Proceedings of the 2007 IEEE
Workshop on Information Assurance
United States Military Academy, West Point, NY 20-22 June 2007

A GLOBAL LOOK AT AUTHENTICATION

Stephen S. Hamilton, Martin C. Carlisle, and John A. Hamilton Jr.

***Abstract*-- In today's world of increased connectivity, authentication issues are becoming increasingly important. Many user accounts, from banking to water bill accounts, are available online. How do we ensure that identity theft occurrences are reduced, and people can enjoy the benefits of managing multiple accounts online? There are numerous schemes and systems that exist today, but many are difficult to implement, especially for the end users. When analyzing a system from a technical standpoint, one might be able to set the security policies and password encryption strength to the appropriate level in order to protect the information it contains, however, when multiple users utilize this system, and they in turn use other systems with identical passwords, the initial system can become less secure. In addition, phishing attacks are becoming more popular, where users may enter the password they reuse on a hostile site, which could result in compromising multiple systems. This paper examines various authentication methods and mechanisms available today, and determines which is appropriate for various uses. In general, users and administrators are responsible for security, and they should treat their systems with regard to the sensitivity of what they want to protect. Users should always consider the importance of each system they use and choose an authentication method or password scheme to match the system they intend to operate. In addition, administrators should always consider the consequences of implementing different security measures, to include the usability of their system.**

***Index Terms*— Authentication, biometrics, key, passwords, smartcard, security**

I. INTRODUCTION

IN order for a system to be secure, all users and administrators of that system must use good security practices. The average user knows that a harder to guess password, a hardware key, or biometric device will increase security, but users do something different: they do the minimum authentication required to make a system work, and then become complacent. The problem is not that users do not care about security; it is that the harder a password is to break, the harder it is to remember. In addition, the more complex an authentication means becomes with the use of extra software or hardware, the less likely users will want to use it (unless they are forced). Computers have increased in complexity and ability, but humans have not responded the same way. It is relatively easy for an administrator to increase the security of a system, but it is difficult for an administrator to make users act in a more secure manner. Mechanisms like increasing password complexity requirements often only lead to users writing down passwords and reusing the same password on multiple systems. One shocking statistic states that approximately 30% of all ATM cards have their PIN written on it [1]. Since nearly 1/3 of the users choose to write down a 4 digit number, this statistic may only be larger when the users are required to remember an eight character alpha numeric upper and lower case password. The problem is only growing with the number of systems people access, and the increase of security threats.

II. MULTIPLE SYSTEM PROBLEM

The number of online accounts users access today are continuing to increase. As more organizations move toward doing business online, users are faced with more passwords. In order to increase security, many organizations have increased password strength and increased the number of times users are required to change their password. The effectiveness of this method depends on the users and their implementation. The perfect user will have different passwords for each different account, and he or she will be able to remember 8 to 10 character length passwords with uppercase, lowercase, special characters, and numbers. If this is the case, the perfect user is not realistically human. Consider the typical accounts: banking, power bill, cable bill, internet bill, investment accounts, water bill, cell phone accounts, student email and/or work accounts. The number of accounts makes it difficult to remember all these different passwords. Therefore users perform various techniques to help themselves: writing the passwords down on paper, saving them in a text file on their computer, using the same password for multiple systems, and creating password patterns. This helps the user, but it also helps an attacker trying to break the same passwords. If an attacker compromises a computer and discovers an unencrypted text file containing passwords to that user's various accounts, complete identity theft can occur. The same can also happen to users who write down their passwords on paper and lose it.

Another concern is what a many users do: use the same password for multiple systems. It is not difficult to remember a 10 character password with good complexity if it is used daily and on multiple systems. This may make the users feel secure; however it can potentially reduce the security of the systems they access. If all the systems are trustworthy and use strong encryption to store their passwords, using the same password may not present a security issue. However, the user takes a risk, since the chance of all the systems using strong hashing techniques on passwords is low. Passwords are stored in plaintext for different systems, and some not intentionally [2]. Consider the systems on the Internet that allow you to have your password emailed back to the user when that user forgets their password. If the password is emailed in plaintext, it may have never encrypted it in the first place or used a reversible encryption scheme. In addition, passwords are often transmitted in plaintext over the Internet.

An example is the widely used MSN Hotmail, which can use Transportation Layer Security (TLS), but by default uses only the plaintext http protocol. Each user has a different scenario, and logically one could expect that some organizations employ stronger security measures than others due to the value of an attack. It would not be worth attacking the city water department to gain access to others' accounts, because the attacker could only gain water usage reports and the ability to pay the water bill. However an attacker would be likely to spend money and resources to gain access to a bank account if the result was the ability to wire money into an offshore account that the attacker owns. The bank is more likely to spend money and resources on security systems, policies, and administration, whereas the water department has no need for bank strength security. This is where the problem lies with using the same password. An astute hacker would spend a little time and resource to compromise all the accounts and passwords on the water department, because it is more likely to go unnoticed and could be an easier task than hacking into the bank. Once the attacker has that information, he or she can then run the username password combinations against the bank, and will get access to all users who use the same password. Therefore, the bank security system is only as strong as the water department's system when passwords are reused, even though the two systems are unrelated other than having common users.

Another issue is password recovery. A bank may have a stringent policy by validating your SSN, account number, places of previous transactions, etc. whereas the water department may only ask for the user's mother's maiden name. Here the problem arises again: Determining the mother's maiden name may be as easy as surfing on a genealogy site, and thus getting the password may be trivial on the water department's site. Therefore the recovery system further increases the risk the users took by using the same password at their bank. Taking this example one step further, an attacker may hack into the water account, and use the same password to get into the victim's email account. This can unleash even a larger problem: unlocking all kinds of accounts by resetting them over email, which is the standard practice for most online accounts. Users must realize that if their systems authenticate them via email, their email account password must be at least as strong as the strongest password for the accounts they manage.

III. THE MULTIPLE SYSTEM PROBLEM

A. Methods

There are many solutions available to solve some of these problems. It is important to evaluate each system in terms of implementation and usability. If either or both are too hard, the average user will typically operate in a less secure, easier to use environment. There are three major ways to authenticate an individual: "by something a person knows, by something a person has, or by something a person is" [3]. We will also discuss a fourth option, who a person knows, but the first three are the most commonly used today. In general, the primary three methods translate to passwords, hardware keys, and biometrics. Based off of these categories, different methods of implementation methods should be examined in order to determine what means should be used. There are pros and cons for each type in the areas of usability and security, and implementation has an impact as well. The following authentication mechanisms and techniques will be discussed throughout this paper to explore which is sensible, able to be implemented, and secure for the users and administrators without making systems too difficult to use.

-Password Restriction based on account type
-Non keyboard text password entry
-Smartcards
-Biometrics
-Centralized Authentication
-Centralized password storage
-Behavior Based Authentication
-Authentication Recovery

B. Password Restriction based on Account Type

This method is policy driven. The concept of this method is categorizing account types, and defining specific restrictions to passwords to ensure a user cannot reuse the password. This does not solve the problem for the user; however it does help increase security in higher risk systems. A basic policy of this type when used against the water and bank accounts, would be the water account would require a password of no more than 6 characters, whereas the bank account would require the user to maintain a password of 8 characters or greater. This restriction ensures that the user cannot have the exact password on both systems. As mentioned earlier, the primary shortcoming is the user has to remember more than one password, so this only alleviates the password reuse problem. This does not ensure the user does not write both passwords down. This method also would be very difficult to implement, since there would be a need for a worldwide governing agency to categorize systems to give them their password complexity requirements. However the concerned individual user can categorize his or her accounts and implement this scheme individually. Some systems encourage strong passwords by giving the user feedback using a password strength meter. Typically users will at least think about the account and try and make the meter hit the maximum point by adding a number or symbols to their password. This method may work for many users, but an attacker could determine that some users just add a simple "1" or "!" to the easy password to make it meet complexity requirements. One other problem that arises from requirements is that if the encryption method isn't strong, the attacker may be able to brute force the accounts. The issue is

if the attacker knows the complexity requirements, he or she will know what to use in the brute force attack. For example, if the system requires 8 and only 8 characters and 2 numbers, the attacker will only run through passwords of length 8 with two numbers, which is a reduced search space. Another possible problem is what users commonly do to increase their passwords: add the current year or their birth year to a dictionary password to satisfy requirements. In this case, an attacker may be able to brute force attack passwords and find the users that use this scheme.

C. Non keyboard text password entry

Another method of entering passwords is by using something other than the keyboard. One method is to present an on-screen keyboard that the user uses with a mouse to enter the password. One of the immediate advantages of this method is that if a keylogger of any sort is running, it defeats the capture. Mouse capture can be performed; however it requires synchronization with on screen coordinates or screen capturing, making the capture more complex than keylogging. Although key loggers could be installed on a home machine via a remote connection, the more likely place to have one installed is on a public internet computer. In this case, the key logger will not capture the password, but "shoulder surfing" is also a lot more probable in a public setting. The on screen keyboard makes it much easier for the shoulder surfer to see the password as it is being entered. In addition to using the screen to authenticate the user, some companies are using a method to authenticate the website to the user, to ensure they aren't logging into a hostile site. This is performed by having the users pick a graphic on account setup, and once they type in their username in the future they should see this graphic. Although technically this works well, one study conducted showed that a very high percentage of users will continue to log in if they do not see their graphic [4]. This further demonstrates the problem that the majority of users perform the bare minimum tasks with regard to computer security.

D. Smartcards

A smartcard is typically a card the size of an ID or credit card, and it contains a silicon integrated circuit chip with a small CPU and memory (ROM and EEPROM). One common type of smartcard that provides authentication is the Advanced Secure Access Control System (ASACS), which provides symmetric and asymmetric keys for authentication and digital signatures [5]. The key advantage of the ASACS card is all the secret information stored on the card never leaves the card. This is due to the internal CPU that is used to perform the encryption/decryption [5].

The authentication piece of the smart card uses the ANSI standard X9.26. The authentication scheme is called Token Based Access Control System (TBACS) [6]. TBACS requires that the user maintains a password or PIN number that is entered after the card is inserted into a reader. The PIN/password is then encrypted and transmitted to the token on the smartcard along with the user ID. The card decrypts the PIN and encrypts the user ID with it and compares it to the value on the card, thus performing the authentication. After the user is verified, the system uses the token to represent the user's identity throughout the session [6]. This method of authentication is more secure than a strong password; however it has its drawbacks as well. The first drawback is the cost. The user that uses systems like Hotmail most likely enjoys free services on the Internet, and may not go out of his or her way to pay for a smartcard and a card reader. The card can range from $3.00 to $30 depending on the feature set (www.market.axalto.com), and the reader can cost around $35. This is reasonable for the conscientious user who understands security; however the majority of the users today are not going to go out of their way for this standard of security. If major computer manufacturers started bundling readers with their computers, and state governments converted driver's licenses to smart cards, then this method might be more acceptable.

There are other difficulties that arise with this card as well. One of the primary difficulties is handling a forgotten PIN or password. A good case study for the widespread usage of the smartcard is the Department of Defense. As of June 30th of 2006, all DOD employees were required to use their issued smartcard to access any government system. The DOD began issuing smart cards (called the Common Access Card or CAC) to the majority of servicemen and women in 2002, and eventually required all DOD members to have them. This was a very difficult task for members at all levels, especially due to the time to create the key and write it to the card (approximately 15-20 minutes). Once they were issued, the DOD policy enforced on the card was three attempts at the PIN which is 6-8 digits long, and then the card locks out the user. Once the user is locked out, he or she has to go to the issuing facility to have the card unlocked and PIN reset. At Ft. Hood, the largest active duty armored post in the United States, the CAC card implementation was an enormous task. At any given time there are approximately 42,000 uniformed soldiers, and 30,000 civilian employees of which the majority required CAC cards for DOD computer access. They tackled this problem by opening the CAC card issue point 24 hours a day, 7 days a week a few months prior to June of 2006. In addition to getting individual cards set up, the users also had to provision their accounts on the Ft. Hood network. The provisioning process is used to link the smart card identity to the existing Active Directory user account. Approximately one week before the deadline, many users experienced problems due to the provisioning server crashing and running at 100% CPU utilization caused by overwhelming requests. Although these kinds of problems were encountered, the

implementation was performed within the time constraint. If state governments were to perform the same feat with Driver's Licenses, the waiting lines would be much longer than they already are. There would have to be a better system for a PIN reset, since it would be extremely inconvenient to return to the Department of Motor Vehicles to have a PIN reset in order to access an online bank account.

The other problem that arises is the potential of having multiple smart card identities. Although it may be feasible for DOD users to have a common card used with their bank and the DOD, the involved organizations would depend on each other's security practices to ensure the legitimacy of the card. This could result in each agency issuing a unique smartcard. Also, there is the other major problem of losing the smartcard. When this happens, the key has to be revoked, and a new card issued. There is cost and time associated with this, to include notifying online systems of the new card and provisioning within their systems. One way of mitigating this problem is replacing the PIN requirement with another form of authentication. This can be accomplished with a biometric authentication in addition to the card. Instead of a user typing in a PIN, he or she provides a fingerprint image that is authenticated against the card, and then the terminal can provide authentication that is based on having the smartcard and the fingerprint [7]. This removes the PIN reset problem and introduces a smaller problem of users who have accidents with their finger. However, this also increases cost of equipment due to a fingerprint reader in addition to the card reader and card. It also increases administration, since a fingerprint is scanned in and saved to the card instead of a simple PIN number. Although the smartcard is a secure system, it comes with a lot of overhead, and may be a viable system for the future, but it is not one for worldwide use for all users today.

E. Biometrics

Biometrics is a method of identifying an individual based on physiological and/or behavioral characteristics like a fingerprint or mouse movements [8]. Due to the differences in all human beings, if a computer can mathematically model these differences, the user can then be identified and validated by a system. One of the great benefits of biometric authentication is the user does not have to remember any kind of PIN or password. There are many types of biometric means of authentication, to include, facial, facial thermogram, retinal scan, hand geometry, iris, written signature, voice print, and even thought processing of brainwaves [8], [14]. The typical biometric authentication system will extract features from a biometric sensor, and save it to a template database during a training or enrollment phase. Then for authentication, the features are extracted again, and matched to the features previously saved in the template database. If our bodies never changed, and we used the system the exact same way, biometric authentication systems would be perfect; however small changes may in fact create a false match or nonmatch. The system sensitivity for a match or nonmatch is calibrated based on its purpose. For example, a system that needs high security will have a low false match rate, which reduces the tolerance of the system. The overhead of the system rejecting valid users may be acceptable due to the sensitivity of the information [8]. Therefore, a system like the city water department may be able to run with far less tolerance, since it is more important to authenticate valid users than prevent attackers from compromising the system.

Using a biometric system for online transactions also creates another issue of transporting the original biometric features from a valid user to the server. If the system has a local office near the user, this presents no issue. For example, when users apply for their checking account they will provide their biometric identity in person. However for an online system, where the company and users are geographically separated, the users would need a method of transmitting their biometric identity to the remote system. This could easily be resolved if users maintained a public key, and stored the private key on a device that used their biometric system to access.

A method of resolving this problem is ZKPK, or Zero Knowledge Proof of Knowledge [17]. The overall concept is proving to a server that biometric authentication has taken place without sending the biometric information to that server. This allows the biometric data to be stored securely at the client, yet it is still useful for authentication. In order to do this, there are a few processes that must occur to securely validate the user and provide the proof without sending the biometric data [17].

In Bhargav-Spantzel's multi-factor biometric authentication system, ZKPK is combined with a random number to provide a multi-factor biometric authentication. Initially, a user must be enrolled into the system. During this time, there are several samples taken from the biometric measurement, and features are extracted and stored in a template database. Once this is performed, a random number is also generated. These two keys satisfy two types of authentication: what you have and what you are. It also preserves the user's biometric identity using the ZKPK technique. After enrollment, authentication is performed using some of the same functions. The system starts with the biometric input and then feature extraction. Once the features are extracted, a biometric key is created from the features. This key is then compared within a pre defined bio key space. An important part of the key space and formula used to generate the key is that weight is given to unique features of an individual to cause the key to change more than those of more common features. This is important to reduce false

positives or negatives. The closest key to the bio key is found using vector math, and the key is used in the next step. This key combined with a very large random number picked at the time of enrollment is sent to the server for authentication [17]. One of the added benefits to this form of authentication is if either key (the biometric or random number) gets compromised, the system is still not compromised. In addition, the separation of the biometric data and the biometric key ensures that if the biometric key is compromised, the biometric features are not. This is important since biometric features cannot be changed easily.

Although these biometric authentication systems can be very secure, there still exists the problem of initial enrollment and setup of these systems. Numerous entities exist for an individual to have to authenticate, and setting up identification means with each individual entity would be time consuming, and difficult to manage. There must be a way that multiple systems can use one centralized authority for authentication.

F. Centralized Authentication

An approach that Microsoft has taken to reduce complexity with managing passwords is to provide a central authentication server that manages a password for the user [9]. Since Microsoft provides various services from email to music and collaboration services, they have created the Passport Network. The Passport Network is a collection of services that share an authentication server. This solves the problem of users remembering multiple passwords, and may allow a user to create a more complex password, since they only need to remember one [9]. However, not all systems implement authentication through Microsoft Passport, so once a user has an account with a service that does not support the Passport service, they cannot be authenticated. The system could be implemented across various systems, but a lot of coordination would have to take place, and there would most likely be a legal aspect of Microsoft owning all passwords that would cause a great deal of issues. Another form of this kind of authentication was created at The University of New Hampshire. They integrated Microsoft Active Directory with their Unix systems through some scripts, which allows a single password for multiple systems, but in this case, it only works for people using accounts within the university [10]. Another way to store passwords for multiple disparate servers would be through a government service, but there may be issues with authenticating with servers outside of the country. Maintaining a worldwide server is probably fundamentally unsafe and infeasible. There is one proposal of Identity Providers (IdPs) that allow other providers to validate users based upon a database of multiple attributes. Registration with the Identity Provider would be in-person verification, and an ability to submit more attributes to the identity as needed [17]. This system is similar to the system examined above with the universal smart card, and would face some of the same difficulties. However, it would be easier to begin this implementation since attributes could be added as needed, and they would cost much less. This may well be a solution in the future if implemented correctly. One fundamental drawback to this system is if a server is compromised and it stores multiple accounts for multiple users, the result could be catastrophic. In addition, a server of this kind would be very likely to encounter numerous consistent attacks, to include denial of service. Once this occurs, users will not be likely to use a single server for multiple account access. This methodology may be viable in the future when cost of implementation is reduced.

G. Centralized Local Password Storage

A different scheme for authentication is locally storing multiple passwords for multiple systems in one local encrypted database that requires only one form of authentication to access all the passwords. This method solves the problem of a user trying to remember multiple passwords, because no matter how many accounts the users have to remember, he or she only needs to know one to authenticate themselves with the password manager. Even if a user is required to use a very complex password, this is far easier for a user to remember than multiple semi-complex passwords. The method for authentication with this database of passwords is important, and the encryption of the database is extremely important. Ideally, this system would store all the passwords in an encryption method equal to or stronger than the strongest system they are used to access. For example, if the most secure system a user uses is a bank that stores all users' passwords with a 128 bit encryption algorithm, the local user passwords should all be encrypted with a 128 bit or greater encryption. Therefore the amount of time it takes an attacker to break their passwords is the same as if they were attacking the actual system the passwords were intended for.

An ideal version of this password storing scheme would be a separate computer with a smart card and biometric authentication that is not connected to the network, and would automatically create and store the maximum complex password for each system the user needs. This is an extreme, and would be quite costly and inconvenient to the user. However, there is a practical system that makes it far easier to manage multiple passwords, and for the majority of users, would actually make it easier for them to operate on multiple accounts while using strong passwords. It is called the Password Multiplier, developed by J. Alex Halderman [11]. This is an add-on to the Firefox browser. The add-on creates an initial hash for each user that includes his or her email address, and takes approximately 100 seconds to generate [11]. Due to the computational complexity, a brute force attack would require the same length of time per iteration, making it computationally infeasible to crack before most or all the passwords it contained expired. Therefore, the

majority of users could accept the risk of storing the database on the computer they use daily to access the Internet. Another program called Passpet has been introduced to perform a similar function as the Password Multiplier, but also makes it easier for a user to login to an account [12]. The user merely needs to click on the Passpet icon, and the user is shown a "pet name" for the account like "my bank", and the web form is filled in with the user id and password. A system like this makes it even easier to login than using an easy password, while allowing for strong passwords. It also helps protect the user from phishing, where a bogus site is created to prompt a user for a username and password, and it saves these values for the attacker [12].

These methods seem like they will work for users, but there are also problems that must be addressed. First, users must validate that these programs do not have back doors. A program of this sort should go through scrutiny like an encryption algorithm, and should be completely open source in order to be scrutinized by all. Once a program of this type is widely spread, servers requiring authentication should integrate seamlessly with these clients, because users tend to do what is easy. If the user is given the choice to create a 12 character password with upper/lower case, numbers, and symbols, or use a manager to do this for them, they will likely choose the latter since it is easier. An advantage to an open source solution is that the software will also be free, which makes it even easier to distribute and promote. Unfortunately, the methods that are currently built into the common web browsers today (Internet Explorer and Firefox), cannot generate passwords, and by default store them without a user entering a password to retrieve them. Although both of these browsers use 3DES symmetric key encryption, the implementation is the flaw. The attacker only needs to be logged in as that user, or impersonate the user in order to make Application Programming Interface (API) calls to retrieve the passwords. Furthermore, there are various Javascript attacks that a hostile web proxy can use to make the browser reveal the passwords over the network. Although Firefox has enabled the option to password protect the master password file, this doesn't solve the problem of the common user who wants to use the system, but not be hassled by security overhead.

H. Behavioral Based Authentication

This is a form of authentication where the user's behavior at the terminal is used for authentication. In a sense, this form of authentication crosses the line between what a person is, and what a person knows, and is not used for initial authentication, but instead used to maintain continuous authentication. If the user has trained a system previously, the user can perform specific functions in order to provide authentication. These specific actions combined with natural behavior provide a correlation of what a person knows, and how he or she behaves. Research in this area is typically with a layered authentication. For example, once a user is authenticated with another means, the behavior is monitored to ensure the user has not changed. As more behavior is captured, the easier it is to validate the behavior.

One study conducted found that behavior authentication can be performed with minimal error for mouse movements on a single application; however, keyboard behavior did not perform nearly as well [16]. There is not a great deal of research in this area due to the fact that it is not suitable for a primary authentication means. However as systems get more complex, this may be viable for a layered authentication approach. Behavior authentication is comparable to biometrics, because it does not require the user to remember anything specific, or physically carry a smartcard or key that gives them access to the system. It also is advantageous over biometrics, because behavior authentication is not likely to require additional hardware to implement. Unfortunately it cannot be used as primary authentication due to the requirement to track multiple user actions over time instead of the instantaneous authentication that a password or key can provide. Therefore this method is only reasonable to use in important systems where the potential for an attacker to assume a session of another user is possible, and there is another form of primary authentication that does the initial validation of the user.

I. Authentication Recovery

In all the systems and schemes mentioned thus far, except biometric, there exists one more problem: recovery when an authentication token is lost. The most basic form taken by websites is either asking one or two questions unique to that user, or sending a password reset link to the user via the user's email address. Although these methods usually are fast and work well, they can be points of failure for security. The security question concept in particular is usually not implemented well, and anyone with knowledge of another person's favorite colors, pets, and/or mother's maiden name will be able to bypass this easily. As information becomes more readily available on the Internet, the questions must be extremely obscure to ensure only the individual knows the answer to them. The advent of blogs has made this more difficult, since it is more common for people to store their personal information on the Internet. The other common solution of emailing the password works, but is typically not implemented securely. If the site can email the actual password, then the site isn't storing it in a one way encryption algorithm, thus it is inherently insecure. If the site sends the password, but doesn't require a change at next logon, then the password is compromised, since the email is sent in plaintext. Ideally, the site should change the password, email the new complex password, and request change after the user logs in. In addition, if the user does not do so within a small period of

Figure 1: Sample Authentication Matrix

Security							
	Very High (Smartcard or Multi Modal Biometric)						X
	High (Strong Password >= 10 letter/ number/symbol combinations)			X	X	X	
	Medium (8 character password)		X				
	Low (Less than 8 characters)	X					
		Utility Bill	Auction Account	Email	Checking Account or Investment Accounts	Online Bill Pay Service	Financial Account able to move > $10,000
				Account Type			

time, the password must be changed again to prevent an attacker from using the new password after seeing the plaintext email. In most systems today, this method is not only widely used, but probably acceptable. If the account compromise will not result in financial loss, then just basic authentication means will work for the common user. However, if account access gives the user the ability to transfer funds or pay bills, the password recovery should be much stronger. In these cases, the common method is to either have the password or PIN physically mailed through the postal service, or validated on the phone with multiple security questions. Security questions asked during a phone call can be fairly secure, since the operator can potentially detect hesitations on data, and ask further questions if the user isn't properly authenticated. Ideally, the questions would also be varied in order to prevent an attacker from determining the questions beforehand so he or she can gather the data before the phone call. Obviously, there needs to be a better way to validate authentication when the primary authentication fails.

In the authentication database system, or Identity Provider, there could potentially be multiple attributes stored, which could be used to further authenticate an individual. However, if there is no database set up for this, another means of emergency authentication must be used. One method proposed is authentication based on someone you know. Most authentication methods can be categorized by something you know, have, or are, but in this system, it is based on another person. The concept is based on an everyday process: when you meet someone new, you are typically introduced to that person by someone you know. In effect, the person you know is a mutual party who performs an authentication based on trust. The same concept can be applied to computer authentication. In this method, a trusted person can log into the system, request a temporary password for the person who lost his or her authentication means, and give that person the temporary password. Then the individual can use the temporary key to get into the system and update his or her password [14]. This method is more secure, but it is obviously more complex and time consuming, especially if the trusted party is not available to perform this kind of authentication. It also must be implemented correctly so that the trusted party cannot take over the account, but merely help the user reset the password through the system.

IV. CONCLUSION

There are many advantages and disadvantages for different mechanisms of authentication; however it still lies in the users' and administrators' hands to solve these problems. A user may never be able to perform perfect 100% identification all the time, but as the technology progresses and identity theft increases, users must be more aware of their own security measures. Poor authentication mechanisms can easily lead to a form of identity theft called "account takeover", in which an attacker gains enough information about a user to impersonate them on his or her existing accounts [13]. The risk of an account takeover can be reduced by increasing user awareness, and providing sensible means for authentication. Currently there is no one solution to fix everyone's authentication problems, however the problem should be viewed the same way encryption is viewed. For a good encryption system, the strength of the password should be high enough so that if an attacker brute force attacks a system, the computational time it will take to crack the password is longer than the life of the password, or the information encrypted is no longer required to be kept secret. Authentication is similar: a user should categorize his or her accounts based on the impact of an attacker performing an account takeover.

In Figure 1, samples of various accounts one user may have are categorized into different security levels. In the example, low is a password less than 8 characters, medium is 8 characters with numbers or symbols, high is 10 characters with upper and lowercase and numbers or symbols, and very high is a hardware device. Biometric devices can be used at any of these levels, depending on the implementation and quality of the device. In some cases, the biometric device may be of low quality (e.g. simple fingerprint reader), or have a high tolerance. In other cases, it may have no tolerance or could be of high quality (e.g. multimodal), and a variation in the user's body may cause a false negative. Since biometric devices vary, they would have to be tested individually to determine the level they could operate at. The accounts are arranged based on the impact of impersonation for the account type. In general, utility bill accounts are not a significant security risk, since a compromise would probably not result in any monetary loss. An online auction account could present

some problems if the attacker bids on items, but the user may determine this since an email is usually sent once a bid is placed. However, an email address could result in account compromise for numerous accounts, since email is usually used for password recovery. Therefore an email account needs to be placed next to the security of all other accounts that use email for password recovery. Any kind of bill paying account should also use a high authentication, since the attacker may use it to withdraw money out of a user's account. Any account that can result in monetary loss of greater than $10,000 if compromised should be accessed only with a hardware device (i.e. smart card, DOD certified biometric device, etc). Some companies may use a telephone call-back when performing such a task, and this may be acceptable in place of hardware devices for authentication if implemented correctly. If one of these authentication means is not used, the account should be limited on how many funds it can withdraw/move.

An administrator must look at the system from a user's viewpoint of authentication and from an attacker's perspective. The administrator cannot keep a user from writing down a password, but he or she can ensure that if one user's account is compromised, it does not compromise the security of the system (e.g. privilege escalation), and means to detect unusual user activity are in place. The administrator should also consider the consequences of an account takeover, and base the authentication scheme on the value of the highest risk account in the system. For example, if a bank allows online money transfers, and none of its account holders have more than $10,000, then an acceptable means may be strong passwords. If another bank has account holders that have more than $10,000 accessible for transfer online, then they should require a hardware authentication means for those accounts (or disallow transfers of that amount), and possibly provide it as an option for all other accounts.

Security is ultimately up to each user. An administrator can only set policy for his or her system, but the users are responsible for the security of their accounts on the system. In addition, administrators must always look at their system from the user's viewpoint to understand if their system encourages good security practices, or not. This is essential when administering a system, because the security of that system is only as good as the least secure user of that system.

V. FUTURE WORK

Many methods of authentication exist today, and one solution doesn't fit all problems. As technology increases and cost of hardware decreases, more authentication options should be examined and implemented. If better security options are available to the user, the security of all systems will benefit, especially if the secure options also provide convenience of use. In addition, the design of future authentication systems should always keep the user in mind. A very difficult to use authentication system may be suitable for a high security government operation, but for the average user it is useless. Future systems should be designed with authentication mechanisms in mind. Future authentication systems for the common user must be easy to use, extremely difficult to impersonate, and usable in insecure networked environments. Potentially, an authentication system could be combined with the server protection to provide unique secure access. A technique called port knocking, where a user requests a series of closed ports in order to open an access port, could be modified to provide authentication [18]. Each user would have a unique port sequence, which would allow access and authenticate the user. However once a secure connection is established, the sequence for the next connection must be negotiated, since the port knocking is performed in the clear.

REFERENCES

[1] A. Jain and S. Pankanti, "A touch of money," in *IEEE Spectrum*, Vol 43 Issue 7, July 2006. 22-27.
[2] R. Hills, "Nortel VP Client Issue: Clear-text password stored in memory". http://www.securityfocus.com/archive/1/393943/2005-03-20/2005-03-26/0
[3] B. Schneier, "Sensible Authentication," in *ACM Queue*, Feb. 04. 75-78.
[4] Schechter, Stuart, et al. "The Emperor's New Security Indicators: An evaluation of website authentication and the effect of role playing on usability studies." http://usablesecurity.org/emperor/
[5] P. C. Clark and L. J. Hoffman, "BITS: A Smartcard Protected Operating System," in *Communications of the ACM* 37, No 11(Nov) 66~94.
[6] J. Dray, M. Smid, R. Warner, "Implementing An Access Control System with Smart Token Technology," *National Institute of Standards and Technology* April 12, 1989.
[7] C.T. Clancy, N. Kiyavash, and D. J. Lin, "Secure Smartcard-Based Fingerprint Authentication," in *Workshop on Biometric Methods and Applications* 7 November, 2003.
[8] A. Jain, L. Hong, and S. Pankanti. "Biometric Identification," in *Communications of the ACM* 43 No. 2 (Feb) 91-98.
[9] Microsoft Corporation www.microsoft.com www.passport.net.
[10] D. J. Blezard and J. Marceau. "One User, One Password: Integrating Unix Accounts and Active Directory." In *SIGUCCS* 2002, November 20-23, Providence, Rhode Island 5-8.
[11] J. A. Halderman, B. Waters, E. W. Felten. "A Convenient Method for Securely Managing Passwords," in the *World Wide Web Conference Committee*, May 10-14 2005, Chiba, Japan. 471-479.
[12] K. Yee and K Sitaker, "Passpet: Convenient Password Management and Phishing Protection," in the *Symposium on Usable Privacy and Security* July 12-14, 2006 Pittsburgh, PA.
[13] A. K. Abdullah, "Protecting Your Good Name: Identity Theft and its Prevention." In *InfoSecCD Conference* 2004, Oct. 8th, Kennesaw, GA.
[14] J. Brainard, A. Juels, R. L. Rivest, M. Szydlo, and M Yung, "Fourth Factor Authentication: Somebody You Know," in *CCS* '2006 October 30-November 3, 2006, Alexandria, VA.
[15] J. Thorpe, P. C. Oorschot, A. Somayaji, "Pass-thoughts: Authenticating with Our Minds," in *NSPW* 2005 Lake Arrowhead, CA.
[16] M. Pusara and C. E. Brodley, "User Re-Authentication via Mouse Movements," in *VizSEC/DMSEC 2004*, Oct. 29, 2004. Wash., DC.
[17] A. Bhargav-Spantzel, A. Squicciarini, and B. Elisa, "Privacy Preserving Multi-Factor Authentication with Biometrics," in *DIM* November 3, 2006, Alexandria Virginia. 63-71.
[18] Christan Borss (2001) Listserv post to Braunschweiger Linux User Group (*lug-bs@lk.etc.tu-bs.de*)