RETIRED DRAFT

April 1, 2016

The attached DRAFT document (provided here for historical purposes):

> Draft NIST Special Publication (SP) 800-118, *Guide to Enterprise Password Management* (posted for public comment on April 21, 2009)

has been RETIRED.

Information on other NIST cybersecurity publications and programs can be found at: http://csrc.nist.gov/.

The following information was originally posted with the attached DRAFT document:

Apr. 21, 2009

SP 800-118

DRAFT Guide to Enterprise Password Management

NIST announces that Draft Special Publication (SP) 800-118, *Guide to Enterprise Password Management*, has been released for public comment. SP 800-118 is intended to help organizations understand and mitigate common threats against their character-based passwords. The guide focuses on topics such as defining password policy requirements and selecting centralized and local password management solutions.

NIST requests comments on draft SP 800-118 by May 29, 2009. Please submit comments to 800-118comments @nist.gov with "Comments SP 800-118" in the subject line.

National Institute of
Standards and Technology
U.S. Department of Commerce

Special Publication 800-118
(Draft)

Guide to Enterprise Password Management (Draft)

Recommendations of the National Institute of Standards and Technology

Karen Scarfone
Murugiah Souppaya

NIST Special Publication 800-118 (Draft)

Guide to Enterprise Password Management (Draft)

Recommendations of the National Institute of Standards and Technology

Karen Scarfone
Murugiah Souppaya

C O M P U T E R S E C U R I T Y

Computer Security Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899-8930

April 2009

U.S. Department of Commerce

Gary Locke, Secretary

National Institute of Standards and Technology

Dr. Patrick D. Gallagher, Deputy Director

Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analysis to advance the development and productive use of information technology. ITL's responsibilities include the development of technical, physical, administrative, and management standards and guidelines for the cost-effective security and privacy of sensitive unclassified information in Federal computer systems. This Special Publication 800-series reports on ITL's research, guidance, and outreach efforts in computer security and its collaborative activities with industry, government, and academic organizations.

National Institute of Standards and Technology Special Publication 800-118 (Draft)
Natl. Inst. Stand. Technol. Spec. Publ. 800-118, 38 pages (Apr. 2009)

Acknowledgements

The authors, Karen Scarfone and Murugiah Souppaya of the National Institute of Standards and Technology (NIST), wish to thank their colleagues who reviewed drafts of this report and contributed to its technical content. The authors would like to acknowledge Tim Grance, Elaine Barker, Bill Burr, and Donna Dodson of NIST; Paul Hoffman of the VPN Consortium; and Steven Allison, Stefan Larson, Lawrence Lauderdale, Daniel Owens, and Victoria Thompson of Booz Allen Hamilton for their keen and insightful assistance in the development of the document.

Additional acknowledgements will be added to the final version of the publication.

Table of Contents

List of Appendices

List of Tables

Executive Summary

Passwords are used in many ways to protect data, systems, and networks. For example, passwords are used to authenticate users of operating systems and applications such as email, labor recording, and remote access. Passwords are also used to protect files and other stored information, such as password-protecting a single compressed file, a cryptographic key, or an encrypted hard drive. In addition, passwords are often used in less visible ways; for example, a biometric device may generate a password based on a fingerprint scan, and that password is then used for authentication.

This publication provides recommendations for password management, which is the process of defining, implementing, and maintaining password policies throughout an enterprise. Effective password management reduces the risk of compromise of password-based authentication systems. Organizations need to protect the confidentiality, integrity, and availability of passwords so that all authorized users—and no unauthorized users—can use passwords successfully as needed. Integrity and availability should be ensured by typical data security controls, such as using access control lists to prevent attackers from overwriting passwords and having secured backups of password files. Ensuring the confidentiality of passwords is considerably more challenging and involves a number of security controls along with decisions involving the characteristics of the passwords themselves. For example, requiring that passwords be long and complex makes it less likely that attackers will guess or crack them, but it also makes the passwords harder for users to remember, and thus more likely to be stored insecurely. This increases the likelihood that users will store their passwords insecurely and expose them to attackers.

Organizations should be aware of the drawbacks of using password-based authentication. There are many types of threats against passwords, and most of these threats can only be partially mitigated. Also, users are burdened with memorizing and managing an ever-increasing number of passwords. However, although the existing mechanisms for enterprise password management can somewhat alleviate this burden, they each have significant usability disadvantages and can also cause more serious security incidents because they permit access to many systems through a single authenticator. Therefore, organizations should make long-term plans for replacing or supplementing password-based authentication with stronger forms of authentication for resources with higher security needs.

Organizations should implement the following recommendations to protect the confidentiality of their passwords.

Create a password policy that specifies all of the organization's password management-related requirements.

Password management-related requirements include password storage and transmission, password composition, and password issuance and reset procedures. In addition to the recommendations provided in this publication, organizations should also take into account applicable mandates (e.g., FISMA), regulations, and other requirements and guidelines related to passwords. An organization's password policy should be flexible enough to accommodate the differing password capabilities provided by various operating systems and applications. For example, the encryption algorithms and password character sets they support may differ. Organizations should review their password policies periodically, particularly as major technology changes occur (e.g., new operating system) that may affect password management.

Protect passwords from attacks that capture passwords.

Attackers may capture passwords in several ways, each necessitating different security controls. For example, attackers might attempt to access OS and application passwords stored on hosts, so such passwords should be stored using additional security controls, such as restricting access to files that

contain passwords and storing one-way cryptographic hashes of passwords instead of the passwords themselves. Passwords transmitted over networks should be protected from sniffing threats by encrypting the passwords or the communications containing them, or by other suitable means. Users should be made aware of threats against their knowledge and behavior, such as phishing attacks, keystroke loggers, and shoulder surfing, and how they should respond when they suspect an attack may be occurring. Organizations also need to ensure that they verify the identity of users who are attempting to recover a forgotten password or reset a password, so that a password is not inadvertently provided to an attacker.

Configure password mechanisms to reduce the likelihood of successful password guessing and cracking.

Password guessing attacks can be mitigated rather easily by ensuring that passwords are sufficiently complex and by limiting the frequency of authentication attempts, such as having a brief delay after each failed authentication attempt or locking out an account after many consecutive failed attempts. Password cracking attacks can be mitigated by using strong passwords, choosing strong cryptographic algorithms and implementations for password hashing, and protecting the confidentiality of password hashes. Changing passwords periodically also slightly reduces the risk posed by cracking. Password strength is based on several factors, including password complexity, password length, and user knowledge of strong password characteristics. Organizations should consider which factors are enforceable when establishing policy requirements for password strength, and also whether or not users will need to memorize the passwords.

Determine requirements for password expiration based on balancing security needs and usability.

Many organizations implement password expiration mechanisms to reduce the potential impact of unauthorized use of a password. This is beneficial in some cases but ineffective in others, such as when the attacker can compromise the new password through the same keylogger that was used to capture the old password. Password expiration is also a source of frustration to users, who are often required to create and remember new passwords every few months for dozens of accounts, and thus tend to choose weak passwords and use the same few passwords for many accounts. Organizations should consider several factors when determining password expiration requirements, including the availability of secure storage for user passwords, the level of threats against the passwords, the frequency of authentication (daily versus annually), the strength of password storage, and the effectiveness or ineffectiveness of password expiration against cracking. Organizations should consider having different policies for password expiration for different types of systems, operating systems, and applications, to reflect their varying security needs and usability requirements.

1. Introduction

1.1 Authority

The National Institute of Standards and Technology (NIST) developed this document in furtherance of its statutory responsibilities under the Federal Information Security Management Act (FISMA) of 2002, Public Law 107-347.

NIST is responsible for developing standards and guidelines, including minimum requirements, for providing adequate information security for all agency operations and assets; but such standards and guidelines shall not apply to national security systems. This guideline is consistent with the requirements of the Office of Management and Budget (OMB) Circular A-130, Section 8b (3), "Securing Agency Information Systems," as analyzed in A-130, Appendix IV: Analysis of Key Sections. Supplemental information is provided in A-130, Appendix III.

This guideline has been prepared for use by Federal agencies. It may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright, though attribution is desired.

Nothing in this document should be taken to contradict standards and guidelines made mandatory and binding on Federal agencies by the Secretary of Commerce under statutory authority, nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other Federal official.

1.2 Purpose and Scope

The purpose of this guide is to assist organizations in understanding common threats against their character-based passwords and how to mitigate those threats within the enterprise. Topics addressed in the guide include defining password policy requirements and selecting centralized and local password management solutions. Non-character-based passwords, such as graphic-based passwords, are outside the scope of this guide.

1.3 Audience

This guide is for computer security staff and program managers, system and network administrators, and other staff who are responsible for the technical aspects of enterprise password management. Managers can also use the information presented in the guide to facilitate the decision-making processes associated with password management, such as password policy creation. The material in this guide is technically oriented, and it is assumed that readers have at least a basic understanding of system and network security.

1.4 Guide Structure

The remainder of the guide is organized into the following major sections:

- Section 2 presents a high-level introduction to passwords.
- Section 3 describes the four major types of threats to passwords: password capture, exploitation of weak passwords and password hashes, password replacement, and attacker reuse of compromised passwords. It also provides recommendations for mitigating these threats.
- Section 4 addresses centralized and local password management solutions.

This guide also contains supporting appendices:

- Appendix A discusses several common types of passwords for devices and other hardware.
- Appendix B provides a glossary of terms.
- Appendix C provides a list of acronyms and abbreviations used in this document.

2. Introduction to Passwords and Password Management

A *password* is a secret (typically a character string) that a claimant uses to authenticate its identity. Using a password with a user identifier, such as a username, is one form of *identification and authentication*.[1] *Identification* is a claimant presenting an identifier that indicates a user identity for the system. *Authentication* is the process of establishing confidence in the validity of a claimant's presented identifier, usually as a prerequisite for granting access to resources in an information system.

Authentication can involve something the user knows (e.g., a password), something the user has (e.g., a smart card), or something the user "is" (e.g., a fingerprint or voice pattern). *Single-factor authentication* uses only one of the three forms of authentication, while *two-factor authentication* uses any two of the three forms and *three-factor authentication* uses all three forms. Using additional factors makes it more difficult for someone to gain unauthorized access to the system. For instance, it is easier to either discover a user's password or steal the user's smart card than it is to both steal the smart card and also discover the user's password. To meet various security and operational needs, the selection of authentication methods varies among systems, but passwords are the most commonly used authentication method, and are often used both by themselves and with other authentication factors.[2]

Passwords are used in many ways to protect data, systems, and networks. For example, passwords are used to authenticate users of operating systems, applications (e.g., email, labor recording), hardware, and remote access solutions. Passwords are also used to protect files and other stored information, such as password-protecting a single compressed file, a cryptographic key, or an encrypted hard drive. In addition, passwords are often used in less visible ways; for example, a biometric device may generate a password based on a fingerprint scan, and that password is then used for authentication.

There are different forms of passwords. One is known as a *personal identification number* (PIN). A PIN is relatively short (usually 4 to 6 characters) and consists of only digits. Examples of PINs are "7352" and "832290". They take less time to enter than other types of passwords, so they are often used when a longer, more complex password might create human safety problems, such as in a fire suppression system or air traffic control tower console. (In these environments, it is assumed that there are physical security controls in place that compensate for the relatively low security provided by the PIN.) PINs are also used for alarm systems, automated teller machines (ATM), security token devices, and other devices that have small keypads. PINs are rarely used as the only form of authentication for IT system access. Throughout the rest of this document, PINs will be considered out of scope in references to the term "password" unless explicitly mentioned.

Another specialized form of password is known as a *passphrase.* This is a relatively long password consisting of a series of words, such as a phrase or a full sentence. An example of a passphrase is "Iamdefinitelyyour#1fan". The motivation for passphrases is that they can be longer than single-word passwords but easier to remember than a sequence of arbitrary letters, digits, and special characters, such as "72*^dSd!" or "C8ke2.e3:". However, a simple passphrase such as "iloverocknroll" is predictable and therefore easier for an attacker to guess than "9j%a#F.0", so a passphrase's length alone does not make it stronger than other passwords. Throughout the rest of this publication, the term "password" includes both regular passwords and passphrases unless otherwise noted.

---

[1] In some cases, passwords are used without a user identifier. This is most common in situations with low-security needs, such as entering a numeric code into an office copying machine. This publication assumes that a password is associated with a user identifier unless specifically noted otherwise.

[2] Additional information on the selection of appropriate authentication methods and on two-factor and three-factor authentication is available from NIST Special Publication (SP) 800-63 Revision 1, *Electronic Authentication Guideline (Draft),* at http://csrc.nist.gov/publications/PubsSPs.html.

Password management is the process of defining, implementing, and maintaining password policies throughout an enterprise. Effective password management reduces the risk of compromise of password-based authentication systems to the extent possible. Organizations need to protect the confidentiality, integrity, and availability of passwords so that all authorized users—and no unauthorized users—can use passwords successfully as needed. Integrity and availability should be ensured by typical data security controls, such as using access control lists to prevent attackers from overwriting passwords and having secured backups of password files. Ensuring the confidentiality of passwords is considerably more challenging and involves a number of security controls along with decisions involving the characteristics of the passwords themselves. For example, requiring that passwords be long and complex makes it less likely that attackers will guess or crack them, but it also makes the passwords harder for users to remember, and thus more likely to be stored insecurely. This increases the likelihood that users will store their passwords insecurely and expose them to attackers.

Organizations may also be concerned about protecting the confidentiality of user identifiers, such as usernames. Concealing these makes it harder for attackers to perform targeted attacks. However, in many cases concealing identifiers is not helpful because the identifiers are based on a user's email address, first and last name, or other information readily available to attackers. For higher-security situations, where targeted attacks are of particular concern, it may be somewhat helpful to use a unique identifier scheme that is unlike any other organization-issued identifier. If a user uses the same password across multiple systems, having different identifiers makes it less likely that an attacker who gets a user's password on one system will be able to reuse it on other systems. However, using different identifiers has limited security value because many threats capture identifiers along with passwords; also, users have to remember each identifier or record the identifiers in a readily accessible location.

An organization should have a password policy that specifies all of its password management-related requirements. These requirements should include password storage and transmission, password composition, and password issuance and reset procedures. In addition to the recommendations provided in this publication, organizations should also take into account applicable mandates (e.g., FISMA), regulations, and other requirements and guidelines related to passwords. An organization's password policy should be flexible enough to accommodate the differing password capabilities provided by various operating systems and applications. For example, the encryption algorithms and password character sets they support may differ. Policies should also take into account the protection provided by different password mechanisms and the compensating controls that may be needed to address weaknesses in those mechanisms.

After developing a policy, organizations should then select security controls that implement the password policy. NIST Special Publication (SP) 800-53[3] identifies a number of security controls specifically related to identification and authentication. The controls required by NIST SP 800-53 vary based on the security categorization of the system, as defined in the Federal Information Processing Standards (FIPS) Publication 199[4]—low, moderate, or high. Under NIST SP 800-53, the minimum requirements for passwords vary according to the FIPS 199 level.

An organization should review its password policy periodically, particularly as major technology changes occur (e.g., new desktop operating system) that may affect password management. Also, an organization should review its password-related security controls periodically to ensure that they comply with the organization's password policy.

---

[3] NIST SP 800-53 Revision 2, *Recommended Security Controls for Federal Information Systems*, is available at http://csrc.nist.gov/publications/nistpubs/800-53-Rev2/sp800-53-rev2-final.pdf.

[4] FIPS 199, *Standards for Security Categorization of Federal Information Systems*, is available at http://csrc.nist.gov/publications/fips/fips199/FIPS-PUB-199-final.pdf.

In addition to securing passwords and password-based authentication mechanisms, organizations should also periodically evaluate the need to move to stronger forms of authentication. There are many types of threats against passwords, and most of these threats can only be partially mitigated. Section 3 describes the threats and possible mitigation measures in detail. Also, users are burdened with memorizing and managing an ever-increasing number of passwords. However, as Section 4 explains, although the existing mechanisms for enterprise password management can somewhat alleviate this burden, they each have significant usability disadvantages and can also cause more serious security incidents because they permit access to many systems through a single authenticator. Therefore, organizations should make long-term plans for replacing password-based authentication with stronger forms of authentication for resources with higher security needs.

3. Mitigating Threats Against Passwords

This section discusses common threats against the confidentiality of passwords. For the purposes of this discussion, the threats are divided into four groups: threats that directly capture passwords, such as installing keyloggers; threats that take advantage of weak passwords and password hashes, such as password guessing and cracking; threats that replace passwords; and threats that involve attackers reusing compromised passwords. These four groups of threats are discussed in detail below, along with recommendations for partially mitigating these threats.

3.1 Password Capturing

Capturing is an attacker acquiring a password from storage, transmission, or user knowledge and behavior. This section discusses common threats in each of these categories and explains how they can be mitigated. Note that password strength policies, such as those described in Section 3.2 for mandating minimum password length and complexity, are ineffective against password capture threats.

3.1.1 Storage

To be used for authentication, operating system (OS) and application passwords are stored on hosts. If the stored passwords are not secured properly, then attackers with physical or logical access to a host may be able to gain access to the passwords. Passwords should not be stored without additional security controls to protect them. Examples of such security controls include:

- Encrypting files that contain passwords. This may be done by the operating system, an application, or a specialized utility such as password management software that is specifically designed to protect the confidentiality of passwords.
- Using OS access control features to restrict access to files that contain passwords. For example, a host could be configured to permit only administrators and certain processes running with administrator-level privileges to access a password file, thus preventing users and user-level processes from accessing passwords.
- Storing one-way cryptographic hashes for passwords instead of storing the passwords themselves. The use of such hashes allows the authentication system to verify during authentication attempts that the correct password has been entered without storing the actual password. An attacker that gains access to hashes cannot determine the corresponding passwords directly from the hashes and must use cracking techniques to attempt to recover the passwords, as discussed in Section 3.

The security controls appropriate for a particular situation are dependent on several factors, such as the host's security capabilities, the threats against the host, and the authentication requirements. For example, cryptographic hashes may not be an option if an authentication protocol requires that an entered password be directly compared to a stored password. Also, if an attacker that gained access to hashes would be likely to crack them within the lifetime of the passwords (i.e., before they expire), then additional controls, such as OS access control lists, may be needed to restrict access to the hashes. Federal agencies must protect passwords using FIPS-approved cryptographic algorithm implementations. Many authentication systems support the protection of passwords only with cryptographic algorithms and implementations that are either no longer FIPS-approved (e.g., DES) or were never FIPS-approved (e.g., MD4, MD5, RC2, RC4). In such situations, agencies must use compensating controls to protect the passwords using FIPS-approved cryptographic means.

Organizations should carefully consider how well passwords and password hashes stored by applications are protected. For example, web browsers, email clients, and other applications can store passwords on

behalf of users, but it is often not apparent how well-secured these passwords are. Also, in most cases these applications automatically fill in passwords as needed without verifying the user's identity, which permits an attacker who can gain access to such a computer to use the passwords immediately. Password management utilities, which are discussed in more detail in Section 4, can also be used to store passwords for users, but they need to be configured properly to achieve the desired level of security. Organizations should decide which types of applications, if any, should be permitted to store passwords and password hashes based on a consideration of the risks of doing so versus the convenience provided to users. Organizations should have requirements in their password policies regarding which types of applications may store passwords and hashes, as well as how those stored passwords and hashes should be protected.

In addition to being stored on a host's storage media (e.g., hard drive), passwords and password hashes are also stored temporarily in a host's memory, swap files, and similar locations. An attacker who gains access to these resources while passwords or hashes are stored there can potentially recover passwords; utilities to extract passwords from certain operating systems are publicly available. For particularly high-risk hosts, organizations should consider evaluating their temporary password storage to ensure that passwords and hashes are in temporary storage for only a short time and are properly cleared from temporary storage once they are no longer needed.

In addition to storing passwords on the host, users and administrators may also keep passwords on paper so that they do not have to remember the passwords. Such papers should be adequately physically secured, such as stored in a locked file cabinet, safe, or office, to prevent the passwords from being acquired by a malicious party with physical access to the workspace. Also, papers containing passwords should be discarded properly, such as shredding them instead of throwing them in a trash can or recycling bin.

3.1.2 Transmission

Many passwords and password hashes are transmitted over internal and external networks to provide authentication capabilities between hosts. The main threat to transmitted passwords and hashes is *sniffing*, which involves using a wired or wireless sniffer to listen to network traffic. Sniffing may occur as passive eavesdropping or active interception, such as a man-in-the-middle attack with an attacker serving as an intermediary through which messages between two other systems pass. Most sniffers offer the ability to decode and analyze the data gathered if the sniffer knows the packet structure. Sniffers can gather usernames and passwords that are sent unencrypted by protocols such as Telnet, File Transfer Protocol (FTP), Post Office Protocol 3 (POP3), and Hypertext Transfer Protocol (HTTP). Other protocols use flawed cryptographic algorithm implementations for password protection that attackers can easily circumvent. Some sniffers can automatically filter out usernames and passwords from other observed information, providing the attacker an uncluttered view of captured account information and data. Some sniffers can also identify password hashes, which an attacker might be able to crack.

Sniffing threats can be mitigated in several ways, including the following:

- Encrypting the passwords or the communications containing the passwords, such as using Transport Layer Security (TLS) or tunneling the communications through a virtual private network (VPN). For Federal agencies, the encryption mechanisms used to protect password confidentiality must use FIPS-approved algorithms and implementations.
- Transmitting cryptographic password hashes instead of plaintext passwords.
- Switching from protocols that do not protect passwords to protocols that do. Examples are switching from telnet to Secure Shell (SSH) and from HTTP to HTTP Secure (HTTPS).

- Using network segregation and fully switched networks to protect passwords transmitted on internal networks. Note that these methods reduce, but do not eliminate, the possibility of sniffing.
- Replacing a password implementation that exposes the passwords to sniffing with a more secure password-based authentication protocol, such as Kerberos.

Because of sniffing threats, passwords, and in most cases password hashes, should not be transmitted across untrusted networks without additional encryption unless the passwords have no value and cannot be used to gain access to any significant resources.

Another threat against password transmission is *replay attacks*, which involve an attacker resending captured traffic in the hopes of getting the same response as the original traffic. When passwords are involved, replay attacks are attempts to gain access to information without having to know valid credentials. For example, if an attacker can sniff packets that contain encrypted authentication credentials, the attacker may be able to re-send the encrypted credentials—without ever decrypting them—and be authenticated by the recipient if the authentication protocol is vulnerable to replay attacks. Organizations should mitigate such attacks on untrusted networks by using an authentication protocol that offers anti-replay features, such as incorporating timestamps into the authentication packets, or by using compensating controls that prevent replay, such as wrapping the authentication protocol within another protocol that protects it (e.g., TLS).

3.1.3 User Knowledge and Behavior

Passwords may be captured by taking advantage of user knowledge and behavior. When users enter passwords into a computer, the passwords can be captured through non-technical means such as *shoulder surfing*—simply watching a user type a password. Although this can be somewhat mitigated by having hosts hide the password by displaying asterisks or other symbols as the user types, a trained observer who is monitoring keystrokes can determine most or all of the characters being typed. Users should be made aware of shoulder surfing threats and advised to be aware of their surroundings before and during password entry.

Password entry can also be monitored by attackers through technical means. For example, a *keystroke logger*, also known as a keylogger, is a form of malware that monitors the keyboard for action events, such as a key being pressed, and provides the observed keystrokes to an attacker. An attacker can use a keystroke logger to acquire the usernames and passwords typed into the infected computer. Many Trojan horses and some other forms of malware can also monitor user activity to gather usernames, passwords, and other sensitive pieces of information for attackers. These sorts of threats can be mitigated by securing users' hosts effectively, including applying patches regularly, using antimalware software (e.g., antivirus software, antispyware software), and having the user run with user-level privileges, not administrator-level privileges, for daily tasks. Another possible mitigation technique is to avoid typing passwords, such as retrieving them from secure storage or using onscreen simulated keyboards to enter them. Users should also be made aware of common attack vectors for malware threats and how to avoid malware infections, such as not downloading and executing files from unknown sources. Users should also be cautioned not to enter passwords into publicly accessible computers, such as kiosk computers at conferences and hotels, because of the high risk of the passwords being compromised.

Users may also reveal their passwords to attackers because of social engineering. For example, an attacker could pretend to be a help desk agent, call a user, and ask the user to provide a password to assist the agent in troubleshooting a problem. Social engineering can take many forms, some of which involve technical methods, such as phishing emails that direct users to a malicious web site that mimics a legitimate site. The goal behind many phishing attacks is to collect usernames, passwords, and other sensitive information from users. Mitigation of social engineering threats primarily involves user

awareness of such threats and how users should handle them, although some technical controls are also available (for example, many web browsers offer anti-phishing capabilities). Social engineering may also target help desk agents, system administrators, and other IT staff with access to privileged accounts, so organizations should ensure that they are aware of how to recognize such attacks and how to respond when an attack is suspected.

Another problem with users revealing passwords is that a malicious insider, such as a disgruntled current or former employee, may know valid passwords and share them with other parties. A malicious insider may also be intimately familiar with authentication processes and protections, particularly their weaknesses. A user might also benignly share passwords with other users, such as to grant a colleague access to a system for which the colleague has not been specifically authorized.

3.2 Password Guessing and Cracking

Attackers attempt to determine weak passwords and to recover passwords from password hashes through two types of techniques: guessing and cracking. *Guessing* involves repeatedly attempting to authenticate using default passwords, dictionary words, and other possible passwords. *Cracking* is the process of an attacker recovering cryptographic password hashes and using various analysis methods to attempt to identify a character string that will produce one of these hashes, thereby being the equivalent of the password to the targeted system. Guessing can be attempted by any attacker that can access the authentication interface, whereas cracking can only be attempted by an attacker who has already gained access to password hashes. This section describes guessing and cracking in detail and recommends strategies for mitigating these threats.

3.2.1 Guessing

There are several forms of guessing. In a *brute force attack*, the attacker attempts to guess the password using all possible combinations of characters from a given character set and for passwords up to a given length. This method is likely to take an extensive amount of time if there are many combinations to be tested. In a *dictionary attack*, the attacker attempts to guess the password using a list of possible passwords. The list may contain numbers, letters, and symbols, but is not an exhaustive list of all possible passwords or combinations that could create a password. In a *hybrid attack*, the attacker uses a dictionary that contains possible passwords and then uses variations through brute force methods of the original passwords in the dictionary to create new potential passwords. Since the attacker is adding characters—and in some cases replacing characters based on a rule set—in a controlled manner, the attack is more exhaustive than a dictionary attack but takes less time than a brute force attack. Another form of guessing attack is to search the victim's information for possible password content, such as family member names or birthdates.

Guessing attacks can be mitigated rather easily by using a combination of two methods. First, ensure that passwords are sufficiently complex so that attackers cannot readily guess them. It is particularly important to change all default OS and application passwords; lists of default accounts and passwords are widely available to attackers. Organizations should also ensure that other trivial passwords cannot be set, such as the username or person's name, "password", the organization's name, simple keyboard patterns (e.g., "qwerty", "1234!@#$"), dates (e.g., "03011970"), dictionary words, and names of people and places. Most password mechanisms have the ability to prevent the use of such passwords. Additional information on password strength is provided in Section 3.2.3.

The second method recommended for mitigating guessing attacks is to configure OS and application password authentication mechanisms to limit the frequency of authentication attempts. Examples of how this can be accomplished include the following:

- Lock out a user account after a number of consecutive failed authentication attempts (often performed within a particular time period, such as the past hour). For example, after a user has failed to provide the correct password 50 times in a row, ignore all additional authentication attempts to the user account for 15 minutes. Locking out an account after only a few failed attempts has a significant impact on legitimate users and tends to cause them to choose simpler passwords or store their passwords insecurely, thus weakening security.
- Have a fixed or exponentially increasing delay after each failed authentication attempt. After the first failure, for example, there could be a five-second delay; after the second failure, a 10-second delay; after the third failure, a 20-second delay, and so on.

Guessing is made easier by password mechanisms that inadvertently provide information about passwords to attackers. For example, information might be available when a password is entered, such as an input field that only accepts a maximum of eight characters, says that the username does or does not exist, or that has its "OK" button grayed out until the minimum required number of characters has been entered. This information is very helpful to authorized users when they are creating new passwords, but when this information is provided during authentication, it may benefit attackers more than legitimate users.

A special case of password guessing is the use of default passwords for password resets, such as when accounts are first created. A password reset is often accomplished by setting a one-time password (OTP), which is a password that is set to expire immediately, and thus can only be used to gain access to a system one time.[5] An example of how OTPs are used is a help desk staff member creating a new account. The help desk member sets an OTP for an account and provides the OTP to the user. The user may log in with the OTP once, at which point the OTP expires and the user is required to set a new password. Randomly generated or arbitrarily chosen OTPs, not default or patterned passwords (e.g., "NIST0722"), should be used during account creation and password reset processes. This ensures that if the user does not promptly change the assigned password, that the password will not be easily guessable. In some automated procedures, using a random OTP can be omitted because the user will set a new password immediately after verifying his or her identity to the system. Also, if a help desk agent or other security administrator walks the user through setting a new password in a timely fashion, a random OTP may not be necessary.

3.2.2 Cracking

Cracking involves attempting to discover a character string that will produce the same encrypted hash as the target password. The discovered string may be the actual password or another password that happens to produce the same hash. If the hash algorithm is weak, cracking may be much easier. Hash functions should be one-way, otherwise attackers that can access hashes may be able to identify passwords from them and successfully authenticate. Another example of a hash algorithm weakness is that some algorithms do not use salting. *Salting* is the inclusion of a random value in the password hashing process that greatly decreases the likelihood of identical passwords returning the same hash. If two users choose the same password, salting can make it highly unlikely that their hashes are the same.

Attackers using cracking techniques often employ *rainbow tables*, which are lookup tables that contain pre-computed password hashes. These tables allow an attacker to attempt to crack a password with minimal time on the victim system and without constantly having to regenerate hashes if the attacker is attempting to crack multiple accounts. For instance, the attacker generates or acquires a rainbow table that contains every permutation for a given character set up to a certain length of characters. The attacker then uses the table against two separate password hash files, but does not have to generate the permutations

[5] Some OTPs are time-synchronized, which means that the password may be used for a short period of time before it expires. Such an OTP could be used multiple times within that time period.

twice since they were previously created. This allows the attacker to avoid re-computation and to perform cracking more quickly by traversing the lookup table versus generating the hashes on-the-fly.

There are some issues with using rainbow tables. They can take large amounts of storage and can take a long time to create (although the latter issue may not be important if the attacker can acquire copies of existing tables or reuse tables that the attacker previously created). Also, the use of rainbow tables can be hampered by using salting. Rainbow tables will not produce the right results if they do not take salting into account, which dramatically increases the amount of space that the tables require; larger salts effectively make the use of rainbow tables infeasible. Many OSs, such as Mac OS X and other Unix-based OSs, often implement salted password hashing mechanisms to reduce the effectiveness of password cracking. Another technique that helps mitigate the use of rainbow tables is called stretching. *Stretching* involves hashing each password and its salt thousands of times. This makes the creation of the rainbow tables correspondingly more time-consuming, while having little effect on the amount of effort needed by the organization's systems to verify password authentication attempts. Section 3.4 provides more information on how salts can affect cracking.

All forms of cracking can be mitigated by making passwords strong, using one-way password hash algorithms, and protecting the confidentiality of password hashes. Changing passwords periodically also slightly reduces the risk posed by cracking. Section 3.2.3 provides details on these mitigation techniques.

3.2.3 Password Strength

Having strong passwords helps mitigate guessing and cracking. Password strength is determined by a password's length and its complexity, which is determined by the unpredictability of its characters. An example of a password complexity policy is requiring that characters from at least three of the following four groups be present in every password: lowercase letters, uppercase letters, digits, and symbols.[6]

Table 3-1 illustrates the effect of password length and complexity by showing the possible approximate keyspace for passwords using various lengths and character sets. *Keyspace* is the total number of possible values that a key, such as a password, can have. For example, a four-digit PIN could have any of 10 different values (0 through 9) for each of its four characters: the keyspace would be 10^4, or 10,000 (i.e., 0000 – 9999). An eight-character password using a character set of 95 has a key space of 95^8, approximately $7 * 10^{15}$—7 quadrillion possible passwords. As the keyspace increases, the time required to perform an exhaustive brute force attack on a password increases.

The table shows that keyspace increases somewhat as the complexity increases and more rapidly as the length increases. Increasing the character set from 26 characters to 95 characters on a four character-length password increases the keyspace almost 200 times. However, if the length of the password is increased from four to 12, given a character set of only 26 characters, the keyspace increases by almost 200 billion times. Although both have significant effect on the overall strength of a password in resisting brute force attacks, outside of cryptographic attacks, length seems to be the dominating factor in determining password strength. Also, password length is often set as a range, such as permitting passwords from 8 to 15 characters long, which further increases the keyspace. Setting a narrow range for password length has implications other than keyspace—for example, a range of six to eight characters significantly limits users in their password choices, such as not being able to use passphrases.

[6] Other types of characters can be entered in some cases, such as ASCII characters that do not appear on the keyboard. These are not supported by many password mechanisms and are much more difficult for users to employ.

Table 3-1. Possible Keyspaces by Password Length and Character Set Size

Char. Set Size	Character Types				Password Length				
	Digits	Letters	Symbols	Other	4	8	12	16	20
10	Decimal				$1*10^4$	$1*10^8$	$1*10^{12}$	$1*10^{16}$	$1*10^{20}$
16	Hexa-decimal				$7*10^4$	$4*10^9$	$3*10^{14}$	$2*10^{19}$	$1*10^{24}$
26		Case-insensitive			$5*10^5$	$2*10^{11}$	$1*10^{17}$	$4*10^{22}$	$2*10^{28}$
36	Decimal	Case-insensitive			$2*10^6$	$3*10^{12}$	$5*10^{18}$	$8*10^{24}$	$1*10^{31}$
46	Decimal	Case-insensitive	10 common[7]		$4*10^6$	$2*10^{13}$	$9*10^{19}$	$4*10^{26}$	$2*10^{33}$
52		Upper and lower			$7*10^6$	$5*10^{13}$	$4*10^{20}$	$3*10^{27}$	$2*10^{34}$
62	Decimal	Upper and lower			$1*10^7$	$2*10^{14}$	$3*10^{21}$	$5*10^{28}$	$7*10^{35}$
72	Decimal	Upper and lower	10 common		$3*10^7$	$7*10^{14}$	$2*10^{22}$	$5*10^{29}$	$1*10^{37}$
95	Decimal	Upper and lower	All symbols on standard keyboard		$8*10^7$	$7*10^{15}$	$5*10^{23}$	$4*10^{31}$	$4*10^{39}$
222	Decimal	Upper and lower	All symbols on standard keyboard	All other ASCII characters	$2*10^9$	$6*10^{18}$	$1*10^{28}$	$3*10^{37}$	$8*10^{46}$

The keyspace numbers shown in Table 3-1 reflect ideal passwords—passwords in which all possible characters are equally likely to be used for each position of the password. However, in practice this is often not the case. Users that create passwords themselves are likely to follow certain patterns—for example, if a system requires at least eight characters in passwords, including upper case, lower case, and numeric characters, users often select passwords with the minimum possible length, with only the first character capitalized and numeric characters at the end (e.g., Nist2008). Other patterns are using digits just as substitutes for particular letters (e.g., the number "1" for the letter "l", the number "0" for the letter "O") and using a special character as the last character in the password. Passwords based on such patterns may meet password complexity and length requirements, but they significantly reduce the keyspace because attackers are aware of these patterns. A similar problem exists with users that select simple passphrases, such as well-known titles—although such passphrases may be long, they consist of concatenated dictionary words and thus have low entropy. Entropy in an information system is the measure of the disorder or randomness in the system. Passwords that do not have sufficient entropy are more likely to be recovered through brute force attacks.

When determining policies for password length and complexity, organizations should consider maximum and likely actual keyspace. If users will be expected to memorize their passwords, then it may be helpful to set policies that make them easier to remember, such as favoring longer passwords over more complex passwords. If passwords are stored and users do not have to remember them, then both length and complexity should be maximized. Another important consideration for password length and complexity policies is the rate at which cracking attacks can be performed. Section 3.4 discusses this in more detail.

[7] For the purposes of this publication, the ten common symbols are the symbols appearing on the 0 through 9 keys on a standard keyboard: !@#$%^&*()

Organizations also need to consider how effectively their password strength requirements can be enforced. Many operating systems and applications may not be able to require compliance with all of the requirements. In such a situation, one way to improve compliance is to add a password filter utility, which is specifically designed to verify that a password being created by a user complies with the password policy; if the password is weak, the user is forced to select a different password. A less rigorous solution is to educate users on password strength requirements and to run password crackers against their stored passwords regularly to identify weak passwords, which are likely violations of the strength requirements.[8] Because this solution is reactive, it should only be used when a proactive solution, such as a password filter, is not feasible. Organizations can also choose to alter their password strength requirements so that they are more easily enforceable—for example, if a system cannot require the use of punctuation marks in passwords but it can require the use of long passwords, then it may be more secure to increase the password length requirement and decrease the password complexity requirement so as not to include punctuation marks.

When setting policy, organizations should also take into account weaknesses in password mechanisms that may undermine password length and complexity requirements. Examples are as follows:

- Some password mechanisms have more limited character sets than users would expect. For example, an application might permit users to enter mixed-case passwords but then converts all lowercase letters to uppercase before hashing the password. This reduces the strength of passwords that use mixed case, often unbeknownst to the user, who may think the password is considerably stronger than it actually is.
- Some password mechanisms accept password characters past the maximum length that is stored or checked. Not knowing the input length limit may inadvertently cause a user to create a weak password. For example, consider a system that accepts an arbitrarily long password string from a user but truncates it to eight characters before hashing it. If a user attempts to set the password to “Security is my #1 Priority!”, the system will truncate the password to “Security”, which is far weaker than the intended password.

3.2.4 User Password Selection

There are two ways to generate passwords: automatic random (or pseudo-random) generation and user selection. Although automatically generated random passwords usually provide greater entropy than user-selected passwords and thus are stronger passwords, they can be hard for users to remember. Conversely, user-generated passwords generally contain less entropy and are weaker and easier to guess or crack but easier for users to remember. For passwords that the user does not need to remember, automatically generated random passwords should be used whenever feasible. A utility called a password generator can be used to create such passwords. A password generator usually has built-in password restrictions, and may also allow the user to specify custom restrictions; the password generator then creates a password that complies with the restrictions. Automatically generated passwords should be as strong as possible, using the full variety of characters allowed (e.g., upper case letters, lower case letters, digits, special characters) and the maximum or nearly maximum length possible. It is generally unrealistic to expect users to memorize these passwords. For passwords that are intended to be memorized, organizations should consider security needs and expected user behavior when deciding which password generation method should be used.

[8] Cracking can be performed offline—on a system other than the system on which the passwords were stored—and in a distributed manner, it can be performed quickly, efficiently, and without affecting the network or the authentication system. The system or systems performing the cracking should be well secured, with these systems contained on a private network that has very limited connectivity, allowing only those connections required by the crackers and only as absolutely needed for the crackers to perform their duties. If possible, the crackers should not be connected to any other network

When users need to choose new passwords, whether user-selected or randomly-generated, the users should be made aware of password requirements, including restrictions on password composition. For example, an application might permit passwords between eight and 20 characters long, require a combination of letters and digits, and prohibit the use of special characters such as punctuation marks. Providing a clear list of password restrictions helps users to select passwords that meet the password criteria and avoid the frustration of having new passwords rejected, which can cause users to choose subsequent passwords more hastily. Organizations should conduct training and awareness activities for their users to ensure that they are aware of the characteristics of strong passwords and the importance of having strong passwords and protecting them.

A variety of methodologies have been created with the intent of helping users select passwords that are both strong and relatively easy-to-remember. Examples of these methodologies are listed below.[9]

- **Mnemonic Method**. A user selects a phrase and extracts a letter of each word in the phrase (e.g., the first letter or second letter of each word), adding numbers or special characters or both. Table 3-2 shows examples of the mnemonic method.

Table 3-2. Mnemonic Method of Password Generation

Phrase	Password
Please be my best valentine!	Pbmbval!
This is the worst car I have ever driven in my LIFE!	TitwcIhedimLIFE!
I am definitely your #1 fan.	Iady#1f.

Although a mnemonic password is generally stronger than a dictionary password—for example, "Pbmbval!" would be much stronger than "valentine"—many mnemonic passwords are still susceptible to brute force guessing attacks. Common phrases converted into mnemonic passwords, without using unusual character substitutions or other alterations, can be guessed by attackers using dictionaries of mnemonic passwords.[10] Users that create mnemonic passwords should either avoid using common phrases, making up their own phrases instead, or should make significant unexpected changes to the passwords, such as changing capitalization and punctuation and spelling out one or more of the words.

- **Altered Passphrases**. A user selects a phrase and alters it to form a derivation of that phrase. This method supports the creation of long, complex passwords. Passphrases can be easy to remember due to the structure of the password: it is usually easier for the human mind to comprehend and remember phrases with a coherent vocabulary than a string of random letters, numbers, and special characters. Table 3-3 shows examples of altered passphrases.

Table 3-3. Altered Passphrases

Passphrase	Alternate Passphrase
to be or not to be	2.be.0r.nOt@to0.bEE
Dressed to the nines	Dressed*2*the*9z

[9] NIST encourages readers of this publication to submit feedback on these methodologies to NIST during the public comment period, as well as to suggest additional methodologies that would be helpful.

[10] Cynthia Kuo, Sasha Romanosky, and Lorrie Faith Cranor, "Human Selection of Mnemonic Phrase-based Passwords", Symposium on Usable Privacy and Security (SOUPS), 2006, Pittsburgh, PA, July 2006.

As with mnemonic passwords, it is important for users who create altered passphrases to avoid the use of common phrases that do not have significant unexpected changes.

- **Combining and Altering Words**. A user can combine two or three unrelated words and change some of the letters to numbers or special characters. Table 3-4 shows examples of combining words.

Table 3-4. Combining and Altering Words

Words	Password
"bank" and "camera"	B@nkC@mera
"mail" and "phone"	m4!lf0N3

Although these techniques are helpful, users may find it difficult to remember several passphrases, mnemonics, or altered word combinations. An alternative strategy is to select a single memorable base password and alter it to form derivations, such as inserting additional letters, numbers, and symbols into the base password. Then each derivation can be used as a password for a different system or application. Table 3-5 shows example password derivations.

Table 3-5. Password Derivations

Derivation	System or Application	Resulting Password
Base password	None (the base password is only used to build other passwords, and is not actually used as-is for any system or application)	G00dTimes
Prepend "42*" to the base	System 1	42*G00dTimes
Append "*42" to the base	System 2	G00dTimes*42
Prepend "42*" to the base and insert "#23" in the middle	Application 1	42*G00d#23Times

The user does not need to memorize all the derivation rules, just the base password. Then the user can write down the derivation rules and refer to the rules as needed to derive the necessary passwords. However, there are two major disadvantages to this method. If an attacker gets access to the rules, they could make it easier for an attacker to guess or crack passwords. Also, if an attacker gets one of the passwords, then the attacker is much more likely to guess or crack the other passwords; this becomes trivial if the attacker can also get the rules. So an attacker who shoulder surfs while a user is entering a password and looking at the rules sheet may be able to gain access to several accounts with little effort.

3.2.5 Local Administrator Password Selection

In most enterprises there are two types of passwords: local and domain. Domain passwords are centralized passwords that are authenticated at an authentication server (e.g., a Lightweight Directory Access Protocol server, an Active Directory server). Local passwords are passwords that are stored and authenticated on the local system (e.g., a workstation or server). Although most local passwords can be managed using centralized password management mechanisms, some can only be managed through third-party tools, scripts, or manual means. A common example is built-in administrator and root accounts. Having a common password shared among all local administrator or root accounts on all machines within

a network simplifies system maintenance, but it is a widespread weakness. If a single machine is compromised, an attacker may be able to recover the password and use it to gain access to all other machines that use the shared password. Organizations should avoid using the same local administrator or root account password across many systems. Also, built-in accounts are often not affected by password policies and filters, so it may be easier to just disable the built-in accounts and use other administrator-level accounts instead.

A solution to this local password management problem is the use of randomly generated passwords, unique to each machine, and a central password database that is used to keep track of local passwords on client machines. Such a database should be strongly secured and access to it limited to only the minimum needed. Specific security controls to implement include only permitting authorized administrators from authorized hosts to access the data, requiring strong authentication to access the database (for example, multi-factor authentication), storing the passwords in the database in an encrypted form (e.g., cryptographic hash), and requiring administrators to verify the identity of the database server before providing authentication credentials to it.

Another solution to management of local account passwords is to generate passwords based on system characteristics such as machine name or media access control (MAC) address. For example, the local password could be based on a cryptographic hash of the MAC address and a standard password. A machine's MAC address, "00:16:59:7F:2C:4D", could be combined with the password "N1stSPsRul308" to form the string "00:16:59:7F:2C:4D N1stSPsRul308". This string could be hashed using SHA and the first 20 characters of the hash used as the password for the machine. This would create a pseudo-salt that would prevent many attackers from discovering that there is a shared password. However, if an attacker recovers one local password, the attacker would be able to determine other local passwords relatively easily.

Regardless of the method chosen, a solution should be implemented that prevents the use of shared local account passwords across many systems.

3.3 Password Replacing

An attacker can successfully authenticate to an account by replacing the account's existing password with another password that is known by the attacker. The attacker does not necessarily need to know the original password to accomplish this—for example, the attacker could intercept a user's legitimate attempt to reset a password. This section describes several ways in which attackers can replace passwords to gain access to accounts.

3.3.1 Forgotten Password Recovery and Resets

When a user forgets a password, generally there are two options: regain access to the old password—*password recovery*—or set a new password—a *password reset*. Password resets are also performed when a new account is created, to set an initial password. There are many ways in which password recovery and resets can be conducted—ranging from an in-person visit with an IT staff member to a fully automated self-service utility. If the identity of the user requesting a password recovery or reset is not properly verified, an attacker could easily pose as a user and gain access to that user's password, so all recovery and reset mechanisms should first verify the user's identity. Examples of verification methods include basic knowledge-based verification (e.g. employee ID number, badge number, date of birth); predetermined challenge response questions set during account creation (e.g., color of first car, favorite pet's name); calling a user back on an office phone; and requiring a face-to-face visit from the user to provide photo identification.

Each verification method has advantages and disadvantages that should be evaluated before use. Privacy concerns should be carefully evaluated; for example, information such as social security numbers and mother's maiden name should not be used for identity verification. User verification should not include data or question answers that can be easily obtained or guessed by an attacker, such as an employee ID number available from a company directory. For each password recovery or reset mechanism, the thoroughness of the user verification can be tailored to the account's relative security needs—for example, organizations might want to require a rigorous, out-of-band verification method for the highest-security passwords and use less rigorous methods for other cases. When selecting verification methods, organizations should consider the relative risk of each method as opposed to its cost and convenience. Organizations should also identify and address any requirements to perform password recovery and resets for people who are not physically located in the organization's main facilities, including users who telecommute or are on travel.

The confidentiality of all sensitive information stored and transmitted as part of password recovery and resets should be protected. For example, if predetermined challenge-response questions or password hint questions are used to verify identity, the confidentiality of the answers should be protected at all times, and the confidentiality of the questions should also be protected if the questions are user-generated or otherwise differ among users. Organizations should also carefully consider using filters to ensure that the answers set by a user to challenge-response questions have reasonable entropy, such as not using the same answer for each question and not using all one-character answers. Organizations should send reset passwords through cleartext email messages and other unsecured applications only in the lowest-security situations because of the risk of interception by attackers.

3.3.2 Access to Stored Account Information and Passwords

Attackers may be able to replace passwords by gaining access to stored user account information and passwords. For example, a host may have incorrect privileges set on its password files that allow a user to overwrite them. The user could set new passwords for others' accounts or create new accounts. A similar attack can be accomplished on many hosts if an attacker gains physical access to the host. There are password reset tools and utilities that can permit an attacker with physical access to reset the built-in administrator account password. Section 3.1.1 contains recommendations for securing stored passwords.

3.3.3 Social Engineering

Attackers may be able to trick users into changing their existing passwords to attacker-selected passwords by using social engineering techniques. Section 3.1.3 contains recommendations for mitigating such attacks.

3.4 Using Compromised Passwords

If an attacker has compromised a password through guessing, cracking, or capture, then the attacker will be able to use that password until it is changed by the user. To reduce the potential impact of such unauthorized password use, many organizations have implemented *password expiration* mechanisms that force a user to select a new password after a certain number of days. Although this is beneficial for reducing the impact of some password compromises, it is ineffective for others—for example, when the attacker can compromise the new password through the same method as the old password (such as a keylogger running on the user's computer) or when the attacker has a way of maintaining access to the target without the password, such as setting up a backdoor on the target. Password expiration is also often a source of frustration to users, who are often required to create and remember new passwords every month or two for dozens of user accounts.

Organizations should decide whether to use password expiration mechanisms and what expiration period to set based on balancing security needs and usability. For example, if the organization provides secure storage for user passwords, so that users do not have to remember passwords, then password expiration will be less frustrating to users. If there are significant threats involving unauthorized access to password hashes, then it may be necessary to set the expiration period to be less than the amount of time required to crack the passwords from the hashes, as discussed in the box below. Another consideration is the frequency of authentication; if an application is accessed only a few times a year by employees and password expiration is enforced, then the passwords will be expired every time the users attempt to authenticate. Other factors for organizations to consider in selecting password expiration requirements include the strength of password storage and transmission algorithms and the system security requirements. Organizations should consider having different policies for password expiration for different types of systems, OSs, and applications, to reflect their varying security needs and usability requirements.

Because of advances in hardware and cracking software and the availability of large numbers of compromised computers through botnets, attackers are constantly increasing their ability to crack passwords. The type of cryptographic algorithm used for the password hashes somewhat affects the cracking speed, but generally does not affect it enough to make cracking ineffective. Security researchers and cracking software vendors claim hash generation speeds for some hash algorithms of hundreds of millions to over a billion per second per computer, with the ability to use thousands of computers simultaneously.[11] Generating a billion hashes per second on each of a thousand machines would equal approximately 2.6 quintillion ($2.6*10^{18}$) hashes per month.

In cases where password hashes are at significant risk of compromise, organizations should take estimates of cracking abilities into consideration when setting policies for password expiration, length, and complexity. Consider the keyspace examples from Table 3-1. A password with a character set size of 72 and a length of 8 characters has a maximum keyspace of $7*10^{14}$. For the example described above, hashes for this entire keyspace could be generated in 12 minutes. Increasing the character set size to 95 only increases the time to 2 hours. However, increasing the length to 12 characters, and keeping the character set size at 72, drastically increases the time needed to generate all the hashes—to over 500 years.

The use of salts also makes cracking more difficult—for example, using 48-bit salting values effectively appends a 48-bit password hash to the original password hash, assuming that the attacker does not have access to the salting values and that the salting values are well-chosen. So a salted password might have the same effective length, and therefore be roughly as time-consuming to crack, as an unsalted password that is several characters longer. Also, salts typically use the full range of possible values, unlike passwords that have limited character sets, so salts can strengthen the effective password complexity. Policies for password expiration, length, and complexity should take into account the use of salts.

In cases where generating all hashes would take many years, having password expiration would be irrelevant for mitigating cracking, even if most users do not take full advantage of the available character set. Generally, password expiration periods are not of much help in mitigating cracking because they have such a small effect on the amount of effort an attacker would need to expend, as compared to the effect of other password policy elements. Suppose that an organization reduced its password expiration period from 60 days to 30 days. An attacker would simply need to use twice the hardware resources to compensate for this change.

When password expiration is enabled and it is expected that users will be memorizing their passwords, it is helpful to provide reminders to users that their passwords will be expiring soon. Giving users at least a

[11] An example is described at http://www.elcomsoft.com/edpr.html?r1=pr&r2=multi-gpu.

few days or a few logons to prepare for a password change will give them a better opportunity to choose a strong password that they are likely to remember. Forcing users to change passwords without warning often results in less complex passwords that are easier to remember, or passwords that are stored insecurely (e.g., written on a notepad, stored in a plaintext user file). If setting a new password requires that a user be physically present at the organization's facilities, but the password can be used remotely (i.e., through remote access for telework), then it is generally prudent to notify users one to two weeks before expiration. This makes it more likely that users will have an opportunity to reset the password before teleworking, particularly if they will be traveling for several days.

Password expiration is not effective unless users select different passwords from those previously used. *Password history* is the retention of one or more previous passwords or password hashes for comparison against new passwords or password hashes. A new password is checked to ensure that it has not been used during the specified history. The period is usually defined as either a certain number of previous passwords or a period of time. Another password attribute closely related to password history is minimum password age. The *minimum password age* is the amount of time that must pass between password changes. To reduce the effort required in remembering passwords, some users will cycle through passwords after expiration until they have exceeded the password history retention buffer and then change their password back to the original. Although enforcing a minimum password age does not prevent this, it is a deterrent.

Some password history mechanisms are also capable of identifying passwords that are not sufficiently different from previous passwords. When forced to select a new password, some users tend to use variations of old passwords (e.g., changing "password07" to "password08"). This makes it trivial for an attacker who knows the old password to guess or crack the new one quickly. Some password history mechanisms can be configured to reject new passwords that have a certain number of characters in common with previous passwords. Without such a mechanism, it is generally easy for users to append counters to their passwords, such as the "password07" and "password08" examples. This renders password expiration largely ineffective, and may actually cause users to choose weaker passwords than they would have without password expiration.

Password history generally only works on a single authentication mechanism and cannot check history from multiple mechanisms. This allows users to use the same password (and previous passwords) on many systems at once. Users often do this because it reduces the number of passwords that they have to remember, but this increases the risk to the enterprise by allowing an attacker who compromises one password to reuse it to gain access to additional resources. In addition, administrators will sometimes reuse password between a local user account on a personal workstation and an account that has domain or centralized administrative privileges. This can pose a major risk to the enterprise because the security of centralized password management is generally higher than on individual workstations. An attacker who compromises the workstation and is able to crack the domain administrator password will have significant access to enterprise resources.

There is generally no easy way to detect password reuse across systems, particularly when both internal and external systems are involved. To attempt to reduce the likelihood of password reuse, organizations can have their password management policies prohibit use of the same or closely-related passwords on organizational IT system and external systems. The password management policy can also explicitly forbid the reuse of centralized (e.g., domain) administrative level credentials with user or local (e.g., local administrator or root) accounts. Proper user training that stresses the importance of proper password management and protection and explains the risks of password reuse should also be implemented. However, without an enforcement mechanism, it is unlikely that policies against reuse will be significantly effective in reducing reuse, given the number of passwords that users typically need to remember.

If an organization believes that a password management system or other source of passwords has been compromised, the organization should act swiftly to mitigate the weaknesses that allowed the compromise, restore the compromised system to a secure state, and require all users to change their passwords immediately. Implementing the enterprise password change will require careful planning and coordination. Procedures should be in place to notify all affected users. This notification should only inform the users of the situation and notify them that their passwords have been reset or need to be changed immediately. Users should be instructed to change their password as they normally would and contact the helpdesk if they need assistance.[12] If users are allowed to change their passwords, a procedure should be in place to force the change and verify that changes have been made. If passwords are reset to assigned passwords, then there should be procedures in place to communicate the assigned passwords to the users in a secure manner. If the procedures that are in place cause a greatly increased workload on help desk staff, there should be resources available to augment help desk staff to ensure they can effectively handle the password resets.

[12] Having a different or special procedure could confuse users and make them more susceptible to phishing attacks in the future.

4. Password Management Solutions

Many organizations implement enterprise password management solutions to reduce the number of user account identifiers and passwords that their users need to remember. Similarly, local password management utilities can also be used for password storage. Enterprise and local password management solutions can reduce the burden on users and help desk staff associated with password changes and resets. Password management solutions reduce the likelihood that passwords will be compromised because they are written down, typed in at a keyboard (and the typing monitored by people or malware), or created weak to make them easier to remember. This section describes two types of centralized password management solutions—single sign-on (SSO) and password synchronization—as well as local password management technologies. It also provides a brief comparison of these technologies, focusing on their usability characteristics.

4.1 Single Sign-On Technology

A *single sign-on (SSO) technology* allows a user to authenticate once and then access all the resources the user is authorized to use. Authentication to the individual resources is handled by the SSO technology in a manner that is transparent to the user. Many authentication methods are supported by SSOs, but this publication only discusses password-based authentication. For this, the SSO typically creates a unique, strong user password for each resource and changes the passwords regularly. Usually the end user does not know any of the resource passwords, just the SSO password. Because a different password is used for each resource and the user does not need to memorize the passwords, the SSO can make each password as strong as each resource will support and change the passwords frequently. SSO solutions can also support the storage and use of multiple identifiers for a single user—for example, "kscarfone" on one system and "scarfon7" on another.

In nearly every environment, it is not feasible to have an SSO solution that handles authentication for every system and resource—most SSO solutions can only handle authentication for some systems and resources, which is called *reduced sign-on (RSO)*. Still, even when only providing a limited RSO capability, an SSO technology can be very effective in reducing the number of usernames and passwords that users need to remember and the number of times that users have to authenticate.

There are many possible architectures for SSO technologies. A common architecture is to have an authentication service, such as Kerberos, for authenticating SSO users, and a database or directory service, such as Lightweight Directory Access Protocol (LDAP), that stores authentication information for the resources the SSO handles authentication for. Regardless of the exact architecture, an SSO solution usually includes one or more centralized servers containing authentication credentials for many users. Such a server becomes a single point of failure for authentication to many resources, so the availability of the server affects the availability of all the resources that rely on the server for authentication services. Also, any compromise of the server can compromise credentials for many resources, which makes the security of the server particularly important.

User authentication to the SSO technology itself is also very important. If proper mutual authentication is not performed, the SSO technology is vulnerable to man-in-the-middle (MITM) attacks. All communications of sensitive authentication information, such as passwords, should have their confidentiality and integrity protected through the use of FIPS-approved cryptography. Replay attacks are also a concern for authentication credentials, so timestamps or other mechanisms to thwart replay attacks should be included in credential transmissions. Another major concern with SSO user authentication is that an SSO password is susceptible to compromise through social engineering, phishing, keylogging, or other means, and such a compromise of a single password could grant an attacker access to many resources.

4.2 Password Synchronization

A *password synchronization* solution takes a password from a user and changes the passwords on other resources to be the same as that password. The user then authenticates directly to each resource using that password; there is no centralized directory or authentication server performing authentication on behalf of the resources. The primary benefit of password synchronization is that it reduces the number of passwords that users need to remember; this may permit users to select stronger passwords and remember them more easily. Unlike SSO technology, password synchronization does not reduce the number of times that users need to authenticate.

Password synchronization solutions are typically easier and less expensive to implement than SSO technologies, but password synchronization also has significant security disadvantages. Because password synchronization causes the same password to be used for many resources, each of which stores the password or a hash of the password, the compromise of any one instance of the password compromises them all. This is particularly damaging if password synchronization is used for resources with significantly different security requirements—for example, a password on a low-security resource may be compromised relatively easily and then reused on a high-security resource that shares the synchronization solution. This allows an attacker to compromise a low-security resource to gain access to a high-security resource.

Another major problem with password synchronization is that it is forced to use a "lowest common denominator" approach to password strength. Suppose that synchronization is planned to be used for 10 systems. Two of these systems do not support the use of special characters, and one of the systems only supports password lengths of 6 to 8 characters. So using password synchronization for these resources would require the use of passwords between 6 and 8 characters long without any special characters. The result of this is that the passwords will be significantly weaker than what most of the individual systems can support.

Passwords may also become unsynchronized. For example, a user might change a resource password directly with that resource instead of going through the password synchronization user interface. This would cause one password to differ from the rest, which would cause future synchronizations with the associated resource to fail. A password could also become changed due to a resource failure that necessitates restoration of a backup, which might contain the previous password for a particular resource.

4.3 Local Password Management

Another approach to password management is local password management software. Password management software is a utility that allows a user to store usernames, passwords, and other small pieces of sensitive information, such as account numbers. Password management software can greatly reduce the number of passwords that users have to remember. The password management software itself has a master password that a user must enter to gain access to the passwords stored by the software.[13] The master password protects the stored passwords from being accessed by someone else and is the only password that the user needs to remember. Some password management software utilities permit users to store the passwords on removable media (e.g., USB flash drive) instead of the local computer; this provides an additional layer of protection if the media is only inserted into the computer when needed and stored separately and securely otherwise.

[13] Some programs allow an authenticator other than a password to be used to gain access to the stored passwords. This can provide stronger protection for the stored passwords.

With most password management software utilities, the user selects an account from a list, which causes the corresponding password to be copied. The user then pastes the password into the password field for the target application or web form. Some utilities further automate this process by automatically pasting the corresponding password in the appropriate application or web form's password field.

The following items are general recommendations for using password management software:

- Set the software's timeout feature so that access to the passwords will be automatically locked after an idle period, such as five minutes.
- Clear the buffer after the password is copied and pasted (many password management software programs do this automatically).
- Back up the password database periodically, especially after a password is changed. If the computer's copy of the password database becomes corrupted or something adverse happens to the computer, the user can get the passwords from the backup copy of the password database.
- Use a strong master password that is not easily guessable or crackable, or an alternate form of authentication that is stronger than a password.
- Password management software should protect the confidentiality of stored passwords using FIPS-approved algorithms and implementations.

Password management software cannot counteract all threats against passwords. For example, if a computer is compromised, such as by malicious code, then a keystroke logger or other malicious means could be used by an attacker to gain access to the password management software and the passwords it is intended to protect. In addition, password management applications have some inherent drawbacks. Users must manually update the password database every time a password changes. If the application has password generation functionality, it would need to be configured each time it is used to generate passwords that comply to the particular password strength requirements for each account. This is highly susceptible to human error and is likely to result in passwords that are weaker than they could be, because users do not take advantage of all the possible password strength features. Finally, some passwords management applications are not centrally managed, so it is possible for a user to improperly configure the application. This could result in unencrypted passwords being cached for long periods or being stored unencrypted, for example. If password management software is not centrally managed, organizations may choose to export passwords from it periodically and check them against the organization's password policy to ensure that they meet the policy's requirements.

Some applications, such as web browsers, commonly provide password management features for the passwords entered into them. In some cases, these applications essentially provide a built-in password management utility that stores the passwords securely and controls access to them through a user-set master password. In other cases, the stored passwords may be stored less securely and may be provided automatically as needed without any user authentication. This permits an attacker who gains physical or logical control over the user's workstation to use the stored passwords without any further steps. Organizations should carefully consider the risks involved in storing passwords locally outside of password management software and generally should permit only the lowest-risk passwords to be stored in such a manner.

4.4 Comparison of Password Management Technologies

Table 4-1 provides a comparison of the three types of password management technologies described in this section. The comparison focuses on the usability of the password management technologies,

including how users and organizations may be affected if the technology fails or a password is compromised.

Table 4-1. Password Management Technology Usability Comparison

	SSO and RSO	Password Synchronization	Local Password Management Software
Is the number of passwords that users have to memorize reduced?	Yes	Yes	Yes
Can the user authenticate without performing additional steps?	Yes	Yes	No
Is the number of authentications that users have to perform reduced?	Yes	No	No
Is the user largely unaffected if the password management technology is temporarily unavailable?	No	Yes	No
Is the impact of a user password for an individual resource being compromised limited to just that resource?	Yes	No	Yes, if the user creates a unique password for each resource

When selecting password management technologies for deployment within an enterprise, organizations should carefully consider how their requirements for usability compare with the characteristics shown in Table 4-1. If a password management technology is deployed that does not meet the organization's usability requirements, users may circumvent the technology, such as writing down passwords instead of storing them in local password management software. Organizations should also consider the security of the password management technologies themselves, such as the security of their password storage and transmission mechanisms. Another important factor is how users authenticate to the password management technologies; if passwords are the form of authentication, then a compromise of one of those passwords would compromise all of that user's passwords that are stored in the password management technology, potentially granting an attacker access to dozens of resources from a single password.

Organizations should also consider the impact level associated with the passwords that would be stored in a password management technology. If a technology is being deployed to support low-impact passwords, and the technology's safeguards have been designed for that level, then it is generally unwise to use the technology for storing higher-impact passwords without reevaluating the strength of its safeguards. In some cases, it is appropriate to deploy a separate password management technology implementation for higher-impact passwords. This is particularly important for password synchronization solutions, which use the same password for each resource. If the resources are of varying security levels, such as both low and moderate-impact systems, an attacker could gain access to a user's password by compromising a low-impact system, which is likely to be less strongly secured than a moderate-impact system. Because the user's password is synchronized across resources, including moderate-impact systems, the attacker could then use this password acquired from a low-impact system to gain access to a moderate-impact systems. Organizations should take care to ensure that attackers cannot gain access to higher-impact systems by taking advantage of password synchronization across systems of varying impacts.

Appendix A—Device and Other Hardware Passwords

This appendix provides recommendations related to several types of passwords for devices and other hardware. These passwords often have significant limitations as compared to the OS and application passwords discussed in the main sections of this guide. For example, many device and other hardware passwords cannot use password filters or otherwise enforce password-related controls, such as password history.

Computer Firmware Passwords

Almost all computer systems with Basic Input Output System (BIOS) firmware come with the ability to password protect it. On many systems, a variety of BIOS passwords can be set. Each type of password provides different protections. For example, BIOS configuration protection passwords are intended to prevent unauthorized changes to the BIOS configuration, such as altering the boot order so that the computer can be booted from removable media. Another example is BIOS boot passwords, which must be entered before the system boots so that unauthorized users cannot boot the computer. However, most motherboard manufacturers have procedures that can be used to remove BIOS passwords and restore the system to a default configuration. In some cases, it requires shorting PINs on the motherboard or removing and replacing a chip on the motherboard, but in other cases, a much simpler procedure is available. For example, many BIOSs include backdoor passwords that will always work; others can be recovered using custom software programs or specific key sequences while the machine is booting. Because of this, BIOS passwords should be considered only a deterrent and do not provide any protection to data on the disk.

A newer technology replacing BIOS firmware is Extensible Firmware Interface (EFI). EFI passwords can be set to protect the system's configuration. However, like BIOS passwords, EFI passwords can be circumvented by anyone who has physical access to the system. EFI passwords should be considered a deterrent to unauthorized access but not a true form of protection.

Hard Drive Passwords

Some hard drives support the use of passwords to restrict access to a hard drive. For example, a drive might have a master password (for administrative purposes) and a user password, and it could support two security modes: high security and maximum security. In a managed IT environment, the user of a system would be given the user password, and the master password would be retained by administrative staff. In high security mode, the drive can be unlocked with either the user or master password, and the hard drive passwords can only be removed from the drive after supplying the master password. In maximum security mode, the drive can only be unlocked with the user password, and the master password can only be used to erase the drive and remove the hard drive passwords (i.e., the drive must be erased before passwords are removed). Unlike BIOS passwords that are stored on a chip on the motherboard, hard drive passwords are stored on the hard drive itself. Even if the disk is moved to a new system, read and write operations cannot be performed on the drive until one of the passwords are entered.

Although hard drive passwords do provide a higher level of security and a more effective deterrent to a casual attacker, there are tools and services available that can retrieve or reset the hard drive passwords, so they cannot be relied on to provide a high level of security. A more effective solution to protect hard drives from unauthorized access is the use of a full disk encryption solution, which encrypts all information on the hard drive and only decrypts it if the appropriate authentication is provided.

Trusted Platform Module (TPM) Passwords

A Trusted Platform Module (TPM) chip is a tamper-resistant integrated circuit built into some motherboards that can perform cryptographic operations (including key generation) and protect small amounts of sensitive information, such as passwords and cryptographic keys. Each TPM chip has an owner password, which is used to gain access to and manage the TPM chip. Although the TPM can be shut off by someone with physical access to the system, it cannot be circumvented: access to the TPM cannot be achieved without the owner password. Therefore, it is important to choose a strong password for the TPM owner password and to protect its confidentiality. If the owner password is lost or forgotten, it can be reset by clearing the TPM, but this action also clears all data stored on the TPM. Therefore, either the owner password or the data on the TPM should be backed up to an alternate secure location, after carefully considering and addressing the security considerations implicit in storing these types of sensitive information.

Network Infrastructure Device Passwords

The simplest method of authentication for a network infrastructure device, such as a router or switch, is local authentication. Authentication credentials are stored on the device, and when a user attempts to authenticate, the presented credentials are compared with stored passwords or password hashes. Passwords stored on network infrastructure devices are sometimes unencrypted, so physical security controls may be needed to protect the passwords from compromise. These devices often have a single administrative account, so if multiple users need to administer a device, a centralized authentication system should be configured for those network devices with a separate account and password for each administrator to provide accountability.

Another common method of network device administration is Simple Network Management Protocol (SNMP). SNMP v1 and v2 rely on cleartext community strings, which are used as passwords to grant access to the device. Since SNMP v1 and v2 send community strings across the network with no cryptographic protection, they should not be used to configure network infrastructure devices over untrusted networks. SNMP v3 provides security feature enhancements to SNMP, including encryption and message authentication. If any version of SNMP is used for remote administration, default SNMP community strings such as “public” and “private” should be removed before real community strings are put into place. If both are present on the device at any time, an attacker could retrieve real community strings from the device using the default string.

General-Use Office Device Passwords

Many general-use office devices, such as printers, scanners, and copiers, can be configured to be network accessible. Although security of these devices is not generally considered a high priority, the specific functionality of the devices should be considered before they are installed in a network environment. For example, many modern copiers are multifunction devices that can be used as printers or scanners and contain a whole OS. By default, any documents scanned into the device are stored for retrieval on a network-accessible server. Without proper authentication in place, any user with network access to the device can retrieve all documents stored in the cache. Unless the temporary loss of availability of the device or loss of confidentiality or integrity of information processed on the device will have minimal impact on the organization, default passwords should not be used. In some cases, simple office devices are designed without consideration given to user management. For example, only a single administrative account is provided and a centralized authentication system cannot be used, so user credentials are shared between administrators. Since these passwords must be shared by administrators, they should be dedicated to these devices and should not be used for any other devices.

Appendix B—Glossary

Selected terms used in the publication are defined below.

Authentication: The process of establishing confidence in the validity of a claimant's presented identifier, usually as a prerequisite for granting access to resources in an information system.

Brute Force Attack: A form of guessing attack in which the attacker uses all possible combinations of characters from a given character set and for passwords up to a given length.

Capturing: The act of an attacker acquiring a password from storage, transmission, or user knowledge and behavior.

Claimant: An entity that has presented an identity but has not been authenticated.

Cracking: The process of an attacker recovering cryptographic password hashes and using various analysis methods to attempt to identify a character string that will produce one of those hashes.

Dictionary Attack: A form of guessing attack in which the attacker attempts to guess a password using a list of possible passwords that is not exhaustive.

Guessing: The act of repeatedly attempting to authenticate using default passwords, dictionary words, and other possible passwords.

Hybrid Attack: A form of guessing attack in which the attacker uses a dictionary that contains possible passwords and then uses variations through brute force methods of the original passwords in the dictionary to create new potential passwords.

Identification: A claimant presenting an identifier that indicates a user identity for a system.

Keyspace: The total number of possible values that a key, such as a password, can have.

Keystroke Logger: A form of malware that monitors a keyboard for action events, such as a key being pressed, and provides the observed keystrokes to an attacker.

Passphrase: A relatively long password consisting of a series of words, such as a phrase or full sentence.

Password: A secret, typically a character string, that a claimant uses to authenticate its identity.

Password Expiration: The process of forcing a user to select a new password after a certain amount of time.

Password History: The retention of one or more previous passwords or password hashes for comparison against new passwords or password hashes.

Password Management: The process of defining, implementing, and maintaining password policies throughout an enterprise.

Password Management Software Utility: A local utility that allows a user to store usernames, passwords, and other small pieces of sensitive information, such as account numbers.

Password Recovery: The process of a user regaining access to a password that the user has forgotten.

Password Reset: The process of a user having a new password set for a user account.

Password Synchronization: A technology that takes a password from the user and changes the passwords on other resources to be the same as that password, so that the user can use the same password when authenticating to each resource.

Personal Identification Number (PIN): A password that is relatively short (usually 4 to 6 characters) and consists of only digits.

Rainbow Table: A lookup table that contains pre-computed password hashes, often used during cracking.

Reduced Sign-On: A technology that allows a user to authenticate once and then access many, but not all, of the resources that the user is authorized to use.

Salting: The inclusion of a random value in the password hashing process that greatly decreases the likelihood of identical passwords returning the same hash.

Single Sign-On: A technology that allows a user to authenticate once and then access all the resources that the user is authorized to use.

Stretching: The act of hashing each password and its salt thousands of times, which makes the creation of rainbow tables more time-consuming.

Appendix C—Acronyms and Abbreviations

Selected acronyms and abbreviations used in the publication are defined below.

ASCII	American Standard Code for Information Interchange
ATM	Automatic Teller Machine
BIOS	Basic Input Output System
DES	Data Encryption Standard
EFI	Extensible Firmware Interface
FIPS	Federal Information Processing Standard
FISMA	Federal Information Security Management Act
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ID	Identification
IT	Information Technology
ITL	Information Technology Laboratory
LDAP	Lightweight Directory Access Protocol
MAC	Media Access Control
MD4	Message Digest 4
MD5	Message Digest 5
MITM	Man-in-the-Middle
NIST	National Institute of Standards and Technology
OMB	Office of Management and Budget
OS	Operating System
OTP	One-Time Password
PIN	Personal Identification Number
POP3	Post Office Protocol 3
RC2	Rivest Cipher 2
RC4	Rivest Cipher 4
RSO	Reduced Sign-On
SHA	Secure Hash Algorithm
SNMP	Simple Network Management Protocol
SP	Special Publication
SSH	Secure Shell
SSO	Single Sign-On
TLS	Transport Layer Security

TPM	Trusted Platform Module
USB	Universal Serial Bus
VPN	Virtual Private Network