Efficient Cooperative Key Exchange Protocol

Priyanka Madhiraju & G Deepa

Asst. Professor, Computer Science & Engineering, Jagruti Institute of Engg. & Tech. Chintapalliguda(V),IBP(M),RR Dist-501510. e-mail: priyankaraomadhiraju@gmail.com

M.Tech Student , Department of Computer Science & Engineering Jagruti Institute of Engg. & Tech. Chintapalliguda(V),IBP(M),RR Dist-501510. e-mail: deepa.gunna@gmail.com

Abstract–

Password-authenticated essential exchange (PAKE) is when a client along with a server, whom share a password, authenticate the other and meanwhile begin a cryptographic essential by exchange of announcements. In that setting, all the particular passwords essential to authenticate consumers are stored in a single server. When the server is compromised, because of, for instance, hacking as well as insider assault, passwords stored inside the server are all disclosed. Within this paper, we look at a scenario wherever two computers cooperate to help authenticate litigant and in the event that one server is compromised, the assailant still are unable to pretend for being the client while using the information from the compromised server. Current remedies for two-server PAKE tend to be either symmetric inside the sense that will two peer servers equally promote the authentication or asymmetric inside the sense any particular one server authenticates the client through another server. This kind of paper offers a symmetric answer for two-server PAKE, the spot that the client could establish distinct cryptographic keys while using the two computers, respectively.

***Keywords*— Wireless Networks, Authentication, Key Exchange**

I. INTRODUCTION

Today, passwords may be used by people during a log throughout process which controls having access to protected computer's, mobile phones, cable TELEVISION SET decoders, automated teller machines and so forth. A laptop or computer user might have to have passwords for many people purposes: logging into computer accounts, retrieving e-mail coming from servers, getting at programs, data source, networks, websites, and even reading the morning paper online. Earlier password-based authentication systems transmitted the cryptographic hash on the password spanning a public channel helping to make the hash value accessible to a attacker. When that is done, in fact it is very typical, the attacker perform offline, rapidly assessment possible passwords against the true password's hash value. Studies get consistently shown which a large portion of user-chosen security passwords are readily guessed automatically. For example, according to help Bruce Schneier, examining data from the 2006 phishing attack, 55 % of passwords

would be crackable throughout 8 hours utilizing a commercially available Password Recuperation Toolkit effective at testing 2 hundred, 000 security passwords per 2nd in 2006.

Recent exploration advances throughout password-based authentication get allowed a customer and the server mutually to help authenticate with a password in addition to meanwhile to determine a cryptographic critical for protected communications after authentication. Generally, current answers for password based authentication stick to two products.

The first model, termed PKI-based model, assumes how the client keeps the server's public key as well as share the password using the server. Within this setting, your customer can mail the password to the server by public critical encryption. Gong et 's. were the first to present this kind of authentication methodologies with heuristic proof to offline dictionary violence, and Halevi in addition to Krawczyk were the first to supply formal classifications and thorough proofs associated with security for PKI-based model.

The 2nd model is called password-only model. Bellovin in addition to Merritt were the first to contemplate authentication according to password solely, and introduced a couple of so-called "encrypted critical exchange" methodologies, where the password can be used as the secret critical to encrypt randomly numbers for key swap purpose. Formal types of security for that password-only authentication ended up first given independently by Bellare et 's. and Boyko et 's.

Katz et 's. were the first to offer a password-only authentication protocol that's both useful and provably protected under standard cryptographic presumption. Based on the identity-based encryption approach, Yi et 's. suggested an identity-based model where the client should remember the password only as you move the server keeps the password as well as private keys related to its identity. In this kind of setting, the purchaser can encrypt the password based on the identity on the server. This model is relating to the PKI-based along with the password solely models.

Typical methodologies for password-based authentication assume a single server stores every one of the passwords required to authenticate consumers. If the server can be compromised, caused by, for example, hacking, or maybe installing the "Trojan horse, " or maybe insider attack, user security passwords stored within the server are usually disclosed. To treat this matter, two-server password-based authentication methodologies were unveiled, where a pair of servers cooperate to authenticate a customer based on password and if one server can be compromised, the assailant still are not able to pretend to get the client using the information on the compromised server.

Current answers for two-server PAKE are usually either symmetric within the sense which two peer servers equally contribute to the authentication, or asymmetric within the sense that you server authenticates your customer through another work. A symmetric a pair of server PAKE paper, for example, Katz et 's. 's paper, can function in parallel in addition to establishes secret session keys relating to the client in addition to two servers, respectively. In case one of several two servers shuts down due

to the denial-of-service attack, another server can continue to provide products and services to authenticated consumers. In conditions of parallel working out and trusted service, a symmetric paper is better than an asymmetric paper. So much, only Katz et 's. ’s two-server PAKE protocol has been symmetric. But their protocol just isn't efficient for practical use. An asymmetric two-server PAKE paper runs throughout series and only the front-end server along with the client should establish the secret program key. Recent asymmetric methodologies, for example, Yang et 's. ’s paper and Jin et 's. ’s paper, need a pair of servers to switch messages for several times throughout series. These asymmetric types are less efficient compared to a symmetric design that enables two servers to compute in parallel.

II. Previous Work

Wireless Humans typically choose weak, low-entropy passwords, while standard authentication protocols assume the use of *cryptographic* (i.e., high-entropy) secrets. Unfortunately, protocols designed and proven secure in the latter setting are generally insecure in the former context because these protocols are not resistant to *online dictionary attacks* in which an eavesdropping adversary derives information about the password from observed transcripts of login sessions. In recent years, much attention has focused on designing password-based authenticated key-exchange protocols resistant to such attacks. We remark that *on-line dictionary attacks* in which an adversary simply attempts to login repeatedly, trying each possible password cannot be prevented by cryptographic means but can be dealt with using other methods outside the scope of this work.

Means of protecting against online dictionary attacks in a single-server setting were first suggested by Gong et al. in a hybrid, PKI-based model where users are required to store the server's public key in addition to a password. Bellovin and Merritt were the first to suggest protocols for *password-only authenticated key exchange* (PAKE), where users are required to store *only* a short password. These initial works were relatively informal and did not provide definitions or proofs of security. Subsequently, formal definitions and provably secure protocols for the hybrid model were given, followed by models for the password-only setting and associated protocols with proofs of security in the random oracle/ideal cipher models or in the standard model. The protocols assume some public information which is available to all parties. Since this information can be hard-coded into implementations of the protocol, clients do not need to memorize or store any high-entropy, cryptographic information as they are required to do in the PKI-based setting.

Passive Adversaries

We assume an adversary who corrupts some servers at the outset of the protocol, such that for any client C at most one of the servers associated with C is corrupted. Our definition does not require the adversary to corrupt one server associated with each client; thus, our definition encompasses security in the case when neither server associated with some client is corrupted. In the case of a passive adversary, a corrupted server continues to operate according to the protocol but the adversary may monitor the corrupted server's internal state.

There are two types of communication: between clients and servers, and between

servers. We give the adversary full control over the client-server communication; thus, the adversary can eaves- drop on this communication, send messages of its choice to servers or clients, or tamper with, delay, refuse to deliver, etc. any messages sent between servers and clients. Server-server communication is assumed to be done over a secure channel this can be realized via standard use of private-key cryptography, since the servers can store long-term, shared keys; thus, the adversary is unable to eavesdrop on the communication between two uncorrupted servers, and communication between *any* two servers is not under adversarial control.

Active Adversaries

The only difference in the active case is that the adversary may now cause any corrupted servers to deviate in an arbitrary way from the actions prescribed by the protocol. Thus, if a server is corrupted the adversary controls all messages sent from this server to any other servers. The adversary can also control messages sent from this server to any clients; note, however, that even

a passive adversary has this ability since it controls the communication channel between servers and clients. Because of this change, we no longer use a Send oracle to model sending messages to a corrupted server, but we formally introduce a new Send oracle to model communication from a corrupted server to a non-corrupted server. We also charge the adversary with an additional on-line attack for using the latter. As in the passive case, however, we continue to assume that the adversary cannot eavesdrop on or control communication between two non-corrupted servers. [1]

Boneh and Franklin were the first to define chosen ciphertext security for IBE under chosen identity attack. In this section, we put forward a new model of security for ID-based group PAKE, on the basis of definitions given by Bresson et al., Abdalla et al. and Boneh et al. Participants, Initialization and Passwords. An ID-based group PAKE protocol involves three kinds of participants: (1) A set of clients (denoted as Client); (2) A set of servers (denoted as Server), which behave in an honest but curious manner; (3) A Private Key Generator (PKG), which generates public parameters and corresponding private keys for servers, and behaves in an honest but curious manner as well. We assume that Client Server Pair is the set of pairs of the client and the server, who share a password.

In the real world, a protocol determines how users behave in response to input from their environments. In the formal model, these inputs are provided by the adversary. Each user is assumed to be able to execute the protocol multiple times (possibly concurrently) with different partners. This is modeled by allowing each user to have unlimited number of instances with which to execute the protocol. We denote instance i of user U as Ui. A given instance may be used only once. The adversary is given oracle access to these different instances. Furthermore, each instance maintains (local) state which is updated during the course of the experiment. In particular, each instance Ui has associated with it the following variables, initialized as NULL or FALSE (as appropriate) during the initialization phase. [2]

Identity-Based Cryptography

The concept of Identity-Based Cryptography, wherein the identity is used as a public key, and

the corresponding private key is derived from the identity and a master secret, owes its origins to Shamir. The main attraction of this approach is the avoidance of the expensive public-key infrastructure (PKI), needed to certify the binding of a public key to an identity.

The idea of identity-based cryptography remained unrealized until 2001, when Boneh and Franklin proposed an implementation based on Weil pairing, soon followed by Joux. These seminal works have catalyzed new areas of exploration, and new public key encryption systems that avoid the use of large-scale Certificate Authorities (CA) (and related PKI) have been proposed.

The riddance of the CAs does not come for free. In ID-based setting, players' private keys cannot be generated by the players themselves, since the generation involves a global master secret. Instead, private keys are computed by a Key Management Service (KMS), and transmitted to players upon identity verification. Thus, in contrast with CAs in the traditional public-key setting, where players can generate their keys themselves, the KMS knows all the keys in the ecosystem, and must be trusted not to abuse this knowledge. In essense, the KMS must be treated as a key escrow entity. [3]

Password-based key exchange:

Password-based key exchange protocols assume a more realistic scenario in which secret keys are not uniformly distributed over a large space, but rather chosen from a small set of possible values (a four-digit pin, for example). They also seem more convenient since human-memorable passwords are simpler to use than, for example, having additional cryptographic devices capable of storing high-entropy secret keys. The vast majority of protocols found in practice do not account, however, for such scenario and are often subject to so-called dictionary attacks. Dictionary attacks are attacks in which an adversary tries to break the security of a scheme by a brute-force method, in which it tries all possible combinations of secret keys in a given small set of values (i.e., the dictionary). Even though these attacks are not very effective in the case of high-entropy keys, they can be very damaging when the secret key is a password since the attacker has a non-negligible chance of winning.

To address this problem, several protocols have been designed to be secure even when the secret key is a password. The goal of these protocols is to restrict the adversary's success to on-line guessing attacks only. In these attacks, the adversary must be present and interact with the system in order to be able to verify whether its guess is correct. The security in these systems usually relies on a policy of invalidating or blocking the use of a password if a certain number of failed attempts have occurred.

First, the standard notion of security for MACs does not imply security against related key attacks. Hence, new and stronger security notions are required. Second, such new security notions may have to consider adversaries which are given access to a related-key tag-generation oracle. These are oracles that are capable of generating tags on messages under related keys, where the related key function is also passed as a parameter. However, it is not clear whether such MACs can even be built. Such security notion, for instance, may completely rule out the possibility of using block-cipher-based MAC algorithms since similar security requirements in the context of block ciphers are

known to be impossible to achieve. Perhaps, hash-based MAC algorithms may be able to meet these goals, but that does not seem likely without resorting to random oracles, which would defeat the purpose of using MACs in the first place. [4]

The seminal work in this area is the Encrypted Key Exchange (EKE) protocol proposed by Bellovin and Merritt. EKE is a classical Diffe-Hellman key exchange wherein either or both flows are encrypted using the password as a common symmetric key. The encryption primitive can be instantiated via either a password-keyed symmetric cipher or a mask generation function computed as the product of the message with the hash of a password. Bellare et al. sketched a security proof for the flows at the core of the EKE protocol, and specified a EKE-structure called the AuthA protocol. Boyko et al. proposed very similar EKE-structures (called the PAK suite) and proved them secure in Shoup's simulation model. The PPK protocol in the PAK suite is similar to our Two-Mask Encrypted Key Exchange protocol; however, arguments in favor of forward-secrecy under the computational Diffe-Hellman (CDH) assumption do not give many guarantees on its use in practice. The KOY protocol is also proved to be forward-secure but it is not efficient enough to be used in practice.

The PAK suite is in the process of being standardization by the IEEE P1363.2 Standard working group. Server machines store images of the password under a one-way function instead of a plaintext password when the \augmented" versions of the PAK suite are used. "Augmented" EKE-like protocols indeed limit the damage due to the corruption of a server machine, but do not protect against attacks replaying captured users' passwords. On the other hand, One-Time Password (OTP) systems protect against the latter kind of attacks but provide neither privacy of transmitted data nor protection against active attacks such as session hijacking. The present paper designs and develops a cryptographic protocol for one-time \augmented" password-authenticated key exchange. [5]

Another application for IBE systems is delegation of decryption capabilities. We give two example applications. In both applications the user Bob plays the role of the PKG. Bob runs the setup algorithm to generate his own IBE system parameters params and his own master-key. Here we view params as Bob's public key. Bob obtains a certificate from a CA for his public key params. When Alice wishes to send mail to Bob she first obtains Bob's public key params from Bob's public key certificate. Note that Bob is the only one who knows his master-key and hence there is no key-escrow with this setup.

Suppose Alice encrypts mail to Bob using the current date as the IBE encryption key (she uses Bob's params as the IBE system parameters). Since Bob has the master-key he can extract the private key corresponding to this IBE encryption key and then decrypt the message. Now, suppose Bob goes on a trip for seven days. Normally, Bob would put his private key on his laptop. If the laptop is stolen the private key is compromised. When using the IBE system Bob could simply install on his laptop the seven private keys corresponding to the seven days of the trip. If the laptop is stolen, only the private keys for those seven days are compromised. The master- key is unharmed. This is analogous to the delegation scenario for signature schemes considered by Goldreich et al.

Suppose Alice encrypts mail to Bob using the subject line as the IBE encryption key. Bob can decrypt mail using his master-key. Now, suppose Bob has several assistants each responsible for a different task (e.g. one is purchasing, another is human-resources, etc.). Bob gives one private key to each of his assistants corresponding to the assistant's responsibility. Each assistant can then decrypt messages whose subject line falls within its responsibilities, but it cannot decrypt messages intended for other assistants. Note that Alice only obtains a single public key from Bob (params), and she uses that public key to send mail with any subject line of her choice. The mail can only be read by the assistant responsible for that subject. More generally, IBE can simplify security systems that manage a large number of public keys. Rather than storing a big database of public keys the system can either derive these public keys from usernames, or simply use the integers as distinct public keys. [6]

The design of most SPAKA protocols overlooks a fundamental problem: The server itself represents a serious vulnerability. As SPAKA protocols require the verifying server to have clear text access to user passwords (or to derivative material), compromise of the server leads potentially to exposure of the full database of passwords. While many SPAKA protocols store passwords in combination with salt or in some exponentiated form, an attacker who compromises the server still has the possibility of mounting on-line dictionary attacks. Additionally, these systems offer no resistance to server corruption. An attacker that gains control of the authenticating server can spoof successful login attempts.

To address this problem, Ford and Kaliski introduced a system in which passwords are effectively protected through distribution of trust across multiple servers. Mackenzie, Shrimpton, and Jakobsson extended this system, leading to more complex protocols, but with rigorous security reductions in a broadly inclusive attack model. Our work in this paper may be regarded as a complement, rather than a successor to the work of these authors. We propose a rather different technical approach, and also achieve some special benefits in our constructions, such as a substantially reduced computational load on the client. At the same time, we consider a different, and in our view more pragmatic security model than that of other distributed SPAKA protocols.

The scheme of Ford and Kaliski reduces server vulnerability to password leakage by means of a mechanism called password hardening. In their system, a client parlays a weak password into a strong one through interaction with one or multiple hardening servers, each one of which blindly transforms the password using a server secret. Ford and Kaliski describe several ways of doing this. Roughly speaking, the client in their protocol obtains what may be regarded as a blind function evaluation of its password P from each hardening server Si. (The function in question is based on a secret unique to each server and user account.) The client combines the set of shares into a single secret a strong key that the user may then use to decrypt credentials, authenticate herself, etc. Given an appropriate choice of blind function evaluation scheme, servers in this protocol may learn no information, in an information theoretic sense, about the password P. An additional element of the protocol involves the user authenticating by

means to each of the servers, thereby proving successful hardening. The harderened password is then employed to decrypt downloaded credentials or authenticate to other servers. We note that the Ford-Kaliski system is designed for credential download, and not password recovery; our system is specially designed to support both. Another important distinction is that in the Ford-Kaliski system, the client interacts with both servers directly. As we describe, an important feature of our proposed system is the configuration of one server in the back-end, yielding stronger privacy protection for users. [7]

The way we have defined dake, the client ends up with *k* shared keys, while the goal of a standard authenticated key exchange is for the client to end up with a single key shared with a server it wishes to communicate with. There are alternative definitions that would more closely mimic this. However, we feel our definition is more general, since once the client can securely communicate with *k* servers, it can use this not only to enable secure communication with any other desired server, but to enable any desired cryptographic functionality. For instance, a secure dake protocol allows for secure downloadable credentials, by, e.g., having the servers store an encrypted credentials file with a decryption key stored using a threshold scheme among them, and then having each send a partial decryption of the credentials file to the client, encrypted with the session key it shares with the client. (To deal with compromised servers, one could require each server to also send a zero-knowledge proof that it performed its partial decryption correctly.) Note that the credentials are secure in a threshold sense: fewer than the given threshold of servers are unable to obtain the credentials. Once the client has securely downloaded its credentials (for instance, it could download its certified public key and the associated private key), it can use these credentials to set up secure communication with another server, or perhaps sign messages, or perform other cryptographic operations. [8]

Proposed System

A. *Initialization*

The two peer servers S1 and S2 jointly choose a cyclic group G of large prime order q with a generator g1 and a secure hash function H(), which maps a message of arbitrary length into an l-bit integer. Next, S1 randomly chooses an integer s1 from Zq and S2 randomly chooses an integer s2 from Zq , and S1 and S2 exchange g1s1 and g1s2. After that, S1 and S2 jointly publish public system parameters G,q,g1,g2,H. In most of existing two-server PAKE protocols, it is assumed or implied that the discrete logarithm of g2 to the base g1 is unknown to anyone. Otherwise, their protocols are insecure. Our initialization can ensure that nobody is able to know the discrete logarithm of g2 to the base g1 unless the two servers collude. It is well known that the discrete logarithm problem is hard, and our model assumes that the two servers never collude.

Registration

Prior to authentication, each client C is required to register both S1 and S2 through different secure channels. First of all, the client C generates decryption and encryption key pairs (xi,yi) where yi=g power xi for the server Si using the public parameters published by the two servers. Next, the client C chooses a password pwC and encrypts the password using the encryption key, according to ElGamal encryption. Then, the client C randomly chooses b1 from Zq. At last, the client C delivers the password authentication

information Auth1to S1 through a secure channel, and the password authentication information Auth2 to S2 through another secure channel. After that, the client C remembers the password pwC only. The two secure channels are necessary for two server PAKE protocols, where a password is split into two parts, which are securely distributed to the two servers, respectively, during registration. Although we refer to the concept of public key cryptosystem, the encryption key of one server should be unknown to another server and the client needs to remember a password only after registration.

B. *Key Exchange*

In this module the two servers S1 and S2 have received the password authentication information of a client C during the registration, there are several steps for the two servers S1 and S2 to authenticate the client C and establish secret session keys with the client C in terms of parallel computation. The client C randomly chooses an integer r from Zq , computes R and then broadcasts a request message M to the two servers S1 and S2. On receiving M1, the server S1 randomly chooses an integer r1 from Zq and computes A2' and B2'. The server S2 randomly chooses an integer r2 from Zq and computes A1' and B1'. Then, S1 and S2 exchange M2 and M3. On receiving (A1', B1') the server S1 randomly chooses an integer r1' from Zq , computes R1, K1 and h1 and replies M4 to the client C. On receiving (A2', B2') the server S2 randomly chooses an integer r2' from Zq , computes R2, K2 and h2 and replies M5 to client C.

C. *Authentication*

In this module after receiving M4 and M5, the client C computes K'1 and K'2 and checks if PwC is same. If so, the two servers S1 and S2 are authentic. The client C computes h'1 and h'2 and broadcasts M6. At last, the client C sets the secret session keys with S1 and S2 respectively. On receiving M6, the server S1 checks if H() * b1 is same as h'1. If so, S1 concludes that the client C is authentic and sets the secret session key with the client C as SK1. The server S2 checks if H() * b2 is same as h'2. If so, S2 concludes that the client C is authentic and sets the secret session key with the client C as SK2

III. RESULTS

The concept of this paper is implemented and different results are shown below, The proposed paper is implemented in Java technology on a Pentium-IV PC with 20 GB hard-disk and 256 MB RAM with Java Environment. The propose paper's concepts shows efficient results and has been efficiently tested on different instances. The Fig 1, Fig 2, Fig 3 and Fig 4 shows the real time results compared.

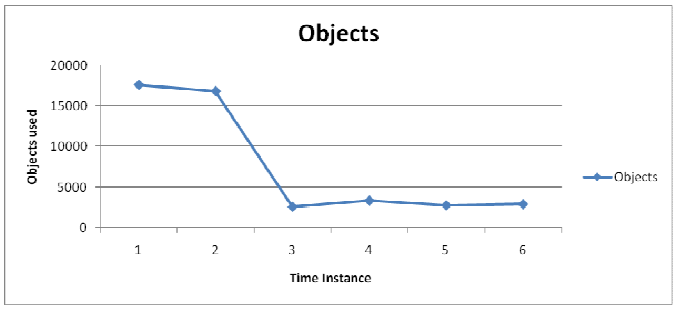

Fig. 1 Time taken by Node to initialize by objects.

Time Instance	Objects
1	17632
2	16854
3	2564
4	3325
5	2754
6	2892

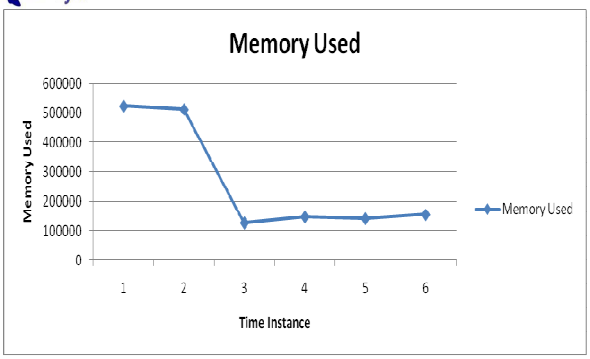

Fig. 1 Time taken by Node to compute Memory Used

Time Instance	Memory Used
1	523752
2	512412
3	127072
4	146904
5	141544
6	154596

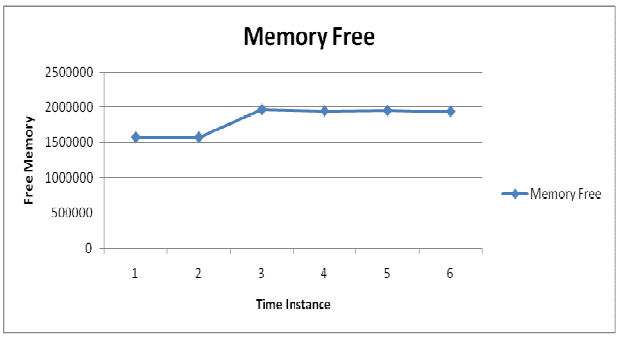

Fig. 4 Time taken by Node Communicating and Free MEmory

Time Instance	Memory Free
1	1573400
2	1572421
3	1970080
4	1950248
5	1955608
6	1942556

IV. CONCLUSIONS

With this paper, we have presented a symmetric method for two server password only authentication as well as key change. Security analysis has demonstrated that each of our protocol is usually secure towards passive as well as active attacks if one of the two hosting space is sacrificed. Performance analysis has demonstrated that each of our protocol is more effective than present symmetric as well as asymmetric two-server PAKE methodologies.

REFERENCES

[1] J. Katz, P. MacKenzie, G. Taban, and V. Gligor, "Two-Server Password-Only Authenticated Key Exchange," Proc. Applied Cryptography and Network Security (ACNS '05), pp. 1-16, 2005.

[2] X. Yi, R. Tso, and E. Okamoto, "Identity-Based Password- Authenticated Key Exchange for Client/Server Model," Proc. Int'l Conf. Security and Cryptography (SECRYPT '12), pp. 45-54, 2012.

[3] X. Yi, R. Tso, and E. Okamoto, "ID-Based Group Password- Authenticated Key Exchange," Proc. Fourth Int'l Workshop Security: Advances in Information and Computer Security (IWSEC '09), pp. 192- 211, 2009.

[4] M. Abdalla and D. Pointcheval, "Simple Password-Based Encrypted Key Exchange Protocols," Proc. Int'l Conf. Topics in Cryptology (CT-RSA), pp. 191-208, 2005.

[5] M. Abdalla, O. Chevassut, and D. Pointcheval, "One-Time Verifier-Based Encrypted Key Exchange," Proc. Eighth Int'l Conf. Theory and Practice in Public Key Cryptography (PKC '05), pp. 47-64, 2005.

[6] D. Boneh and M. Franklin, "Identity Based Encryption from the Weil Pairing," SIAM J. Computing, vol. 32, no. 3, pp. 586-615, 2003.

[7] J. Brainard, A. Jueles, B.S. Kaliski, and M. Szydlo, "A New Two- Server Approach for Authentication with Short Secret," Proc. 12th Conf. USENIX Security Symp., pp. 201-214, 2003.

[8] P. Mackenize, T. Shrimpton, and M. Jakobsson, "Threshold Password-Authenticated key Exchange," Proc. 22nd Ann. Int'l Cryptology Conf. (Crypto '02), pp. 385-400, 2002.

[9] J. Katz and M. Yung, "Scalable Protocols for Authenticated Group Key Exchange," Proc. Advances in Cryptology Conf. (Crypto '03), pp. 110-125, 2003.

[10] M. Di Raimondo and R. Gennaro, "Provably Secure Threshold Password Authenticated Key Exchange," Proc. 22nd Int'l Conf. Theory and Applications of Cryptographic Techniques (Eurocrypt '03), pp. 507-523, 2003.

[11] Y. Yang, F. Bao, and R.H. Deng, "A New Architecture for Authentication and Key Exchange Using Password for Federated Enterprise," Proc. 20th IFIP Int'l Information Security Conf. (SEC '05), pp. 95-111, 2005.

.

Authours Biography

Mrs Priyanka Madhiraju.

M.Tech

Mrs Priyanka Madhiraju, has done her M.Tech in Software Engineering from JNTUH. She has 05 years of teaching experience. Currently working as Asst. Professor CSE at Jagruti Institute of Engineering &Technology, Chintapalliguda(V),Ibrahimpatnam(M) R.R.Dist, She has guided many projects for UG and PG students.

G Deepa

M.Tech Scholar , Department of Computer Science & Engineering,

Jagruti Institute of Engg. & Tech. Chintapalliguda(V),IBP(M),RR Dist-501510.

e-mail: deepa.gunna@gmail.com

M.Tech scholar in **Jagruti Institute of Engg. & Tech**.

I completed my bachelor of technology from raja mahendra college of engineering.An enthusiastic fresher with highly motivated and leadership skills having bachelors of engineering degree in Computer Science & Engineering. Always willing to innovate the new things which can improve the existing technology. Eager to learn new technologies and methodologies.