Volume 6, Issue 6, June 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

# Mind Metrics: Identifying Users without Their Login IDs and Two-Server Password-Only Authenticated Key Exchange with Implementation

[1]Durgesh M. Chaudhari, [2]Prof. Yogesh S. Patil, [3]Prof. D. D. Patil
[1] P.G.Student, S.S.G.B.C.O.E.T., Bhusawal, Jalgaon, Maharashtra, India
[2] Assistant Professor, S.S.G.B.C.O.E.T., Bhusawal, Jalgaon, Maharashtra, India
[3] H.O.D. of Computer Science & Engg. S.S.G.B.C.O.E.T., Bhusawal, Jalgaon, Maharashtra, India

***Abstract—having been a core feature of IT systems for several decades, passwords continue to represent both one of the most familiar and most maligned aspects of security technology. While their potential weaknesses have been well recognized mainly over the past decade, no permanent solution has come up yet as in terms of all-round usage and applicability. Shoulder surfing, simple guessing, external eavesdropping, side channel attacks etc are the common methods which lead to password leakages. The situation gets worse when a user puts a very obvious password which can be easily guessed by anyone knowing the person even vaguely. Most systems propose to improve both identification and verification of user but this method of mind-metrics can augment the current password based systems by strengthening the identification process. Mind-metrics utilizes personal secret data instead of a login id to identify a user uniquely. The proposed system also creates a scenario where two servers cooperate to authenticate a client and if one server is compromised, the attacker still cannot pretend to be the client with the information from the compromised server. The proposed system presents a symmetric solution for two-server key encryption, where the client can establish different cryptographic keys with the two servers, respectively.***

***Keywords: User authentication, Password leakage, Mind-metrics, Mind-metrics token, two-server password, El-Gamal Encryption, Diffie-Hellman key exchange.***

## I. INTRODUCTION

Computer systems employ an authentication mechanism to allow access only to legitimate users. The authentication procedure is composed of two parts, identification & verification. The identification is for answering the question, "who am I?", and the verification is for answering, "Am I who I claim I am?". Traditionally the identification is performed with a username or login ID and the verification is done with a password. In a password-based system, the plaintext passwords are transformed into hash values with a one-way hash function, and stored in a password hash file. During the verification process, a new hash value is generated from the newly entered password, and compared with the stored hash value in the password hash file. If the hash values match, access is granted. This password verification process is the heart of the most authentication systems. There are a number of ways to acquire other users' password for illegal access. Plaintext passwords can be captured from the network, by malware or by key logging software. When the plaintext password is not available, the attackers can try password-guessing attack where they try possible values for the victim user. In the password cracking attack, the attackers obtain a password hash file and tries different inputs to find an input that produces the same hash value as the victim user's hash value. Sample password hash file Password hash files are stolen quite often and cracked by hackers. Password cracking attack is a statistical attack, and some of the weak passwords can be broken through a dictionary attack or a hybrid attack. After the attackers crack some passwords, they can access the system using the known login IDs for the cracked passwords.

The attack against passwords is a serious threat to current authentication systems, and additional security measures are needed to mitigate this threat. Once an account is breached, the cost from the damage is high for both victims and the companies. In 2013, the average organizational cost of data breaches in US was $5.4 million and the average per capita cost was $188 in US. [1]. While passwords are supposed to be random characters, login IDs are not random. They are used for communication or accounting purposes, and must carry a meaningful pattern. It may be part of users' first and/or last names, part of social security number, combination of names and numbers, account number, or email addresses. Thus login IDs are publicly known or can be guessed easily. In other words, obtaining the login ID is generally not a barrier for the attackers, and the success of an attack depends on the difficulty of the password. While a great emphasis was given to the verification, i.e., password system. somewhat less attention was given to the identification, i.e., login ID. By fortifying the identification part, the overall authentication system can be strengthened. The goal of this research is improving the security of the authentication system by supplementing it with a secure identification process. To make false login attempts difficult, 2014 IEEE International Conference on Systems, Man, and Cybernetics October 5-8, 2014, San Diego, CA, USA our method does not use a publicly known login ID for identification. Instead it uses private information known only to the computer system and the user. This process makes the stolen password files unusable for the attackers. In the following sections, we survey the previous works, describe the proposed method, explain the demonstration system, and present the survey results from the test users.

 

## II. RELATED WORK

Adhikari and S. Sikdar [1] says that Visual cryptography (VC) is a s pecial type of secret sharing. In this Report have proposed a ne w approach for phishing ebsites classification to solve the problem of phishing. The use of images is explored to preserve the privacy of image captcha by decomposing the original image cap tcha into two shares that are stored in separate database servers such that the original image captcha can be revealed only when both are simultaneously available; the individual sheet imag es do not reveal the identity of the original image captcha. Once the original image captcha is revealed to the user it c an be used as the password. Several solutions have been proposed to tackle phishing.

Juyeon Jo, Yoohwan Kim, and Sungchul Lee [2] Authentication to a computing system is composed of two parts, identification and verification. Traditionally, login IDs have been used for identification and passwords for verification. Many schemes have been proposed to improve both parts, but they may require specialized devices or they may not be always reliable. We propose a method that can augment the current password-based system by strengthening the identification process. It utilizes personal secret data instead of a login ID to identify a user uniquely, hence mind metrics. It then asks the user to choose a correct login ID among multiple choices of partially obscured IDs. Since it does not accept a login ID during the authentication process, a stolen or cracked password cannot be used for gaining an access to the computing system unless the attacker provides a correct identification material, i.e., mind metrics token. This additional step raises the security of an authentication system considerably over single or double password systems. Since the stolen passwords cannot be used immediately by the attackers, account holders can have extra time to change their passwords before the attackers gain an access. This scheme does not require any specialized hardware and can be implemented easily. It may be used where biometrics schemes cannot be used cost-effectively, e.g., on public e-commerce web sites. Mind metrics scheme separates the identification server and the verification server, thus it is scalable to a large system. We implemented a proof-of-concept system and evaluated it with test users. The survey indicates that the system is not as intrusive as other schemes, users feel better protected, and they are willing to use the scheme on public web sites**.**

Xun Yi, San Ling, and Huaxiong Wang [3] Password-authenticated key exchange (PAKE) is where a client and a server, who share a password, authenticate each other and meanwhile establish a cryptographic key by exchange of messages. In this setting, all the passwords necessary to authenticate clients are stored in a single server. If the server is compromised, due to, for example, hacking or even insider attack, passwords stored in the server is all disclosed. In this report, we consider a scenario where two servers cooperate to authenticate a client and if one server is compromised, the attacker still cannot pretend to be the client with the information from the compromised server. Current solutions for two-server PAKE are either symmetric in the sense that two peer servers equally contribute to the authentication or asymmetric in the sense that one server authenticates the client with the help of another server. This report presents a symmetric solution for two-server PAKE, where the client can establish different cryptographic keys with the two servers, respectively. Our protocol runs in parallel and is more efficient than existing symmetric two-server PAKE protocol, and even more efficient than existing asymmetric two-server PAKE protocols in terms of parallel computation.

C. Blundo, S. Cimato, and A. De Santis Stinson [4] introduces A visual cryptography scheme encodes a black and white secret image into n shadow images called shares which are distributed to the n participants. Such shares are such that only qualified subsets of participants can "visually" recover the secret image.

## III. IDENTIFICATION WITHOUT LOGIN ID

In this paper I present the main results of this Seminar. I prove that there is a Identification is a process to recognize an unknown individual out of many (1:N relationship). All authentication systems require some form of ID for user identification, such as text-based login ID, fingerprint, or facial image. Then in the verification process, the system asks a proof of the ID since the system does not know whether the user is the legitimate ID holder (1:1 relationship). Generally the login ID is not considered a secret. For example, a social security number or an email address may serve as a login ID. It is impractical to keep the login ID secret because it is used for many other purposes such as accounting or email. So an alternative secret is needed for identification to recognize a user uniquely. This secret is referred as Mind metrics token. We coined the term Mind metrics due to the similarity to biometrics. In case of biometrics, only the legitimate holder of the physical trait (e.g., fingerprint) can pass the identification stage. Similarly, with some kind of personal secret knowledge (mind metrics token), a user can pass the identification stage, thus the effect of biometrics is simulated. Without this secret knowledge, the attackers cannot pass the identification stage before they can even try the password the process of identification in Mind metrics uses something in user's mind instead of something on their body. There are two parts in the mind metrics-based authentication process. First, mind metrics token is requested in the login page. A user specifies the token with which a computing system can identify a user account. Then the identification server looks up the registered access tokens to find a matching token and login ID. Second, the server presents multiple login IDs to the user, with one of the login IDs being the correct login ID for the user account and some more real or fake IDs. To prevent the attackers from recognizing the login IDs, the login IDs are partially obscured among these partial login IDs, a legitimate user can still recognize the correct login ID and choose it. This completes the identification stage, and the rest is same as password verification system. If the login ID and password match the credentials stored for the user account, the user is authenticated to access information associated with the user account. for no information is given back to the user, so the attacker will not know whether he entered a valid access token, chose a wrong login ID, or entered a wrong password. Compares the conventional password-based system and the proposed mind metrics-based system. While the

 

password-based system allows anyone to try publicly known login IDs without any restriction, mind metrics-based system allows only the legitimate users to pass the identification stage. Now the password verification server is hidden, and users cannot access it unless they pass the identification server. Compared with the conventional password-based system, the only difference is the addition of the identification server, so an existing password-based system can be easily upgraded by just adding the identification server. Mind metrics system has several benefits over biometrics. The characteristics in mind metrics can be changed if needed. It does not requires any specialized device, so it can be used for accessing local or remote computing systems from any conventional private or public computers. It is a deterministic process, and thus there is no uncertainty. Thus mind metrics is more practical, cheaper, and can be used by any public web sites such as e-commerce web sites

## IV. TWO-SERVER PASSWORD –ONLY AUTHENTICATION AND KEY EXCHANGE

Now Days, passwords are commonly used by people during a log in process that controls access to protected computer operating systems, mobile phones, cable TV decoders, automated teller machines and so on. A computer user may require passwords for many purposes: logging in to computer accounts, retrieving e-mail from servers, accessing programs, databases, networks, web sites, and even reading the morning news report online. Earlier password-based authentication systems transmitted a cryptographic hash of the password over a public channel which makes the hash value accessible to an attacker. When this is done, and it is very common, the attacker can work offline, rapidly testing possible passwords against the true password's hash value. Studies have consistently shown that a large fraction of user-chosen passwords are readily guessed automatically. For example, according to Bruce Schneider, examining data from a 2006 phishing attack, 55 percent of MySpace passwords would be crack able in 8 hours using a commercially available Password Recovery Toolkit capable of testing 200,000 passwords per second in [1].

Recent research advances in password-based authentication have allowed a client and a server mutually to authenticate with a password and meanwhile to establish a cryptographic key for secure communications after authentication. In general, current solutions for password based authentication follow two models. The first model, called PKI-based model, assumes that the client keeps the server's public key in addition to share a password with the server. In this setting, the client can send the password to the server by public key encryption. Gong et al. were the first to present this kind of authentication protocols with heuristic resistant to offline dictionary attacks, and Halevy and Krawczyk [2] were the first to provide formal definitions and rigorous proofs of security for PKI-based model.

The second model is called password-only model. Bellowing and Merritt [2] were the first to consider authentication based on password only, and introduced a set of so-called "encrypted key exchange" protocols, where the password is used as a secret key to encrypt random numbers for key exchange purpose. Formal models of security for the password-only authentication were first given independently by Bellaire et al. [2] were the first to give a password-only authentication protocol which is both practical and provably secure under standard cryptographic assumption. Based on the identity-based encryption technique suggested an identity-based model where the client needs to remember the password only while the server keeps the password in addition to private keys related to its identity. In this setting, the client can encrypt the password based on the identity of the server.

This model is between the PKI-based and the password only models. Typical protocols for password-based authentication assume a single server stores all the passwords necessary to authenticate clients. If the server is compromised, due to, for example, hacking, or installing a "Trojan horse," or even insider attack, user passwords stored in the server are.

**Two servers Model**

In our system, there exist two servers S1 and S2 and a group of clients. The two servers cooperate to authenticate clients and provide services to authenticated clients. Prior to authentication, each client C chooses a password pwC and generates the password authentication information Authð1Þ C and Authð2Þ C for S1 and S2, respectively, such that nobody can determine the password pwC from Authð1Þ C or Authð2Þ C unless S1 and S2 collude. The client sends Authð1Þ C and Authð2Þ C to S1 and S2, respective, through different secure channels during the client registration. After that, the client remembers the password only, and the two servers keep the password authentication information. Like all existing solutions for two-server PAKE, we assume the two servers never collude to reveal the password of the client. When the two servers cooperate to authenticate a client C, we assume that the client C can broadcast a message to both of S1 and S2 simultaneously, but stress that we do not assume a broadcast channel and, in particular, an attacker can deliver different messages to the two servers or refuse to deliver a message to a server.

In our protocol, the client and the two servers communicate through a public channel which may be eavesdropped, delayed, replayed, and even tampered by an attacker. Our protocol is symmetric if two peer servers equally contribute to the authentication in terms of computation and communication Definition.

Our protocol is correct if each server establishes a secret session key with the client in the end. An adversary in our system is either passive or active. We consider both online dictionary attacks, where an attacker attempts to login repeatedly, trying each possible password, and offline dictionary attack, where an adversary derives information about the password from observed transcripts of log sessions.

The online dictionary attack cannot be prevented by cryptographic means but can be easily detected and suspended once the authentication fails several times. Lower computational cost since the secret message is recognized only by human eyes and not cryptographically computed.

 

## V. SYSTEM DESIGN AND IMPLEMENTATION

### *A. Step 1: Mindmetrics token registration*

Figure 5.1 Show a webpage for creating an account with a mindmetrics token account login ID and password. In this illustration, a user types the access token "This is my secret #7", and specifies a desired login ID and password. Alternatively, the login ID may be given by the server. After the user has entered them, the user selects the "Create Account". Then the authentication server validates the entered information, for example, to ensure that the login ID is unique among all login IDs and that the password satisfies the password requirements (e.g., a length of the password). If there is an error, it may prompt the user to enter another login ID or password. To increase security, it may use special keys (e.g., CTRL and ALT) in combination with other alphanumeric keys or insert spelling errors intentionally in random locations. During the account creation process, we can examine the existing tokens and reject similar tokens. In particular, we use similar text () function, a recursive divide-and-conquer algorithm with O (n3) complexity where n is the longest string. Given two strings, the function returns a similarity value in percentage. Show the result of the account creation. The webpage displays the token, the username, and the password (which may be displayed masked or unmasked). A number of fake IDs are created together and will be displayed during normal login process. In some implementations, the user may be able to specify the fake usernames. The account information is stored in separate places. The plaintext tokens are used only for similarity test, and not used for normal login process. The Mindmetrics tokens are stored in a hashed form in the token hash file as {token hash value, index} tuple and the matching login IDs are stored in the index file as {index, fake login ID, fake login ID, true login ID, fake login ID…} tuple. The password hash file contains <login ID, password hash value> same as before.

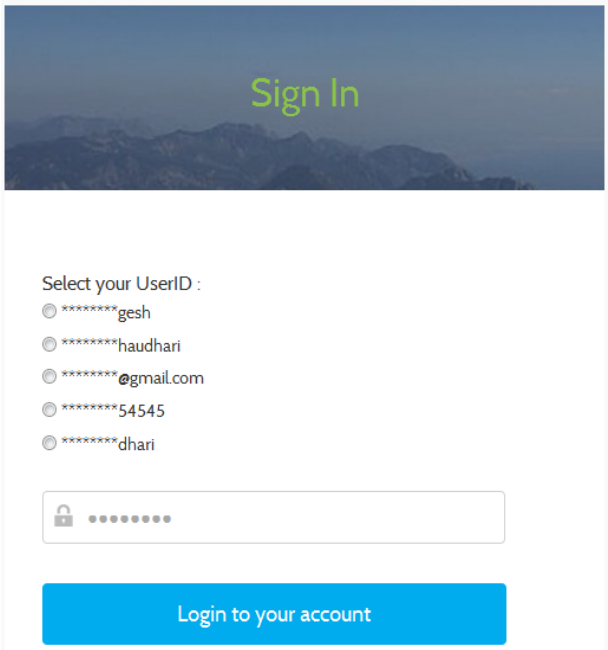


Figure 5.1 Generated User Name after Token

### *B. Step 2 AES Algorithm*

In our system, there exist two servers S1 and S2 and a group of clients. The two servers cooperate to authenticate clients and provide services to authenticated clients. Prior to authentication, each client C chooses a password pwC and generates the password authentication information Authð1Þ C and Authð2Þ C for S1 and S2, respectively, such that nobody can determine the password pwC from Authð1Þ C or Authð2Þ C unless S1 and S2 collude. The client sends Authð1Þ C and Authð2Þ C to S1 and S2, respective, through different secure channels during the client registration. After that, the client remembers the password only, and the two servers keep the password authentication information. Like all existing solutions for two-server PAKE, we assume the two servers never collude to reveal the password of the client. When the two servers cooperate to authenticate a client C, we assume that the client C can broadcast a message to both of S1 and S2 simultaneously, but stress that we do not assume a broadcast channel and, in particular, an attacker can deliver different messages to the two servers or refuse to deliver a message to a server. In our protocol, the client and the two servers communicate through a public channel which may be eavesdropped, delayed, replayed, and even tampered by an attacker. Our protocol is symmetric if two peer servers equally contribute to the authentication in terms of computation and communication Definition. Our protocol is correct if each server establishes a secret session key with the client in the end. An adversary in our system is either passive or active. We consider both online dictionary attack, where an attacker attempts to login repeatedly, trying each possible password, and offline dictionary attack, where an adversary derives information about the password from observed transcripts of log sessions. The online dictionary attack cannot be prevented by cryptographic means but can be easily detected and suspended once the authentication fails several times. We assume that an adversary can compromise one server n only and obtain all information stored in the server. A passive adversary is able to monitor the communications among the client and two servers. An active adversary is able to pretend to be both one server and the client to communicate with the honest server or pretend to be both two

 

servers to communicate with the legal client, deviate in an arbitrary way from the actions prescribed by the protocol. In our protocol, the adversary attempts to learn the secret session key established between the client and the honest server. In an active attack, an adversary can learn the secret session key between the client and the honest server if the adversary can determine the password of the client. In general, we say that our protocol is secure if no adversary can succeed in any passive and active attacks in case that one server is compromised.

### *C. Step 3: Mindmetric token submission*

The text that is typed may be masked so that a nearby individual is not able to discern the token. The identification server receives user input specifying a token, and determines whether the token matches a token stored for a user account. For example, it may identify that the login ID "frank1982" is assigned to a user account to which the token "ilikemathematics" is assigned. We expect that the tokens will be long, diverse, and mixed with special keys and intentional misspellings. It is difficult to type such a complex token without an error. Moreover, with non-text inputs such as voice are used as a token, the accuracy may be lower, too. In this case, the similar text() function can be used to allow a certain level of error. The tolerance level can be configured dynamically by the user. For example, the token hash value may be used at the first try, then we can gradually increase the tolerance level upon each failure. As Shown in Figure 5.2

USER DETAILS REPORT

| # | User ID | User Name | Email | Mindmetrics Token (Hash) | Password Part1(Hash) | Password Part2(Hash) |
|---|---|---|---|---|---|---|
| 1 | durgeshid | Durgesh Chaudhari | durgesh.mcoeit@gmail.com | durgesh@123 (zcHV?Ê-ú®\|) | durge (zcHV) | sÔ?Tô (?Jòì) |
| 2 | domalsanket | sanket domal | domalsanket@gmail.com | i am sanket (/0æû?ÂÛ©LÉ) | doma (`÷ª) | l123 («2) |
| 3 | durgeshchaudhari | Durgesh Chaudhari | sanketdml@gmail.com | I like mango (=ëý=Ø8¾?d»[) | mang ( kö) | o@123 (¬ñt ) |

Figure 5.2 Mindmetric token

### *D. Step 4: Login ID selection*

After the matching login ID is determined, the server does not display it immediately because an attacker can discover the token and the matching login ID. It presents multiple login IDs and let the user select one. The number of choices can be configured as desired by the identification server or the user as shown in Figure 5.1.

1) Partial obscurity The displayed login IDs are partially-obscured by replacing some characters in the username with asterisks. For example, a user knows that his username is "frank1982," and therefore selects a radio button to indicate that the text "****k1982" matches the username. Any part of the username can be obscured, for example, f*a*k*9*2 or frank****. This is not a problem for a legitimate user, but it makes it difficult for the attackers to recover the full login ID.

2) Multiple login ID The authentication server identifies the associated fake login IDs from the index table ( (b)). They may include real login IDs of other users.

3) Mutual authentication the names and the order of the login IDs are fixed at time of account creation. Over time, a user is able to learn through user experience which partially-obscured login ID matches the true login IDs. In this way, the user is able to quickly log on to the account without much additional effort. This helps the user to recognize a fake server performing a man-in-the-middle attack (MITM). When the displayed login IDs are different from what the user expects, it is either due to a typo in the token input, or due to an MITM attack.

4) Preventing inference attack with fixed choices. The displayed list of partially-obscured login IDs is same and appears in the same order every time. An attacker cannot simply supply the same token with a different computing system (using a different IP address) in order to identify which partially-obscured login ID does not change.

5) Handling non-existant tokens If the token does not match any token in the token hash file, the identification system creates random login IDs and show it the user. This prevents an attacker from knowing whether the entered token matches any user accounts. To prevent an inference attack, the choice of the displayed login IDs do not change when the attacker types the non-existing token again. A hashing algoithm can be used to generate a fixed set of login IDs consistently for a given non-existing token.

### *E. Step 4: Password verification*

After selecting a radio button to choose a login ID, the user types the password that matches the login ID. The login ID and password is given to the verification server. If they match the credentials assigned to the user account, the verification server grants an access. Should the user have entered credential information that did not match all the information assigned to the user account, the server indicates that the user entered incorrect credentials and try it again from the beginning. The verification server may not indicate to the user which of The token, login ID, or password caused the failure. Computer systems employ an authentication mechanism to allow access only to legitimate users. The authentication procedure is composed of two parts, identification & verification. The identification is for answering the question, "who am I?", and the verification is for answering, "Am I who I claim I am?". Traditionally the identification is

 

performed with a username or login ID and the verification is done with a password. In a password-based system, the plaintext passwords are transformed into hash values with a one-way hash function, and stored in a password hash file. During the verification process, a new hash value is generated from the newly entered password, and compared with the stored hash value in the password hash file. If the hash values match, access is granted. This password verification process is the heart of the most authentication systems. There are a number of ways to acquire other users' password for illegal access. Plaintext passwords can be captured from the network, by malware or by key logging software. When the plaintext password is not available, the attackers can try password-guessing attack where they try possible values for the victim user. In the password cracking attack, the attackers obtain a password hash file and tries different inputs to find an input that produces the same hash value as the victim user's hash value. Sample password hash file Password hash files are stolen quite often and cracked by Hackers Password cracking attack is a statistical attack, and some of the weak passwords can be broken through a dictionary attack or a hybrid attack. After the attackers crack some passwords, they can access the system using the known login IDs for the cracked passwords. The attack against passwords is a serious threat to current authentication systems, and additional security measures are needed to mitigate this threat. Once an account is breached, the cost from the damage is high for both victims and the companies. In 2013, the average organizational cost of data breaches in US was $5.4 million and the average per capita cost was $188 in US. While passwords are supposed to be random characters, login IDs are not random. They are used for communication or accounting purposes, and must carry a meaningful pattern. It may be part of users' first and/or last names, part of social security number, combination of names and numbers, account number, or email addresses. Thus login IDs are publicly known or can be guessed easily. In other words, obtaining the login ID is generally not a barrier for the attackers, and the success of an attack depends on the difficulty of the password. While a great emphasis was given to the verification, i.e., password system, somewhat less attention was given to the Identification, i.e., login ID. By fortifying the identification part, the overall authentication system can be strengthened. The goal of this research is improving the security of the authentication system by supplementing it with a secure identification process. To make false login attempts difficult, Identification is a process to recognize an unknown individual out of many (1:N relationship). All authentication systems require some form of ID for user identification, such as text-based login ID, fingerprint, or facial image. Then in the verification process, the system asks a proof of the ID since the system does not know whether the user is the legitimate ID holder (1:1 relationship). Generally the login ID is not considered a secret. For Example, a social security number or an email address may serve as a login ID. It is impractical to keep the login ID secret because it is used for many other purposes such as accounting Or email. So an alternative secret is needed for identification to recognize a user uniquely. This secret is referred as Mindmetrics token. We coined the term Mindmetrics due to the similarity to biometrics. In case of biometrics, only the legitimate holder of the physical trait (e.g., fingerprint) can pass the identification stage. Similarly, with some kind of personal secret knowledge (mindmetrics token), a user can pass the identification stage, thus the effect of biometrics is simulated. Without this secret knowledge, the attackers cannot pass the identification stage before they can even try the password The process of identification in Mindmetrics uses something in user's mind instead of something on their body. There are two parts in the mindmetrics-based authentication process. First, mindmetrics token is requested in the login page . A user specifies the token with which a computing system can identify a user account. Then the identification server looks up the registered access tokens to find a matching token and login ID. Second, the server presents multiple login IDs to the user, with one of the login IDs being the correct login ID for the user account and some more real or fake IDs. To prevent the attackers from recognizing the login IDs, the login IDs are partially obscured Among these partial login IDs, a legitimate user can still recognize the correct login ID and choose it. This completes the identification stage, and the rest is same as password verification system. If the login ID and password match the credentials stored for the user account, the user is authenticated to access information associated with theuser account. For a failed login attempt, no information is given back to the user, so the attacker will not know whether he entered a valid access token, chose a wrong login ID, or entered a wrong password. compares the conventional password-based system and the proposed mindmetrics-based system. While the password-based system allows anyone to try publicly known login IDs without any restriction, mindmetrics-based system allows only the legitimate users to pass the identification stage. Now the password verification server is hidden, and users cannot access it unless they pass the identification server. Compared with the conventional password-based system, the only difference is the addition of the identification server, so an existing password-based system can be easily upgraded by just adding the identification server as Shown in Figure 5.3 .

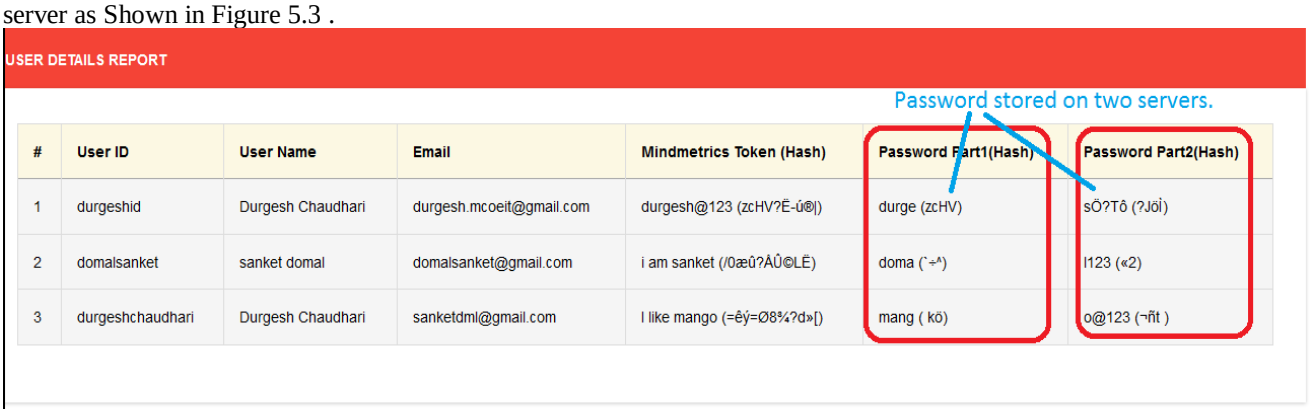


Figure 5.3 Mindmetric Password verification

 

## VI. CONCLUSION

Authentication is done in two steps, identification and verification. The traditional password-based verification system has been challenged by sophisticated attacks, but new schemes are being made to cover the weaknesses of the password-based systems. However, the identification part is still done based on a public login ID. We proposed a new scheme called mind metrics to strengthen the identification process with personal secret information. In mind metrics systems, a login ID is not asked. Instead a user must provide the correct token to pass the identification stage. In case the password file is stolen, the login attempts by attackers. I have presented a symmetric protocol for two-server password-only authentication and key exchange. Security analysis has shown that our protocol is secure against passive and active attacks in case that one of the two servers is compromised. Performance analysis has shown that our protocol is more efficient than existing symmetric and asymmetric two-server PAKE protocols blocked by the identification server. Thus it may stop or slow down attackers, and account holders can change their account credentials before attackers can gain access.

## ACKNOWLEDGMENT

I would like to express my sincere thanks to my guide Prof. Yogesh S. Patil & H.O.D. Prof. D. D. Patil for his motivation and useful suggestions which truly helped me in improving the quality of this paper and heartily thankful to IJARCSSE for giving me such a wonderful opportunity for publishing my paper.

## REFERENCES

[1] Juyeon Jo, Yoohwan Kim, and Sungchul Lee, "Mind metrics: Identifying users without their login IDs," IEEE International Conference on Systems, Man, and Cybernetics, 23- October -2015, vol. 5-8, PP. 448-161.

[2] Xun Yi, San Ling, and Huaxiong Wang, "Efficient Two-Server Password-Only Authenticated Key Exchange," IEEE TRANSACTIONS., Sep. 2013 vol, 24, no. 9,pp. 86–106.

[3] G. Ateniese, C. Blundo, A. De Santis, and D. R. Stinson, "Extended schemes for cryptography, "Theoretical Compute., vol. 25 , Aug. 2001, pp. 143–161.

[4] C. Blundo, A. De Bonis, and A. De Santis, "Improved schemes for visual cryptography, Des" Codes Cryptogr., vol. 24, no. 3, Dec. 2001, pp. 255–278.

[5] C. Blundo and A. De Santis, "Visual cryptography schemes with perfect reconstruction of black pixels," J. Compute. Graph., vol. 22,Jan. 1998 pp. 449–455.

[6] C. Blundo, A. De Santis, and D. R. Stinson, "On the contrast in visual cryptography schemes," J. Cryptol., vol. 12, no. 4,Sep. 1999, pp. 261–289.

[7] C. Blundo, P. D'Arco, A. De Santis, and D. R. Stinson, "Contrast optimal threshold visual cryptography schemes," SIAM J. Discrete Math., 2003vol. 16,no. 2, pp. 224–261.

[8] C.-C. Chang, C.-C. Lin, T. H. N. Le, and H. B. Le, "Self-verifying visual secret sharing using error diffusion and interpolation techniques," *IEEE* Vol 26 , Jan 2014, Page No 1-4..

 