

ITProTM
SERIES

WindowsITPro  **eBooks**

Keeping Your Business
SAFE from Attack:

Passwords and Permissions

By Roger Grimes

Microsoft[®]



Contents

Chapter 3 Protecting Passwords	50
The Windows Logon	50
Local Logons	51
Domain Logons	52
Password Conflict Resolution	53
Windows Password Uses	54
Computer Account	54
Directory Services Restore Mode Password	54
Service Account Passwords	55
Creating Strong Passwords	56
Password Policy	56
Applicant Identity	58
Secure Authentication Protocol	58
Secure Credential Database	59
System Key Protection	59
Strong Password	61
Password Length	61
Randomness	63
Password Storage	64
Dates of Use	64
Offsetting Protections	64
Password Crackers	66
LOphcrack (LC5)	66
Petter Nordahl-Hagen's Offline NT Password & Registry Editor	67
John the Ripper	68
NTAccess	68
Winternet's Administrator's Pak	68
EBCD-Emergency Boot CD	68
Windows XP/2000/NT Key	68
Austrumi	68
O&O BlueCon XXL	68
Password Best Practices	69

Chapter 3:

Protecting Passwords

This chapter is devoted to a discussion of developing and maintaining strong password systems. It discusses how passwords figure into the Windows logon process, other ways Windows uses passwords, the characteristics of a strong password, how to crack Windows passwords, and password best practices.

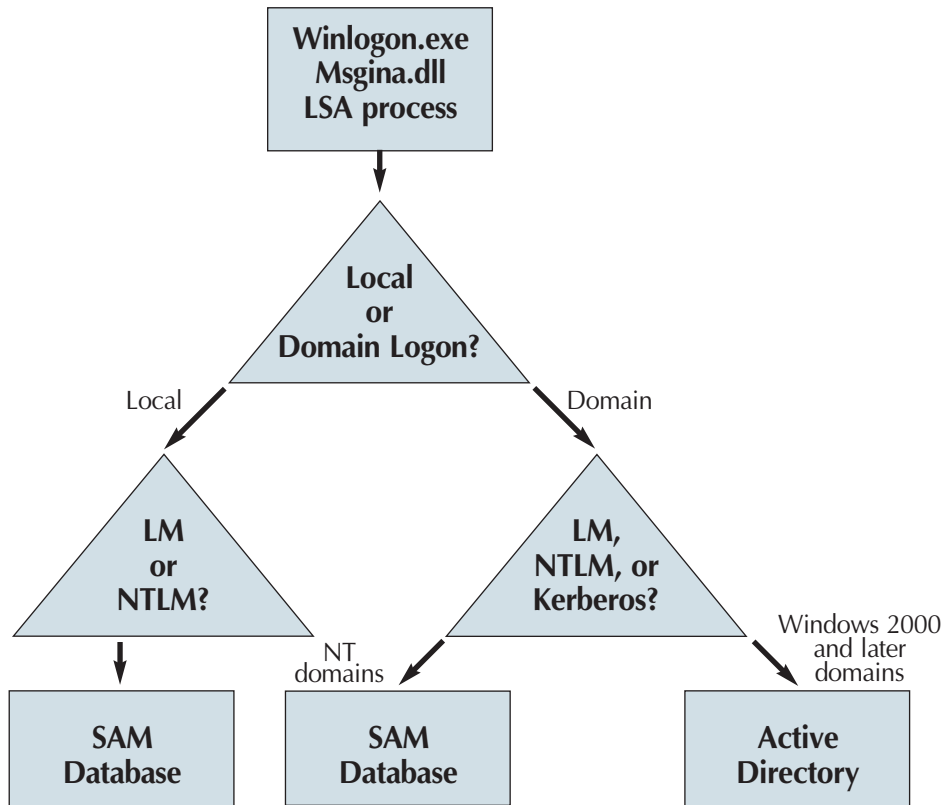
The Windows Logon

As discussed in Chapter 2, “Windows Authentication,” passwords are the most common type of authorization check in Windows, although a security token, such as a smart card, or a Kerberos ticket is also a valid security credential. Passwords are used by security principals — users, groups, and computers — and many different processes, including trusts, clustering, and services. Passwords can be submitted during logon or while accessing a resource; they can be local or submitted over a network.

One of the most common password authentication events is that of a user logging on locally. The user types the logon name followed by the password. The user’s logon name must be unique within the authentication database; also, theoretically, only the user knows his or her password. Thus, a successful logon uniquely identifies the user to the authentication server. After successfully logging on, authentication processes help authorize the user access allowed resources.

During an interactive logon, the Winlogon.exe process is responsible for managing the security-related user interactions with the operating system and for coordinating the logon and logoff processes. Winlogon.exe calls the Msgina.dll (graphical identification and authentication interface), which displays the standard logon box. Msgina.dll prompts the user for the logon name and password, and possibly more information, such as the domain name. The Winlogon process passes the information that is input securely to the Local Security Authority (LSA) process, which determines whether the logon request must be authenticated locally or against a domain database and whether it requires Kerberos or NTLM authentication (see Figure 3-1).

Figure 3-1
Windows Interactive Logon Authentication Pathway



Local Logons

If the local computer name is referenced during the logon, a local logon is initiated. The local credential database is a Security Account Manager (SAM) database. The SAM database is stored in a protected subdirectory that is usually accessible only to the LSA process. The SAM database is in a specially protected part of the registry in the HKEY_LOCAL_MACHINE\SECURITY\SAM subkey and duplicated to the HKEY_LOCAL_MACHINE\SAM subkey. At the file system level, the SAM registry files are stored together with the registry files under %systemroot%\system32\config (SECURITY and SAM files). Many password cracker utilities attack these keys or physical file locations.

The password captured by Msgina.dll is hashed, using the LM or NTLM algorithm (which was covered in Chapter 2), and sent to the NTLM driver (Msv1_0.dll). A challenge-response session is initiated and compared against the stored credentials. Local accounts can be used to access resources only on the local computer where the credentials are authenticated.

Domain Logons

If a domain name is referenced during the logon process, the next step is to determine whether to use Kerberos or NTLM authentication. In Windows 2000 and later, the LSA does this by passing the logon credentials to the Security Support Provider Interface (SSPI). If the user's logon name cannot be found in the Kerberos Key Distribution Center (KDC) domain database, the credentials are resubmitted to the NTLM driver (Msv1_0.dll), which then uses the Net Logon service to complete the authentication process.

The user's domain logon account can exist in only one domain in the Windows forest. Users must successfully log on to their account's "home" domain and one of its domain controllers to be authenticated (setting aside for the moment the existence of cached logon credentials). In a large domain, a user might log on with his or her user principal name (UPN) to ensure that the logon request reaches the appropriate domain and domain controllers. The UPN uniquely identifies the user account in an Active Directory forest. Typically, the UPN is the account's logon name followed by the account's local domain name (although UPNs can be virtually any suffix). For example, Fred Smith's UPN might be fsmith@contoso.com.

The domain credential database is queried to see whether the submitted logon credentials are valid. The domain database can be either a SAM database (NT 4.0 or earlier) or Active Directory (Windows 2000 or Windows 2003 domains). The Active Directory database is a Jet database called NTDS.DIT and is stored by default on domain controllers in the %windir%\NTDS folder (the location is chosen during the Dcpromo). In Active Directory, user passwords are stored in the user object in the unicodePwd attribute; they are Unicode octet-strings that are encoded using Basic Encoding Rules (BER).



Note

Unicode characters can represent several different language and character sets using a single set of ASCII characters.

If the NTLM authentication protocol is used, the Net Logon service queries the SAM database. The Net Logon service passes the user's hashed response through a secure channel to the SAM domain database. In addition, the Net Logon service performs a variety of other functions related to the logon process, such as periodic password updates for computer accounts and domain controller discovery.

If Kerberos is used, the KDC service authenticates logon requests to the Active Directory database (the Kerberos authentication process was covered in Chapter 2). The KDC service runs on all domain controllers (Windows 2000 and above). Any future authentication events are handled by the KDC or each computer's LSA or Net Logon service. Access tokens or Kerberos tickets are created using the security identifiers (SIDs) assigned to the user's account and group memberships. Then, for interactive users, the Windows Explorer shell is started. When a user wants to access an object protected by an access control list (ACL), the user's access token or ticket is analyzed to determine whether the request is allowed.

Password Conflict Resolution

In a (Windows 2000 or later) domain environment, passwords are managed by the domain controller that is acting as the primary domain controller (PDC) Flexible Single-Master Operation (FSMO) role owner for the domain. By default, password changes for user and machine accounts are sent immediately to the PDC FSMO.

In a mixed-mode domain, if a Microsoft Windows NT 4.0 domain controller receives the request, the client is sent to the PDC FSMO role owner (which must be a Windows 2000- or 2003-based computer) to make the password change. This change is then replicated to other Windows 2000 domain controllers using Active Directory replication and to down-level domain controllers through the replication process.

If a Windows 2000 domain controller receives the request (either in mixed or native mode), the password change is made locally and sent immediately to the PDC FSMO role owner using the Net Logon service in the form of a Remote Procedure Call (RPC). The password change is then replicated to its partners using the Active Directory replication process. Down-level domain controllers replicate the change directly from the PDC FSMO role owner (or from a replication partner if they do not directly replicate from the PDC FSMO role holder).

During logon, if a user's (or computer's) logon name is found in the credential database, but the passwords do not match the domain controller's local copy of the Active Directory database, the PDC FSMO is contacted for resolution. If the collected password matches the one stored at the PDC FSMO, then the logon authentication is allowed and the domain controller denying the original request is updated. If the passwords don't match, the PDC FSMO indicates the failure in its return code to the domain controller, and the logon fails. Because the PDC FSMO is the ultimate arbitrator of password synchronization, it is the service that determines when to lock out an account because of invalid logon attempts with incorrect passwords.

Therefore, in a distributed domain environment with more than one domain controller, administrators need to pay special attention to password changes. Password resets done at the PDC FSMO may take time to replicate to the domain controller that the user is logging onto. You can bypass the password synchronization lag problem by resetting the password on the domain controller (if you know it) where the user is logging on. To find out which domain controller the user is logging onto, you can audit Logon Account events or use one of several other Microsoft utilities.

To reset the password on a particular domain controller, you connect to it (in Active Directory Users and Computers) first — before resetting the password. This way, the user will be able to take advantage of the password reset immediately, and the new password is then propagated to the PDC FSMO. Of course, if the affected domain controller is not a direct replicating partner of the PDC FSMO role holder, the change will be replicated from the domain controller to its replication partner and eventually make its way to the PDC FSMO holder, depending on your replication topology.

**Caution**

Resetting passwords can cause a user's EFS files to be inaccessible. EFS-protected files are encrypted using a user's EFS public key. Decrypting requires the user's EFS private key. The user's EFS private key is stored in the user's local profile (and on the server if roaming profiles are used). The EFS private key is protected with another "master key," which is associated with the user's password. If a password is reset by a third party, the user's master key can no longer unlock the EFS private key, which leaves the EFS-protected files unavailable (unless other data recovery methods were implemented).

Windows Password Uses

Password credentials are used throughout Windows, more than only for user logons and authentication events. Passwords are created for computer accounts, services, trusts, and all sorts of other events that require authentication. Many users don't fully understand these other common accounts with passwords.

Computer Account

Each computer that accesses a domain must have a valid logon name and password. Usually, Windows sets and updates computer account credentials without user intervention. The machine account is authenticated, both as an additional security measure and to establish a secure channel between the workstation and the domain controller. The secure channel is used for authentication but is not encrypted.

Each Windows-based computer maintains a machine account password history that contains the current and previous passwords for the account. When two computers try to authenticate with each other and a change to the current password has not yet been received, Windows relies on the previous password. If the sequence of password changes has occurred more than twice, the computers involved may not be able to communicate, and you may receive error messages. For example, you may receive an "Access Denied" error messages when Active Directory replication occurs. This behavior also applies to replication between domain controllers of the same domain.

You cannot change the machine account password with the Active Directory Users and Computers snap-in, but you can reset the password with the Netdom.exe tool. The Netdom.exe tool is included in the Windows Support Tools (on the install media located in \SUPPORT\TOOLS\SUPPORT.CAB). The Netdom.exe tool resets the account password on the computer locally (known as a *local secret*) and, at the same time, writes this change to the computer's computer account object on a Windows domain controller that resides in the same domain. Simultaneously writing the new password to both places ensures that at least the two computers involved in the operation are synchronized. Normal Active Directory replication ensures that the other domain controllers receive the change.

Directory Services Restore Mode Password

When a member server is promoted to a domain controller (i.e., using Dcpromo.exe), accounts in the SAM, along with a new set of default users and groups, are migrated to the jet-based Active Directory

database. A new registry-based SAM, containing an “offline” administrator account and other built-in accounts needed to recover and manage the domain controller, is created. The original local accounts are no longer available or accessible in the local SAM database.

The new SAM-based accounts and passwords are computer-specific and are not replicated to other domain controllers in the domain. They are available only in *Directory Services Restore* mode, accessed by pressing F8 in the early part of the Windows boot process (which takes Active Directory offline). During the domain controller creation process, the user is prompted for a Directory Services Restore mode administrative account password. This password is used only when starting the domain controller in Directory Services Restore mode or when using the Windows Recovery Console. More important, password changes to the domain administrator account have no effect upon the Directory Services Restore mode password.

Because this newly created administrative account is rarely accessed, and many administrators don’t understand its implications, the password is often forgotten or lost. You can change it in Windows 2000 using Microsoft’s Setpwd.exe utility (<http://support.microsoft.com/default.aspx?kbid=810037>) or in Windows 2003 using Ntdsutil.exe (<http://support.microsoft.com/kb/322672>).

Service Account Passwords

Windows services must log on to the operating system to begin operation. Services can use one of the built-in service accounts in Windows (Local System, LocalService, or NetworkService) or a custom-created service account. In simple terms, a service account is nothing more than a normal user account that is granted the *Logon as a Service* user right. This right lets the service account log on to the operating system (under the auspices of the Service Control Manager) when Windows is booted without waiting for the actual user to log on; this right also confers additional rights for interacting with the operating system.

The default built-in service accounts do have passwords, but they are managed by the operating system. Custom-created service accounts must have credentials supplied to the service (which is then stored in Service Control Manager database located in the registry at HKEY_LOCAL_MACHINE \SYSTEM\CurrentControlSet\Services) and also set up in the normal user credentialing database (SAM or Active Directory). Passwords for custom-created service accounts should be strong and should be changed more frequently than passwords for accounts with lesser privileges. Although there is no general rule of thumb, many hardening guides recommend changing passwords at least monthly. Password changes must be synchronized between the service’s logon properties and the credential database. You can synchronize them manually, by creating your own script (check http://msdn.microsoft.com/library/default.asp?url=/library/en-us/ad/ad/changing_the_password_on_a_serviceaccount_user_account.asp), or by using one of the scripts widely available on the Internet.

Of course, many other types of passwords are used throughout Windows, depending on your configuration and services. Other services that require passwords include Windows trusts, clustering, Microsoft Credential Manager, remote access, telnet services, web sites, and SQL server passwords. Passwords are even required to unlock screen savers. Anywhere passwords are used in Windows, it is best to use strong passwords, which leads us to the question, “What does ‘use strong passwords’ mean?”

Creating Strong Passwords

The exact definition of what is and isn't a strong password is a subject of debate, especially if you're looking for specific and exact figures. Older textbooks and discussion papers often select an absolute minimum password size and paint that size as constituting a strong password. Conventional wisdom says that a password's length should increase as the value of the asset it protects increases. However, the truth is that strength depends on more than only the password itself and is certainly more complicated than the single characteristic of password size. For example, experts agree that a weak password is one that can be easily guessed.

A strong password is one that can outlast expected and near-term future attacks. It is much like a building with an alarm system and locked glass doors. An alarm system with an audible alert is never designed to stop intruders – it's a defense-in-depth countermeasure to alert other dissuaders. Similarly, locked glass doors won't stop a sufficiently motivated attacker for long. Instead, they're intended to stop most attackers from the attempt in the first place and to stop the dedicated intruder just long enough for the cops to show up.

A good password system works the same way. An appropriate password system will discourage or prevent most intruders from attacking in the first place; it will slow down those would-be intruders who do attack. A password has to be strong enough only to last until another offsetting control can actively participate – or until the protected asset has limited value. That last statement may sound strange, but consider this: confidential World War II secrets only had to outlast the war. Yesterday's secret ciphers are today's classroom examples.

The strength of a password should flex according to the value of the asset it is protecting. The password that protects this document's transportation during its pre-publication phase is not nearly as important as a password protecting an Active Directory domain database for a large company.

A strong password requires a strong password system. A strong password system is made up of

- A strong, enforced password policy
- The reliable identification and verification of applicants
- A secure authentication protocol
- A secure credential database
- And yes, a secure password

Let's consider each element in turn.

Password Policy

A password policy establishes minimum password system characteristics as discussed and defined by management. A password policy should be written, approved, and shared with users. It should detail the characteristics of a strong and secure password, such as those listed in the next paragraph. In short, a password policy should control user behavior.

For example, passwords should never be shared, not even with IT help desk support (unless they're changed immediately). Users who freely share their passwords should be penalized. Passwords should not be posted in public places or documented in a way that can allow for easy disclosure. Passwords should not be made up of words that could easily be associated with the user. Passwords should not be named after loved ones, pets, or sports teams. A user who learns of a password compromise should report it to the appropriate team members.

However, password policy should cover more than just computers and more than Windows operating system passwords. Company should require passwords for cell phones, PDAs, PBX remote

maintenance consoles, fax machines, copiers, and in any other place where individual auditing and access control is appropriate. Passwords should be required on bootup for laptop and PDAs to protect confidentiality; they should also be required for PC BIOS changes. A good password policy should minimize the number of privileged accounts that can change or reset passwords, unless users are supposed to take care of their own password changes.

A password policy should recommend that logon credentials should not be shared among users, including IT users. For example, it is a common practice in some companies to have computers that stay logged on with the same user name (e.g. Front Desk or Accounting1, etc.), regardless of who is actually using the computer. This practice should be discouraged. Even if the logon name is identical, each user should have a unique password. Similarly, the domain administrative password should not be the same as local administrative passwords. BIOS passwords should never be the same as operating system passwords. The Directory Services Restore mode password should be different, too. Users should be encouraged to use different passwords for different applications and web sites. Failure to do so could result in a more sensitive compromise.

Windows authentication supports a default mechanism, *pass-through authentication*, that can cause problems if logon credentials aren't different. If a login-account-and-password combination is identical in two domains or forests, Windows assumes you want to manage the objects in the destination resource domain with the credentials of the security principle in the destination domain or forest — even though you began by using the source domain's credentials. It will appear as if the resource access in the destination domain is occurring from the source domain security principal account, even though it is not.

For example, assume I have two Active Directory forests, Forest A and Forest B, on the same IP subnet but without a Windows trust. Theoretically, an administrator in Forest A should not be able to manage and access resources in Forest B, because they have two different security principal accounts. But if the two security principal accounts are the same, Windows will assume I want to use the destination security principal's credentials and give me access. This Windows "feature" is an unintended security credential collision that probably, in most cases, is desired by network administrators.

Unless good password policy is communicated to employees, they will likely use the same password at work and to access untrusted websites on the Internet. On a weakly protected web site, a hacker could discover a user's password and then use it to attack the user's online bank account — or the company's payroll database.

When you have created and communicated a password policy, you should enforce it. In practice, enforcement can be hard. For example, you might want to enforce a policy that passwords must have more than 15 characters so that the LM hash can't be stored. However, Active Directory and Local Computer Policy allows the minimum password length to be 14 characters (this limit will be increased in the future). Even if you require a certain size password or a certain combination of characters, some systems do not support them. Many web sites cannot handle passwords with more than eight characters. Some mainframe systems have a maximum password size of 6 characters. Some operating systems don't handle case-sensitive passwords. Some Unicode or control characters are not available on all keyboard language sets.

Regardless of the policy and enforcement method, passwords should be periodically audited (manually or using automated tools) to ensure compliance. No matter how strong the rest of the password system is, a single password can be compromised by lazy policy enforcement.

For more information about developing a password policy, see

- “Password Policy” on the SANS (SysAdmin, Audit, Network, Security) Web site at <http://go.microsoft.com/fwlink/?LinkId=22205>. SANS has advice about creating formal corporate security policies and sample policies.
- “Sample Generic Policy and High Level Procedures for Passwords and Access Forms” on the National Institute of Standards (NIST) Web site at <http://go.microsoft.com/fwlink/?LinkId=22206>. NIST has a sample password policy that many government agencies have used.
- “Account Passwords and Policies” on the Microsoft TechNet Web site at <http://go.microsoft.com/fwlink/?LinkId=22208>.

Applicant Identity

If a password credential is to be worth anything, the user’s identity needs to be independently verified before it is associated with a credential set (logon name and password). If any person can call to get a password or request a password change, you have poor password control. All password requests should be accompanied by some form of third-party verification. Employees should request password-related changes in person; the requests should be approved by management or verified by some other process (for example, the administrator may recognize the applicant). The process for identifying applicants should be covered in the password policy. Universal enforcement of the process reduces fraud from social engineering attacks.

Secure Authentication Protocol

As covered in Chapter 2, using a reliable and secure authentication protocol is significant. Weak authentication protocols lead to trivial exploits and compromises. For that reason, use of the LM and NTLM (not NTLM version 2) authentication protocols should be discouraged and they should be disabled wherever practical.



Note

You should disable LM and NTLM only if you aren’t using any applications and services relying on those protocols. Most Windows password recovery tools rely upon the existence of LM or NTLM password hashes to be successful.

All newer Windows operating systems are capable of authenticating using LM, NTLM, NTLMv2, and Kerberos protocols. Kerberos and some version of NTLM must be used. Disable the use of LM and NTLM (version 1.0) protocols, and disable LM hashes. After you disable LM hashes (according to the recommendations in Chapter 2), force all users to change their passwords; you can use the query feature in Active Directory Users and Computers to make all user passwords expire at the same time. This additional step is needed because the current password’s hash stays in storage, even when LM hashing is disabled. After users change their passwords, the old hash remains; however, if a hacker recovers the hash, the current password is not recovered. Of course, the user’s new password should not be similar to the old password.

Secure Credential Database

The credential database that holds the logon accounts and passwords should be secure. The first step in database security is ensuring physical security. To be successful, most Windows password cracking or resetting tools require local access to the Windows password database, and most require the ability to boot around the operating system protections that are normally in place. Simply preventing unauthorized local access to the Windows password databases will thwart most password cracking or resetting attacks. Also, you should disable the ability to boot from any removable media drives, such as floppy disks, CD-ROMs, or USB drives. In fact, you should disable removable media from even being considered as a boot-up option in the ROM-BIOS and then password-protect the ROM-BIOS settings from alteration.

Consider setting up credential database servers to require boot-up passwords or password tokens to start. Some high-security web sites tie boot-ups to the successful validation of an externally located device. For example, the entire hard drive may be protected by an external cryptographic system in which the hard drive is “encased” with a cryptographic cipher. Booting the hard drive requires one or more smart cards, with appropriate digital keys and manually input passwords. Verisign (<http://www.verisign.com>) protects their most valuable key management servers this way. Two smart cards, one carried by an IT employee and one carried by management, are necessary for the server to boot.

System Key Protection

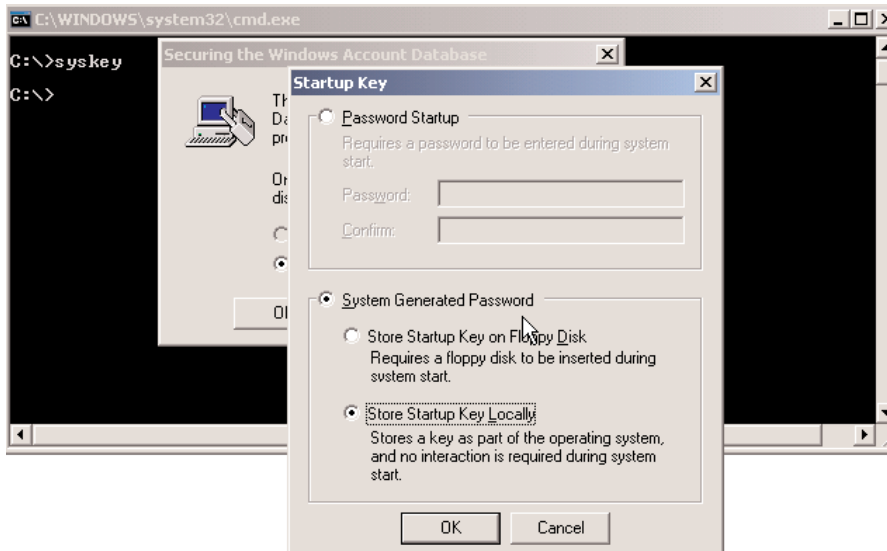
Security credential databases, even if locally compromised, should prevent non-trivial password compromises. To start with, passwords shouldn't be stored in plain-text in an easily accessible database. Windows meets this standard by storing all passwords as hashes and then storing even the password hashes in encrypted form.

Since Windows NT, Microsoft's Syskey utility safeguards SAM databases (but not Active Directory) from physical attacks. The SAM database stores hashed copies of passwords. This database is encrypted with a locally stored system key. To keep the SAM database secure, Windows requires that the password hashes be encrypted. Windows prevents the use of stored, unencrypted password hashes.

You can use the Syskey utility to provide extra security to the SAM database; move the SAM database encryption key off the Windows-based computer or require a start-up password that decrypts the system key so that Windows can access the SAM database. By default, Windows Server 2003 enables Syskey, which requires a locally stored key to access the SAM database. You can configure Syskey to require the additional protection of one of the remotely stored key settings. To do so, follow these steps:

1. At a command prompt, type **Syskey**, and then press Enter.
2. In the Securing the Windows Account Database dialog box (see Figure 3-2), note that the Encryption Enabled option is already selected and is the only option available. When this option is selected, Windows always encrypts the SAM database.

Figure 3-2
Configuring Syskey



3. Click Update.
4. If you want to require a password to start Windows, click Password Startup. Use a complex password that contains a combination of uppercase and lowercase letters, numbers, and symbols. The startup password must be at least 12 characters long and can be up to 128 characters long.



Note

If you must remotely restart a computer that requires a password (if you use the Password Startup option), a person must be at the local console during the restart. Use this option only if a trusted security administrator is available to type the startup password.

5. If you do not want to require a password, click System Generated Password. Then select one of the following options:
 - Store Startup Key on Floppy Disk — this option requires that someone insert the floppy disk containing the encryption key to start the operating system. This choice provides the highest level of protection for the SAM database.
 - Store Startup Key Locally — the encryption key on the hard disk of the local computer. This is the default option. Click OK two times to complete the procedure.

If you use the Store Startup Key on Floppy Disk option, always create a back-up floppy disk. You can restart the system remotely if someone is available to insert the floppy disk into the computer when it restarts.



Note

The Microsoft Windows NT 4.0 SAM database was not encrypted by default. You can encrypt the Windows NT 4.0 SAM database using the Syskey utility.

Syskey is a wonderful utility, but its protection applies only before the Windows operating system is booted. After the Windows operating system boots Syskey provides no additional protection. Further, several Windows password cracking utilities (discussed below) can disable Syskey and its boot-up protections.

Strong Password

Finally, a secure password system requires a strong password. The exact makeup of a strong password depends on the value of your data, the strength of the password management system protecting the password, and the types of hacker tools available to attack your system. Here are some characteristics of a strong password:

- Length
- Randomness
- Secure storage
- Dates of use
- Offsetting protections

Password Length

A password must be long enough to discourage password cracking attacks. Types of password-cracking attacks include

- Brute-force guessing
- Dictionary or word list attacks
- Birthday attack
- Pre-computed hash attacks

Brute force attacks begin attacking the password by guessing every possible password sequence, starting with the lowest available character, and cycling through all possible symbol sequences until a correct password is found. This strategy is the worst possible way to guess a password. NTLM and Kerberos passwords can be made up of 65,536 different character symbols. Starting with one-character passwords and trying all 65,536 characters, and then trying all two-character password combinations, and so on. Very quickly, the password cracker is weighed down by the sheer volume of potential password sequences.

To put it in perspective, even with all the foreseeable computing power and hard drive space that will be available in the next hundred years, it would take decades to find a random match for one single password. The data being protected by that password has a greater chance of losing its value than a cracker has of finding the password – assuming it didn't change in that time.

It makes more sense to guess only those passwords that might be used. Here's where the process can get interesting. Dr. Jesper M. Johansson's paper, *The Great Debates: Pass Phrases vs. Passwords: Part 1 of 3* (<http://www.microsoft.com/technet/security/secnews/articles/itproviewpoint091004.mspx>), provides some important data. First, although it is possible to use 65,536 different

Unicode characters in a Windows password, most passwords use just 32 different characters – the 26 lowercase alphabetic characters, a handful of uppercase alphabetic characters, and the non-alphabetic exclamation point symbol (!). A cracker could limit the attack to those 32 characters, plus maybe the pound (“#”) and at sign (“@”) to be inclusive, reducing the work from churning through 65,363 symbols to checking only 34 symbols. Most Windows passwords have eight or fewer characters (in fact, most administrators require a minimum length of only five characters), so instead of guessing at all possible password lengths, from zero to 127 characters, hackers can further reduce the scope of their tries. Again, guessing 35 different characters in a password of 5 to 8 characters is significantly easier to compute than 65,536 characters at every combination between 0 and 127.

A birthday attack takes password cracking one step further. Statistical models show that although each person has a 1 in 365 chance of being born on a particular day of the year, in a random group of 23 people, there is a 50% chance that two people share the same birthday. This statistical phenomenon means that a randomly generated guess at the password will probably arrive at the correct password significantly faster than a sequential search from either the beginning or the end of the available possibilities.

Instead of guessing the plain-text password, the pre-computed password hash attack starts in the other direction. First, an attacker creates a database of all the possible passwords and their hashes. These databases are also known as *rainbow tables*. When a password hash is found, it is directly matched to a pre-computed hash and its plain-text counterpart in the database. The weakness of this attack is the amount of storage required — all the possible pre-computed hashes can easily take up gigabytes. Small rainbow tables are at least a few hundred megabytes and larger ones are over 24 GB (<http://www.antsight.com/zsl/rainbowcrack>).

In fact, large rainbow tables don't really contain all the possible password hashes for a Windows system. Even if you limit password hashes to weak 8-character (or less) LM hashes, pre-computing all the possible combinations requires more than 62 GBs of storage space. If users take advantage of the total number of available password characters available in Windows, even relatively short passwords (8 characters or less) require rainbow tables bigger than all the available drive storage in existence today. Unfortunately, because most users use passwords with easy-to-guess character sequences, rainbow tables can be significantly smaller and still be very accurate.

It's still easier to get a password by asking the user for it or even knowing that the user has a weak password. Social engineering attacks are far more adept at gaining access to passwords than are the (over-romanticized) efforts of a remote password cracker compromising a domain database. One of the best social engineering scams occurred at a national university. A student posted his dorm phone number on a flyer as the IT Help Desk number. Within days he had “helped” dozens of unsuspecting users and helped himself to their logon credentials. As stated in Chapter 1, in several studies over the years, most users will share their passwords with strangers on the street for prizes like pens and chocolate candy. Why hack hard when you can hack easy?

Bottom line: how many characters are necessary for a good password? Passwords should be at least six characters, to even begin to approach a low minimum number. Each character adds significantly more computations to a password cracking attack. Some security guides say that eight is the magic length; others say nine. Windows security experts often recommend passwords of 15 or more characters simply to disable LM hashing. If you don't have LM hashing disabled globally in your environment, this recommendation good. One Microsoft security analyst recommends a minimum length of 42 characters (http://blogs.msdn.com/robert_hensing/archive/2004/07/28/199610.aspx).

As you can see, there are no hard and fast rules. Anyone who tells you a rule without asking you about the rest of your password management system isn't getting enough detail. It is known that that every increase in password length, if the password symbol is random enough (more on this below), will result in a stronger password. Of course, if passwords become too long and your users revolt, then the negatives offset any security gains.

Randomness

For a password to be strong, and to defeat password cracking tools, it must be random. Using a spouse's name as a password is not random. As discussed above, most people don't vary the range of characters they use. Although 65,536 different Unicode symbols can be used in a Windows password, most users choose from among a set of about 35. The actual level of randomness in their passwords is significantly less than the theoretical possibility.

Lately, there has been a big push for the use of passphrases instead of passwords. The concept is that multiple-word passphrases are longer than average passwords, thereby making the job that much harder for password crackers. However, this holds true only if the passphrase is made up of random symbols; however, most passphrases are made up of English words. Johansson's paper demonstrates that random 9-character passwords are roughly as secure as 5- to 6-word passphrases – although a password may use random characters, a passphrase is more likely to use very common English words. The *entropy* (a measure of randomness) of the words in a passphrase is less than the entropy of a randomly chosen password. The conclusion of his paper, and many others, is that if passphrases become commonplace, without any additional randomness, they will be no better than significantly shorter passwords. Instead of using individual characters or symbols, password-cracking tools will simply use whole English words instead.

Asking a user to create a random password on their own, especially if the requirements are not enforced, is futile. Most passwords will be 6 or fewer characters, chosen from that set of 32 characters most likely to be used. Microsoft Windows 2003 requires a slightly more complex password, using the following rules:

- The password must be at least six characters
- The password cannot contain more than three characters from the user's account name
- The password can contain English uppercase characters (A through Z)
- The password can contain English lowercase characters (a through z)
- The password can contain numerals (0 through 9)
- The password can contain non-alphabetic characters (such as !, \$, #, %)

Further, a Windows password must contain at least three of the four character groups listed in the last four bullet points. You can turn off Windows password complexity using a group policy object setting. Changing the specific complexity requirements requires custom programming.

Strong passwords include both length and randomness. Some users create passphrases that include English words with intentional misspellings or symbol replacement. For example, the passphrase, "I'll always love our time in paris" can be typed in as "I'l1 Always Love OuR T1M3 n P@r!5". As complicated as the example seems, and as difficult as it is to type, its random element is still not too strong for future passphrase crackers. Still, it (or something like it) will trump a short six- to eight-character password every time.

Password Storage

Having many complex passwords can be a memory problem, especially when the passwords aren't used often (for example, for service accounts or Directory Restore mode). Writing them down is an obvious solution, but is it possible to write down passwords securely? A crude method is to store the written password in a secure location, such as a fireproof vault, preferably in two locations for redundancy.

A more high-tech solution is a password storage program. Password storage programs are small databases that store passwords securely. By remembering only one complex password, you can always retrieve all your important passwords. Many password storage database programs are available, including the popular Password Safe (<http://passwordsafe.sourceforge.net>).

Dates of Use

Another way to help protect a password is to limit the time period when it is used. The more valuable the data and the higher the risk, the shorter the time period should be. Again, although there is no hard and fast rule, most companies follow some general rules of thumb. Environments or accounts at high risk should set maximum password age to 30 days or less. Companies that can tolerate more risk can extend maximum password age to 3 to 6 months. Pick a maximum password age that fits your environment.

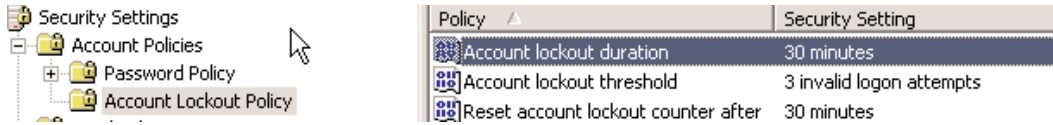
You enable the minimum password age and password history settings in Windows. The Minimum Password Age setting imposes a minimum length of time a password must be used before changing the password. The Password History setting instructs Windows how many old passwords to remember as the user chooses a new password. Windows does not let a user re-use a password that is in the user's password history list. By enabling both minimum password age and password history, an administrator can ensure that users don't change the password several times in a row simply to re-gain the original password they liked.

Remember, different accounts with varying levels of risk require different maximum password ages. Unfortunately, Active Directory currently only allows one password policy per domain, but this is scheduled to change.

Offsetting Protections

The defense-in-depth philosophy applies to password policies. Windows contains several password policies that limit the exposure of passwords to cracking attempts. The single biggest step any administrator can take is to enable account lockouts. Enabling lockouts ensures that any account that experiences a certain number of invalid login attempts during a given time period (all customizable) is disabled. Like password policies, account lockouts (see Figure 3-3) can be set at the local or domain level. Set to almost any threshold settings, account lockouts are an excellent way to deter password cracking.

Figure 3-3
Account lockout settings



Without account lockouts, most password crackers operating over a network are limited to 2 or 3 password guesses per second because of the way network logons against a single logon account are processed by Windows; a locally installed password guessing tool could try hundreds to thousands of guess per second. Enabling account lockouts significantly slows any progress a password cracking tool could make against an active database.

A word of caution: although you can require that the administrator unlock any disabled accounts, doing so increases network support costs — some surveys say a single password reset or account unlock costs \$75 — or could lead to a denial of service attack. Hackers have been known to attack networks, intentionally guessing bad passwords to disable as many accounts as they can. Often the best cost-benefit tradeoff is to re-enable disabled accounts automatically after a short time (for example, 5 minutes). Even if you lock out the account only for a few minutes, you've defeated most online password hacking tools that try to brute-force the database.

Windows Kerberos goes one better and requires pre-authentication before a password guess can be made. Pre-authentication is a challenge-response process in which the Kerberos password cannot even be submitted for review at the KDC unless a pre-logon challenge is answered correctly. The challenge can be answered correctly only by a Winlogon.exe service that already has the correct password hash. Therefore, to guess the password you have to know the password. Without a successful pre-authentication challenge, a brute force tool would not even be allowed to take a guess.

Another offsetting mechanism is to limit the number of times and the computers at which a logon account can be used. For example, most custom service accounts are needed on only one computer at a time. Because these accounts also usually carry high privileges, limiting the number of their potential logons, and limiting them to only the machine they are needed on, significantly minimizes risk.

Consider renaming highly-privileged default accounts. Give your administrator account a different name. Make it look like a normal end-user account. Change the account description, even. Rename the administrators group. Renaming these accounts and groups won't change their permissions, but it will frustrate password cracking tools that don't know you've renamed them. Of course, hackers who can do SID enumeration can still find the highly privileged accounts, but you can turn off anonymous SID and account enumeration, and most hackers don't even try that anyway.



Note

Sometimes, a renamed administrator account is still referred to as its original default name. For example, if you boot up into the Windows recovery console, it will ask for the *administrator* password even though the account is renamed.

If the original account name is missing or disabled when a hacker tries to use it to log on, a message indicating the account's true status will be returned to the cracker. Therefore, to frustrate password crackers even more, create new accounts with the old accounts' original names but with highly restricted privileges, Full Control-Deny on all NTFS permissions, and complex passwords. Crackers will burn all their CPU cycles trying to break a tough password, and even if they do, they've still got less access than is given to the guest account.

As is the case when implementing any security recommendation, experiment and test before implementing changes to your password policies. Changing password sizes and complexity can have operational and end-user emotional effects that are outside the control of the network administrator. Operationally, dozens of documented problems can occur as you increase password length and complexity. Some of the problems are

- Clustering passwords must be greater than 14 characters (15 or larger) if LM hashes are disabled; otherwise, newly joining cluster nodes will try to join using an LM hash first (see Microsoft Knowledgebase article# 828861)
- Disabling LM authentication causes problems with legacy operating systems
- Third-party SMB shares often require simple or plain-text passwords

Password Crackers

Windows operating systems are highly resistant to password cracking attacks if you take simple preventative steps. Most attack tools rely on local access to the computer or the presence of the LM hashes. Hackers with local access to a PC can do anything. Forget about stealing or resetting passwords; they can take the whole PC. Most Windows password crackers are successful only on older versions of the operating system; just a handful can crack Windows Server 2003 password databases. Even then, most of those tools crack only the local SAM database and won't even attempt to crack domain logon credentials.

In this section, we review some common password cracking tools – both free and commercial, both password recovery utilities and password re-setters. Here are some other general observations:

- Password crackers are more successful on weak, short passwords.
- Most password crackers are really password re-setters, which allow passwords to be changed, not recovered. They might cause a successful intrusion, but you would probably detect it.
- Password dictionary attacks require a word list database. Although most databases are sufficient to guess most of today's passwords, adding just a little complexity and length to your passwords will defeat most of them.
- To be successful, online dictionary attacks require that account lockouts be disabled.
- To be successful, offline dictionary attacks require LM hashes.
- Most password cracking tools assume passwords have 14 or fewer characters.
- Most password cracking tools do not work against a Windows computer secured with the best practices recommended at the end of this chapter.

Now, let's review a few popular password cracking tools.

LOphtcrack (LC5)

You can find this password recovery tool at <http://www.atstake.com/products/lc>. LOphtcrack, now called LC5, is a commercial tool recently purchased by Symantec. LOphtcrack is the best Windows

password-cracking tool available. It can work locally or remotely. It first recovers password hashes and then attempts dictionary attacks against the recovered hashes. It can capture passwords over the network in “sniffing” mode, but it cannot break NTLMv2 or Kerberos authentication. It can recover Active Directory accounts. It requires an administrative level account to work. If you need to do Windows password cracking or auditing for a living, this is the tool.

Petter Nordahl-Hagen’s Offline NT Password & Registry Editor

Find this password re-setter at <http://home.eunet.no/~pnordahl/ntpasswd/bootdisk.html>. It is a free, self-contained Linux boot diskette image with fairly easy-to-understand, scripted menu commands. It will work on all versions of Windows SAM databases, but it cannot access Active Directory. It can reset and blank passwords but not recover them, and it cannot reset domain passwords. Nordahl menus are similar to the one shown in Figure 3-4.

Figure 3-4

Example of the Nordahl password reset screen

```

=====
. Step THREE: Password or registry edit
=====
chntpw version 0.99.2 040105, (c) Petter N Hagen

[.. some file info here ..]

* SAM policy limits:
Failed logins before lockout is: 0
Minimum password length      : 0
Password history count       : 0

<=====> chntpw Main Interactive Menu <=====>

Loaded hives: <sam> <system> <security>

 1 - Edit user data and passwords
 2 - Syskey status & change
 3 - RecoveryConsole settings
   - - -
 9 - Registry editor, now with full write support!
 q - Quit (you will be asked if there is something to save)

What to do? [1] -> 1

===== chntpw Edit User Info & Passwords =====

RID: 01f4, Username: <Administrator>
RID: 01f5, Username: <Guest>, *disabled or locked*
Select: ! - quit, . - list users, 0x<RID> - User with RID (hex)
or simply enter the username to change: [Administrator]

```

John the Ripper

This password recovery tool is available at <http://www.openwall.com/john>. It's a free password cracker ported from Unix that requires local access and weak password hashes; it works only on SAM databases. Basically, it compares an input file against an inputted word list database, using the related cryptographic routines. To use it, you must first extract the password hashes from the local SAM using another tool, such as Todd Sabin's Pwdump2 or Phil Staubs' Pwdump3 (<http://www.polivec.com/pw3dump/default.htm>), both of which require administrative credentials. However, after you have extracted the LM password hashes, you can use John the Ripper to convert to their resulting passwords.

NTAccess

A password re-setter, NTAccess is available at <http://www.mirider.com/ntaccess.html>. This commercial product, made by Dieter Spaar, resets the local administrator password. It's notable because it can locate a renamed administrator account and enable a disabled administrator account. It requires local access and cannot defeat offline Syskey passwords. It does not work with Windows Server 2003 domain accounts, but it claims to be successful with NT and 2000 domain controllers. It requires Windows boot floppies.

Winternals' Administrator's Pak

This password re-setter, available at <http://www.winternals.com/products/repairandrecovery/index.asp?pid=ap>, is a commercial product that can do a whole lot more than reset password resets. It's a beautiful tool with nice GUIs. It requires local access to the system, works only on local administrator passwords, and requires access to the SAM registry hive.

EBCD-Emergency Boot CD

You can find this password re-setter at <http://ebcd.pcmintistry.com>. It is open source and contained on a Rescue Linux distribution. It requires local access and only resets local SAM account database passwords.

Windows XP/2000/NT Key

This commercial password re-setter is available at <http://www.lostpassword.com/windows-xp-2000-nt.htm>. It needs Windows install boot diskettes to work. It can only reset passwords, but it claims to reset domain administrator passwords, too. It works with Windows Server 2003.

Austrumi

An open-source password re-setter, Austrumi is available at <http://sourceforge.net/projects/austrumi>. It's an open source bootable Linux image. It resets only local SAM account databases.

O&O BlueCon XXL

You can find this commercial password re-setter at <http://www.oo-software.com/en/products/oobluecon/index.html>. It can reset local SAM passwords but requires local access. It claims to work with XP, 2000, and NT, but it does not mention Windows Server 2003.

Using password cracking tools isn't always a bad thing. In fact, they make great password auditors. Password re-setters are helpful if you forget or lose the local administrator's password. Only two of the tools, L0phtcrack and Windows XP/2000/NT Key, claim to work with domain accounts. Most of the tools are simple password re-setters that operate on local SAM accounts. Only L0phtcrack and Jack the Ripper claim to do password recovery. However, even these recovery tools can be stopped by passwords that are longer and more complex than is normally expected.

Password Best Practices

Microsoft's latest operating systems are resistant to password cracking attacks. By requiring NTLMv2 authentication, disabling LM hashing, disabling booting from removable media, and increasing password length and complexity, you can significantly diminish the risk of password hacking attacks. Here is a recap of recommended best password practices, plus a few new ones.

- Create and enable a strong password policy
- Disable booting from removable drives
- Use Syskey
- Enable account lockouts
- Increase password length and complexity, and shorten the time they can be used
- Enforce a minimum password age
- Enable password history tracking
- Disable LM hashes
- Force NTLMv2 authentication
- Use EFS
- Create different passwords for different accounts and uses
- Rename sensitive accounts
- Enable auditing on the following audit categories: Audit Account Logon Events, Audit Account Management, Audit Logon Events, and Audit Policy Change
- Enabling Audit Object Access on SAM file located in %WINDIR%\SYSTEM32\CONFIG folder
- Consider requiring smartcard logons for sensitive accounts

As an administrator, if you do nothing else after reading this chapter, start to lead by example. Use longer and more complex passwords. Make sure your personal passwords are not simple passwords. Make sure administrative and service account passwords are longer and more complex than the average end-user password. Begin to live and practice what you have been learning in the computer security field. You can use your experiences and lessons as a way to teach others.

After a successful authentication, a security principal has access to the resources that are allowed them by their security permissions and rights. Chapter 4 will cover NTFS permissions. If it's appropriately configured, Windows can significantly reduce the risk of malicious exploitation.