

Password Policy Guidelines



Contents

1	Introduction	1
2	The Role of Passwords in Authentication	1
3	Threats to Password Security	1
4	Making Passwords Hard to Guess	2
5	Making Passwords Hard to Intercept	3

1 Introduction

This document introduces the basic concepts of network authentication. In particular, it focuses on the use of login IDs and passwords to verify the identity of users. Various strategies for selecting strong, hard-to-guess passwords are then discussed.

2 The Role of Passwords in Authentication

Most shared computer systems limit access to data and resources, based on the identity of users who request that access. Access control is therefore dependent on reliable user identification.

Authentication is the process of identifying users in a manner which makes it difficult for one user to impersonate another.

A number of technologies are available for user authentication. The most popular authentication systems are:

- Secret passwords.
- Cryptographic certificates.
- Smart cards.
- Biometric devices (fingerprints, retina scans, head scans, etc.).

Since they are the least expensive to implement, most systems rely on passwords to authenticate users. As well, passwords are often used *in addition to* physical or cryptographic proofs of identity to further strengthen security.

3 Threats to Password Security

A typical case involves a malicious user (M) trying to access a network resource for which M is not authorized. One of the easiest ways for M to access that network resource is to guess the password of a valid user (V).

There are several methods that M could use to guess V's password. First, M could use a computer program to try out possible values for V's password very quickly. M could also acquire V's password by watching as V enters it. M could literally watch V typing, or could use electronic means, such as installing software on V's computer to record his keystrokes, or installing a network analyzer to monitor V's keystrokes as they are transmitted over the network.

4 Making Passwords Hard to Guess

The responsibility of selecting a password that is hard to guess generally falls to users, like V.

If users choose a one-character password, and that character could be any uppercase letter, lowercase letter or digit, then there would be 62 possible passwords. Clearly, M could try all 62 possibilities very quickly.

V could make his/her password harder to guess by using more characters. Using the same possible characters, there are 3844 possible two-character passwords, and 218340105584896 (about 218 trillion) 8-character passwords.

Even if M could try out 5000 eight-character passwords per second, it would take, on average, 700 years for M to guess V's 8-character password. Clearly, longer passwords are more secure!

Unfortunately, V might choose a long password based on something he knows - like his login ID, name or some dictionary word. If V does this, then instead of trying 218 trillion passwords, M could probably guess V's password after a few thousand attempts. If M uses a computer program to guess passwords, this will only take a few minutes.

To decrease the chances of M ever guessing his/her password, V must select a hard-to-guess, or *strong* password. A strong password must:

- Be as long as possible (never shorter than 6 characters).
- Include mixed-case letters, if possible.
- Include digits and punctuation marks, if possible.
- Not be based on any personal information.
- Not be based on any dictionary word, in any language.

While most shared systems can enforce at least some of these rules, almost none have features to enforce all of them.

No matter how many strength rules V uses, though, the persistent M will eventually guess V's password - given enough time. Thus, V must also:

- Change his password regularly, in order to limit the amount of time available to M to guess it.
- Never use the same password twice.

Some systems have a password expiry feature, which forces V to change his password periodically.

As well, some systems incorporate a password history feature, which disallows V from reusing one of his last N passwords.

When faced with a password history mechanism, some users may change their password N times, and return it to its original value, so as to avoid having to remember a new password value. To prevent this, systems should either have an unlimited-length password history, or prevent users from changing their password more than once daily.

5 Making Passwords Hard to Intercept

When a user enters his/her password, it might be intercepted at his/her workstation (by a keyboard monitor program), on the network (by a packet sniffer program), or on the server he is accessing (by a Trojan Horse program).

To protect the user's workstation, a strong operating system must be installed, such as Unix or Windows NT. Furthermore, the workstation must be physically secured against tampering.

If an operating system without security features is used (such as DOS, Windows or MacOS), then an intruder only needs temporary physical access to the console to insert a keyboard monitor program. If the workstation is not physically secured, then an intruder can reboot even a secure operating system, restart the workstation from his own media, and insert the offending program.

To protect against network analysis attacks, both the workstation and server should be cryptographically secured. Examples of strong protocols are the encrypted Netware login and Kerberos. Some systems (like the Windows NT file server protocol – SMB or CIFS) make an attempt at cryptography, but are easily defeated by cryptanalysis. Systems that make no effort to encrypt remote access sessions, such as mainframes and Unix hosts, can be trivially compromised by a network analyzer.

Finally, to protect against Trojan Horse login programs, the server should be physically secured, closely monitored, and should automatically log off unattended sessions.