

Password Management Best Practices



Contents

- 1 Introduction** **1**

- 2 User authentication and passwords** **2**
 - 2.1 Definitions 2
 - 2.2 Authentication technologies 2

- 3 Security threats** **4**

- 4 Human factors** **5**

- 5 Composing hard-to-guess passwords** **6**

- 6 Unicode and other non-Latin passwords** **8**

- 7 Changing and reusing passwords** **9**

- 8 Keeping passwords secret** **10**

- 9 Intruder detection and lockout** **11**

- 10 Encrypting passwords in storage and transit** **12**

- 11 Synchronizing passwords** **13**

- 12 Single signon** **14**
 - 12.1 Taxonomy of SSO Systems 14
 - 12.2 Security of SSO Systems 14

- 13 IT support for forgotten and locked out passwords** **16**
 - 13.1 Security challenges 16
 - 13.2 Mitigating controls 17
 - 13.3 User authentication 17

- APPENDICES** **18**


- A Sample Security Questions** **19**

A.1	Questions with textual answers	19
A.2	Questions with numeric answers	20
B	References	21

1 Introduction

This document describes and justifies password management best practices as applied in medium to large organizations. It offers reasoned guidance to IT decision makers when they set security policy and design network infrastructure that includes passwords.

The guidance in this document is focused on how to best manage user passwords. It is not intended to address the special challenges and techniques that arise when managing privileged passwords, that belong to administrators, service accounts, etc.

Look for the  marks throughout this document to find best practices.

2 User authentication and passwords

2.1 Definitions

- **Identification**

The unique identifier that a user types to sign into a system or application is that user's login ID on that system.

- **Authentication**

Authentication is a process by which a user proves his identity to a system – normally when logging in.

- **Authentication factor**

An authentication factor is something a user presents to a system in order to prove his identity. It may be something he (and hopefully only he) knows, or proof of possession of a physical object, or a measurement of some physical characteristic (biometric) of the living human user. In other words, something the user knows, or something he has, or something he is.

- **Multi-factor authentication**

Multi-factor authentication means authentication using multiple factors . For example, a user might sign into a system with a combination of two things he knows, or a combination of something he knows and something he has, or perhaps something he knows, something he has and something he is.

The premise is that adding authentication factors makes it more difficult for a would-be attacker to simulate a legitimate authentication and consequently impersonate a legitimate user.

- **Strong authentication**

Strong authentication refers to an authentication process which is difficult to simulate. It may be based on use of multiple authentication factors or use of a single but hard-to-spoof authentication factor .

2.2 Authentication technologies

Authentication technologies may be categorized as follows:

Authentication Factor	Description	Example
Secrets: Something you know.	Secret information known only the user.	A password or PIN.
Tokens: Something you have.	A physical device possessed only be the user.	A one time password (OTP) token or smart card.
Biometrics: Something you are.	A unique, measurable characteristic of the user.	Voice print verification, fingerprint, vein pattern, retina or iris scan.

Generally, it is more secure but less convenient to use multiple forms of authentication – e.g., a one time password token combined with a password or PIN.

These types of authentication may be further characterized as follows:

Characteristic	Secrets	Tokens	Biometrics
Reliable identification?	Good	Very good	Excellent
Requires client-side hardware	No	Sometimes	Yes
Requires client-side software	No	Sometimes	Yes
Typical deployment cost/user	0	\$50	\$100
Works with legacy systems	Yes	No	No

Due to cost and compatibility with legacy systems, the most popular form of user authentication continues to be a secret password.

Moreover, as non-password authentication technologies are deployed, they are usually combined with passwords to improve their security. For example, tokens are secure – unless they are stolen, in which case the person who stole the token can impersonate the legitimate user. This is easily remedied by requiring the user to sign into systems with the OTP displayed on a token plus a password or PIN, which a thief would not know.

The remainder of this document discusses how to best manage passwords, to maximize security and minimize cost of ownership. The guidelines here apply primarily to passwords used as a single authentication factor, but can also be applied to passwords used as a second authentication factor, in conjunction with biometrics, tokens or smart cards.

3 Security threats

Passwords are simply secret words or phrases. They can be compromised in many ways:

- Users may write them down or share them, so that they are no longer really secret.
- Passwords can be guessed, either by a person or a program designed to try many possibilities in rapid succession.
- Passwords may be transmitted over a network either in plaintext or encoded in a way which can be readily converted back to plaintext.
- Passwords may be stored on a workstation, server or backup media in plaintext or encoded in a way which can be readily converted back to plaintext.

Each of these vulnerabilities make it easier for someone to acquire the password value and consequently pose as the user whose identity the password protects.

Conversely, if passwords are managed securely by users and if password systems are constructed so as to prevent brute-force attacks and inspection or decryption of passwords in transit and in storage, then passwords can actually be quite secure. This document will describe some of the mechanisms for securing passwords.

4 Human factors

Users in a large organization frequently have many passwords, each protecting their account on a different system or application. Users are people, not machines, so their ability to securely manage passwords is intrinsically limited. In particular, it is hard for most people to remember:

- Complicated passwords.
- Many different passwords.
- Passwords that change frequently.
- Passwords for systems that are used infrequently.

Without these constraints, password security would not be a problem and there would be no market for other authentication technologies. For example, if every password was changed every day, was remembered perfectly without being written down and consisted of 100 randomly chosen letters and digits, there would really be no need for authentication technologies other than passwords.

Effective password management is therefore taken to mean password management that is secure, user friendly and supportable within the confines of both technology and human behavior.

When people have trouble remembering their passwords, they usually resort to one or more of the following:

- Write down their passwords – and reduce the security of systems to the security of their physical building, desk or wallet.
- Forget their passwords – and require frequent assistance from an IT help desk to reset it.
- Use very simple, easily compromised passwords.
- Reuse old passwords as often as possible.

Clearly, sound password management practices must take into consideration human limitations, to reduce the frequency of these insecure practices.



5 Composing hard-to-guess passwords

As outlined in Section 3 on Page 4, one of the weaknesses of passwords is that they can be guessed. While a human may give up after ten or a hundred guesses, software is available to automate the guessing process and try millions of possible passwords per second:

- L0phtCrack: <http://www.l0phtcrack.com/> for Windows.
- John the Ripper: <http://www.openwall.com/john/> for Windows, Mac OS X or Linux.
- Many others.

To combat password guessing attack, users should pick hard-to-guess passwords. One way to do this is to ensure that the set of all possible passwords is too large to search thoroughly and then to eliminate probable guesses, which an attacker might try first.

The number of possible password combinations is calculated by taking the number of legal characters in a password and raising that number to the number of characters in the password. The possibilities for some typical combinations are shown below:

Character set	Password length					
	5	6	7	8	9	10
0-9	1.00e05	1.00e06	1.00e07	1.00e08	1.00e09	1.00e10
a-z	1.19e07	3.09e08	8.03e09	2.09e11	5.43e12	1.41e14
a-z,0-9	6.05e07	2.18e09	7.84e10	2.82e12	1.02e14	3.66e15
a-z,0-9,3 punct	9.02e07	3.52e09	1.37e11	5.35e12	2.09e14	8.14e15
a-z,A-Z	3.80e08	1.98e10	1.03e12	5.35e13	2.78e15	1.45e17
a-z,A-Z,0-9	9.16e08	5.68e10	3.52e12	2.18e14	1.35e16	8.39e17
a-z,A-Z,0-9,32 punct	7.34e09	6.90e11	6.48e13	6.10e15	5.73e17	5.39e19

Users should choose their passwords from the widest possible set of characters, subject to the constraints of the systems where those passwords reside. For example, most mainframes do not distinguish between uppercase and lowercase and only allow three punctuation marks (fourth row in the table above).

The policy should be based on the smallest acceptable set of possible password values. This, in turn, depends on how fast an attacker can check if a possible password is valid; how much time an attacker has to attack a given password and how log we want the attacker's probability of success to be.

This is best illustrated with an example:

1. Assumptions:

- (a) Encrypted passwords are not available to an attacker and consequently passwords can only be guessed at a rate of 100/second (this number is high / conservative for an over-the-network attack).

- (b) An attacker can make an unlimited number of guesses without being detected (this is easy to fix).
 - (c) Passwords are changed, at most, every 90 days.
2. Objective: the attacker's probability of success should be no more than 1%.
 3. Technical constraints: passwords may only contain lowercase letters, uppercase letters and digits.
 4. Calculation:
 - (a) An attacker can make $90 \times 24 \times 60 \times 60 \times 100 = 777,600,000$ guesses before a password will expire.
 - (b) There should be 77,760,000,000 possible passwords – about $8e10$.
 - (c) From the above table, a 7 character password is sufficient, so long as users don't pick easily guessed values (e.g., their name or a dictionary word).

In practice, assuming that encrypted passwords are not available to an attacker and that passwords cannot be captured or decrypted in transit, the following policy is a often reasonable:



- To ensure that the search space is sufficiently large:
 - Passwords must be at least seven characters long.
 - Passwords must contain at least one lowercase letter, at least one uppercase letter and at least one digit.
 - If technically possible, passwords must contain at least one punctuation mark, so long as there are many (10 or more) available punctuation marks.
- To eliminate trivial passwords, passwords should not:
 - Contain the user's name or login ID.
 - Contain a dictionary word, in any language that users can reasonably be expected to know.
 - More than two paired letters (e.g. abbcde is valid, but abbcdd is not).

Some of these rules merit further discussion, since a naive implementation of the rule can actually reduce password security. Consider the rule “password should not contain a dictionary word.” The letters “a” and “l” are legitimate English words, and if the rule is enforced using a dictionary that contains these, then in effect we have reduced the set of available letters from 26 to 24. To avoid this sort of problem, rules of the form “shall not contain” should be fine-tuned:

- Ignore words with 3 or fewer characters, since including such words in a dictionary check may actually make passwords easier to guess, by eliminating too many legitimate password values.
- Consider dictionary words without regard to case. “Hello12” should not be a valid password, even if the dictionary contains “hello” but not “Hello”.



6 Unicode and other non-Latin passwords

Most password composition rules focus on how passwords composed of Latin characters (i.e., “English language” letters), digits and punctuation marks are composed.

Unfortunately, English is the native language of no more than 400 million people and is spoken by no more than 1.5 billion people – out of a global population of over 6.5 billion people. In other words, even if other European languages that use a similar character set are included, a focus on Latin characters ignores at least 2/3 of the world’s population.

This not just an academic problem. Many organizations are global in nature and include users who speak a variety of languages. A system or application whose users reside around the world needs to enforce a password policy that makes sense for all of them, not just for those who happen to prefer English.



Password management for multi-lingual users is a broad subject, but following are some important considerations:

1. Most users, in most countries, have keyboards that include most Latin characters. This means that they are able to type passwords composed of “English” letters, even if they don’t read or write the language.
2. Not every system or application supports non-Latin characters in the password field. When composing an enterprise password policy, it’s important to consider the limitations on the password fields of in-scope systems and applications. Failing to do this can yield a policy that is relevant to only some systems.
3. In countries where Chinese, Japanese or Korean (CJK) is spoken, text input often involves a display component. For example, a Chinese user might type the pinyin representation of a character and the input software will display a list of matching characters. The user will then choose the desired character and start on the next character.

This pattern is normal for input of text in the CJK languages but is problematic for passwords since password input should not be visible to other people in the physical vicinity.

Most CJK users avoid this problem by choosing passwords that consist only of Latin characters – something that may surprise native English speakers.

Some best practices follow from this discussion:


1. Encourage or require users to restrict their passwords to Latin characters, both for compatibility and to avoid input methods which display the characters users type. 
2. Do use a dictionary lookup to ensure that new passwords are hard to guess, and do include dictionaries of non-English words represented in Latin character sets (e.g., Pinyin, etc.). 


7 Changing and reusing passwords

Over time, passwords may be compromised in many ways, including:

- Users may share them with friends or coworkers.
- Users may write them down and they may subsequently be exposed.
- Passwords may be guessed, either by humans or software.
- The servers that store passwords may be compromised and their passwords acquired by an intruder (encrypted, hashed or plaintext).
- The networks over which passwords travel, between a user's workstation and the system into which the user signs in may be compromised, again leading to compromise of a password.
- Users may be tricked into revealing their passwords (phishing).
- The help desk may be tricked into giving an intruder a valid password (social engineering).


To limit the usefulness of passwords that have been compromised, a best practice is to change them regularly. A common rule in many organizations is to force users to change their passwords when they log in, every 60 or 90 days.

In general, users should be required to change their passwords regularly. The password expiry interval should not be longer than 90 days. 

For the same reasons, users should not reuse old passwords, as they may already have been compromised. Many systems support this by recording some representation of old passwords and ensuring that users cannot change their password back to a previously used value. 

When password history is enforced, users may figure out the number of passwords stored for this purpose. As this number is usually 10 or fewer, a user who would prefer to keep his password the same may make a series of consecutive password changes where the final password value is the same as the first one.

To defeat such “smart” users, some systems also enforce a password rule that limits the number of password changes that a user can make in any given day. By making the process of defeating password history take several days, users are discouraged from trying to reuse old passwords.


A better approach, though not available on all systems, is to simply maintain an unlimited number of entries in each user's password history. Since disk storage is inexpensive, this approach is practical on modern systems. With this approach, users can change their passwords as often as they like – and still never be able to reuse old ones. 


8 Keeping passwords secret


Passwords are intended to reliably differentiate between the authentic user and impostors. As such, they must be secrets – known only to the user they identify.

Users frequently behave in ways that lead to password disclosure. In particular, users may:

- Choose passwords which are easily guessed – so are not really secret.
- Share their passwords with coworkers, friends or family.
- Write down their passwords and place the written password near their computer or in a supposedly private place like a wallet.

A comprehensive password policy should explicitly forbid these behaviors and specify negative consequences to users who violate the policy. 

To help users comply with an effective password policy, user friendly password management tools and processes should be provided. Key among such aids to users are password synchronization and single signon. 

With either synchronized or automatically filled-in passwords, users have fewer passwords to remember so they are less likely to write down their passwords. 

9 Intruder detection and lockout

Many systems can detect repeated attempts to sign into an account with incorrect passwords. A pattern of failed logins may be due to:

1. Typing problems. For example, the user may have the Caps Lock or Num Lock key enabled and not realize it.
2. A forgotten password.
3. The wrong password being used. For example, the user may be trying to sign into system A using the password for system B.
4. An intruder trying to sign into an account by guessing the legitimate user's password.

To deter the last of these, it's reasonable for a system that detects too many invalid attempts during a short period of time to lock out the user's account and prevent further attempts, at least for a while. This is called an intruder lockout.

Many systems include an "intruder lockout" feature. If N invalid attempts to login to the same account are made during an interval of T_{detect} minutes, then the account is disabled for $T_{lockout}$ minutes.

While this mechanism is effective against concerted attacks against a single account, it does nothing to prevent an intruder from simultaneously trying to guess the passwords of many users. A more sophisticated mechanism might extend this as follows: If M invalid login attempts are made from network address A , to any account, during an interval of T_{detect} minutes, then the address A is disabled for $T_{lockout}$ minutes.

This will deter intruders with access to just a single network device/address from which to carry out their attack.

Intruder lockout also has a down side – it can be used to carry out a denial of service – by systematically locking out many accounts, including administrator IDs.

Because of these fundamental limitations, the best practice is to combine several intruder lockout-related policies:

- Apply only to users, exempting at least one administrator login. This limits the scope of denial of service attacks.
- A compensating control is required here, such as requiring administrator IDs to have very complex passwords that are changed more frequently than user passwords. This way, they cannot be guessed (too hard) or locked out (feature not available).
- Apply a high threshold for intruder detection – for example, 10 failed login attempts in 5 minutes. This prevents spurious lockouts caused by users who have trouble remembering or typing their own passwords.
- Automatically and quickly clear lockouts – for example, after 10 or 15 minutes. This reduces the impact on legitimate users while continuing to significantly impair the ability of intruders to make bulk password guessing attacks.



10 Encrypting passwords in storage and transit

Passwords may be stored on user workstations or servers. They must be transmitted, in some form, from the user's workstation to a server when the user signs into systems and applications.

Most organizations are powerless to improve the manner in which existing infrastructure uses encryption to protect passwords. However, staff responsible for developing a new applications that include password authentication can and should take effective precautions, such as storing passwords using a well-known and trusted hashing algorithm such as SHA-256 or (somewhat less effective, but still infeasible to find a collision in 2010) SHA-1.



Passwords transmitted from a client device to a server are similarly subject to weaknesses in the protocol used by the relevant application. For example, Telnet and TN3270 are plaintext while SSH and SSL are strongly encrypted. Where password security is important and where one cannot trust the physical security of all communication media between a user and the systems he logs into, then additional mechanisms, such as IPSec should be considered to protect that communication.



Some client devices and applications cache passwords and automatically provide them to servers on behalf of users, as required. Cached passwords should also be cryptographically protected, but in this case encryption rather than hashing is required, since the password must be recoverable.



In some cases, the protocol provided by a vendor may encrypt passwords when they are used to login to a system, but not when a password change is transmitted. This happens with Oracle SQL*Net, for example. In these cases, if password management software is deployed, it is helpful for that software to implement its own encryption for password transmission, beyond that provided natively by the application vendor.



11 Synchronizing passwords

Password synchronization technology helps users to maintain the same password on two or more systems. This, in turn, makes it easier for users to remember their passwords, reducing the need to write down passwords or to call an IT help desk to request a new password, to replace a forgotten one.

There is some debate as to whether password synchronization makes systems more secure or less. The arguments are as follows:

- **Synchronization reduces security:**

If a single system is very insecure, then compromising that system will give an intruder the passwords for every other system which uses synchronized passwords.

This weakness is avoided by requiring that users employ unsynchronized passwords on such weak systems.

- **Synchronization improves security:**

Users with many passwords have trouble remembering them and consequently tend to write them down. System security is reduced to the security of a piece of paper – i.e., close to zero.

In practice, it is very difficult to get users who have many passwords to stop writing them down. The two arguments above therefore can be restated as:

- Synchronized passwords: as secure as the least secure system in scope.
- Unsynchronized passwords: as secure as slips of paper.

Since most systems and applications make at least some effort to protect their passwords, synchronized passwords are normally more secure. To mitigate the risk of a single system compromise being leveraged by an intruder into a network-wide attack, there are some password management guidelines to follow:



- Very insecure systems, such as those that use little or no cryptography to protect passwords, should not participate in a password synchronization system.
- Synchronized passwords should be changed regularly, on the assumption that either some in-scope systems are vulnerable or that passwords may be compromised through non-digital means (social engineering, etc.).
- Users should be required to select strong (hard to guess) passwords when synchronization is introduced.

The bottom line is that a single, hard-to-guess, regularly changing password is normally more secure than multiple passwords, some of which may be easy to guess, some of which may not be changed regularly and all of which may be written down.

12 Single signon

Single sign-on (SSO) is any technology that replaces multiple, independent system or application login prompts with a consolidated authentication process, so that users don't have to repeatedly sign in.

12.1 Taxonomy of SSO Systems

There are several types of single sign-on systems:

1. Enterprise SSO – application login prompts continue to ask users to type their login IDs and passwords, but software on the user's PC automatically identifies and fills in these dialogs.
2. Agent-based web SSO – agents on each web application check whether the incoming web browser (and by extension, its user) has already been authenticated. If so, no further authentication is required.
3. Proxy-based web SSO – web browsers access web applications through a proxy, which looks up user credentials and injects them into the data stream sent to each web application.
4. Kerberos – authentication is performed using a separate infrastructure, outside of other systems and applications. Users sign into the Kerberos infrastructure and their computer receives a token, which can be offered to Kerberos-enabled applications instead of filling in a login prompt.
5. Federation – similar to Kerberos, but designed to allow single signon between different organizations (administrative domains). Users sign into a system in one domain and when they try to access an application in another domain, a security assertion is forwarded from their source domain indicating who they are.

12.2 Security of SSO Systems

When organizations consider various approaches to single (or reduced) sign-on, they generally need to know:

- **Is it secure?**

The short answer, in general, is "yes."

The longer answer is more nuanced:

- *Strength of the initial authentication:*

SSO works by extending the reach of some initial authentication step. So long as that step is robust, subsequent application logins can also be trust worthy. In other words, if users sign into the SSO system with a strong password (as described elsewhere in this document) or using a two-factor technology such as OTP+PIN, Smart card+PIN or biometric+password, then the SSO infrastructure rests on a robust platform. On the other hand, if the SSO system depends on a weak password or PIN, on a proximity card, on a single physical authentication factor (which could be stolen) or on an easily recorded biometric, then the entire system is vulnerable.

– *Strength of stored credentials:*

Some of the SSO approaches described above depend on assertions, while others depend on the injection of login IDs and passwords into application login prompts. The latter set includes enterprise SSO and proxy-based web SSO.

Injection of application passwords depends on reliable, secure storage of those passwords. This raises the question of where and how these passwords are stored and how a user's SSO agent is authenticated before it is allowed to retrieve these passwords.

For example, if biometrics or proximity badges are used to authenticate users, how can a key be securely generated to decrypt application passwords? If a weak password is the primary means of authentication, what is to stop an attacker from guessing the encryption key that protects other passwords?

• **Is it more or less secure than password synchronization?**

The answer here is also nuanced, and depends on how users sign into the SSO system in the first place.

– *Users sign into SSO with a password:*

If password synchronization covers only secure systems, whose password databases have been protected against attack using strong encryption, good password composition rules, regular changes, history, intruder lockout, etc., then the security difference between SSO and synchronization is the difference between compromising a single robust password database (the one used to sign into the SSO system) and compromising any one of several such systems. i.e., infeasible in either case.

– *Users sign into SSO with a weak authentication factor:*

If users sign into the SSO system using a weak password, proximity badge or single physical authentication factor, then the energy that an attacker must expend to compromise all credentials is lower than what the attacker would have to expend to compromise one of several strong password systems.

– *Users sign into SSO with a strong authentication factor:*

If users sign into the SSO system using a two-factor technology or strong password, then the energy that an attacker must expend to compromise all credentials is at least as high as what the attacker would have to expend to compromise one of several strong password systems, with the possible exception of injected passwords (see enterprise SSO and proxy-based web SSO above) with a vulnerable encryption key management scheme.

13 IT support for forgotten and locked out passwords

Sometimes users forget their passwords or type the wrong one often enough to trigger an intruder lockout. When this happens, users generally call their IT help desk and ask for the lockout to be cleared or the password to be reset.

The help desk process normally proceeds as follows:

1. User experiences problems trying to log in.
2. User calls the help desk and may wait in a service queue.
3. Support analyst asks the user for his name and a problem description.
4. Support analyst asks the user for information to prove his identity (i.e., authentication of a human by a human).
5. Support analyst compares this information with some on-line resource.
6. If the user is authenticated, the support analyst will generally do one of three things:
 - (a) Reset the forgotten password.
 - (b) Clear the intruder lockout.
 - (c) Forward the support incident to another IT support person, who has the requisite privileges to perform one of the above functions.
7. The user tries to log in with the newly assigned password.
8. The user may be required to change the new password immediately.

13.1 Security challenges

The process described above may be vulnerable to security weaknesses:

- The IT support analyst may forget to authenticate the user at all.
- Information suitable for authenticating the caller may not be available to the IT support analyst.
- Information about caller may be easy for an attacker to guess, making the whole process vulnerable to “social engineering” attacks.
- Many IT staff need to be able to reset passwords and clear lockouts for this process to work effectively. This proliferation of privileged accounts (i.e., those of the IT staff) creates an unnecessarily large attack surface.
- Password resets on some systems may not be traceable back to individual IT support analyst. This creates a lack of accountability and allows IT analysts to compromise user accounts without consequences.

- The support analyst knows the user's new password at the end of the process, so it is no longer truly secret. This is especially problematic when newly reset passwords do not immediately expire.

13.2 Mitigating controls

These problems can be overcome through:

- IT support training, to ensure robust authentication of callers prior to every password reset.
- Ensuring that reliable authentication information is available for every user – including employees, contractors, etc.
- Providing an application proxy that resets passwords on behalf of IT support staff. This way, both IT staff and callers are authenticated before a password can be reset. This reduces the number of privileged accounts and ensures the process is consistent and secure.
- Allowing users to reset their own forgotten passwords and/or clear their own lockouts, after suitable non-password authentication – i.e., self-service password reset. This reduces the number of privileged accounts, enforces a robust authentication process and removes the situation where an IT support analyst knows the user's new password.

13.3 User authentication

It is important to ensure that users who request a password reset, either by calling the IT support help desk or using self-service, are reliably authenticated. Following are some practical considerations when designing a non-password authentication system:

- One-time-password tokens, smart cards and biometrics are generally more reliable than security questions, because of the vulnerability of security questions to social engineering attacks.
- Where security questions are used (for example, due to their lower cost or easy availability):
 - As many questions as possible should be used.
 - The difficulty of guessing the answer to each question should be estimated. The total difficulty of guessing enough answers to authenticate as a user should be approximately the same as the difficulty of guessing a user's password.
 - Users should provide answers to both personally-chosen and standardized questions. In the former, the questions are harder to guess but answers are sometimes easy to guess. In the latter, the questions are well known to would-be
 - A randomly chosen subset of a user's available security questions should be chosen, to make guessing harder for attackers with a limited set of information about a user.
 - Intruder lockout should apply to authentication with security questions just as it does to passwords.

Sample security questions are provided in [Appendix A](#) on Page 19.

APPENDICES

A Sample Security Questions

A.1 Questions with textual answers

Sample security questions, which may have alpha-numeric questions and so are suitable for a text user interface, include:

- Which bank branch do you live closest too?
- What car do you wish you owned?
- What is your favorite food?
- Who is your favorite book character?
- What is your favorite game or sport?
- What is your favorite movie?
- What is your favorite pizza topping?
- What is your favorite restaurant?
- What is your favorite season of the year?
- What is your favorite sports team?
- In which department did you first work?
- What was your first position in the company?
- What was your first car?
- Who is the person you admire the most?
- What was the most memorable day in your life?
- Who was your childhood hero?
- What is the nickname of your sibling?
- Who was your first boss?
- What award are you proudest of?
- What city were you born in?
- What is the farthest from home you have traveled?
- What is the name of the first school you attended?
- What is the name of the first person you were romantically interested in?
- What is your astrological sign?
- What is your father's middle name?
- What is your mother's' middle name?
- Who is your favorite actor, actress or celebrity?
- What is your favorite musical band?
- What is your favorite beverage?
- What is your favorite board game?
- Who is your favorite book character?
- What is your favorite dessert?
- What is your favorite hobby or pastime?
- What is your favorite ice cream topping?
- What is your favorite song?
- What is your favorite television show?
- What is your favorite vacation spot?
- What is your mother's maiden name?
- What is your place of birth?
- What is your school team's mascot name?
- What was the breed of your first pet?
- What was the color of your first automobile?

- What were the make and model of your first car?
- What was the name of a favorite childhood pet?
- What was the name of your first girlfriend/boyfriend?
- What was the street name of your childhood home?
- What was your favorite toy when you were a child?
- What did you do on your first job?
- What was your first phone number as a child?
- What year did you purchase your first car?
- What was the name of your first pet?
- Who is your favorite politician?
- Who is your most disliked politician?
- Who is a famous, living person you would most like to meet?
- Who was a famous, now deceased person you would have liked to meet?
- Who is your favorite artist?
- Who is your favorite author?
- With whom did you share your first romantic kiss?
- Who was your favorite elementary school teacher?

A.2 Questions with numeric answers

Sample security questions, that have numeric answers and so are suitable for authentication using a touch-tone phone, include:

- What is your favorite radio station (number on the dial - NNNN)?
- In what year did you start with your company?
- On what date were you hired?
- What is your parents' wedding anniversary date?
- Type a significant date in your life (YYYYMMDD)?
- What are the last 4 digits of your SSN?
- What are the last 4 digits of your home phone?
- What is a birth date of a family member?
- What is a relative's telephone number that is not your own?
- What is the date of your anniversary [mm/dd/yyyy]?
- What is the mileage distance you live from your furthest relative?
- What is your Country or employee identification number?
- What is your date of birth (MM/DD/YYYY)?
- What is your driver's license number?
- What is your favorite dial number of radio station?
- What is your favorite or lucky number?
- What is your passport number?
- What is your social security number?
- On what year you will be eligible to retire?

B References

This document was prepared by Hitachi ID and reflects expertise with password management accumulated through many deployments of Hitachi ID Password Manager.

- About Hitachi ID:
 - <http://Hitachi-ID.com>
- About Password Manager:
 - <http://Password-Manager.Hitachi-ID.com>
- Password guessing programs:
 - <http://sectools.org/crackers.html>
 - <http://www.l0phtcrack.com/>
 - <http://www.openwall.com/john/>
 - <ftp://coast.cs.purdue.edu/pub/tools/unix/pwdutils/crack/>
- NIST / FIPS documents about password management:
 - <http://www.itl.nist.gov/fipspubs/fip112.htm>
 - <http://csrc.nist.gov/publications/drafts/800-118/draft-sp800-118.pdf>
- Best practices for managing passwords for privileged users (as opposed to normal users):
 - <http://privileged-password-manager.hitachi-id.com/docs/privileged-password-management-best-practices.html>
- Unicode and code pages (for multi-lingual users):
 - <http://unicode.org/>
 - <http://msdn.microsoft.com/en-us/goglobal/bb964654.aspx>
 - <http://msdn.microsoft.com/en-us/goglobal/bb964656.aspx>
- Password cracking and password security:
 - http://en.wikipedia.org/wiki/Password_cracking
 - <http://www.cs.unibo.it/~montreso/doc/papers/Morris-PasswordSecurity.pdf>
- Choosing secure passwords:
 - http://www.schneier.com/blog/archives/2007/01/choosing_secure.html
- Disabling LAN Manager passwords on AD:
 - <http://support.microsoft.com/kb/299656>
- Cryptographic hash functions:
 - http://en.wikipedia.org/wiki/Cryptographic_hash_function

- IPsec:
 - <http://en.wikipedia.org/wiki/IPsec>
 - <http://www.securityfocus.com/infocus/1519>
 - <http://www.ipsec-howto.org/>