. Stiglic, "Security issues in the Diffie-Hellman key agreement protocol, Tech. Rep., Zeroknowledge Inc., Sep. 2000".

[24] F. Hao and P. C. van Oorschot, "SoK: Password-authenticated key exchange – theory, practice, standardization and real-world lessons," in *ASIA CCS '22: Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security, pp. 697–711, 2022, doi: https://doi.org/10.1145/3488932.35232*.