Chapter 7

Passwords

In This Chapter

- Identifying password vulnerabilities
- Examining password-hacking tools and techniques
- Hacking operating-system passwords
- Hacking password-protected files
- Protecting your systems from password hacking

Password hacking is one of the easiest and most common ways hackers obtain unauthorized computer or network access. Although strong passwords that are difficult to *crack* (or guess) are easy to create and maintain, users often neglect this. Therefore, passwords are one of the weakest links in the information-security chain. Passwords rely on secrecy. After a password is compromised, its original owner isn't the only person who can access the system with it. That's when bad things start happening.

Hackers have many ways to obtain passwords. They can glean passwords simply by asking for them or by looking over the shoulders of users as they type them in. Hackers can also obtain passwords from local computers by using password-cracking software. To obtain passwords from across a network, hackers can use remote cracking utilities or network analyzers.

This chapter demonstrates just how easily hackers can gather password information from your network. I outline common password vulnerabilities that exist in computer networks and describe countermeasures to help prevent these vulnerabilities from being exploited on your systems.

If you perform the tests and implement the countermeasures outlined in this chapter, you're well on your way to securing your systems' passwords.

Password Vulnerabilities

When you balance the cost of security and the value of the protected information, the combination of *user ID* and *secret password* is usually adequate.

However, passwords give a false sense of security. The bad guys know this and attempt to crack passwords as a step toward breaking into computer systems.

One big problem with relying solely on passwords for information security is that more than one person can know them. Sometimes, this is intentional; often, it's not. You can't know who has a password other than the owner.

Knowing a password doesn't make someone an authorized user.

Here are the two general classifications of password vulnerabilities:

- **Organizational or end-user vulnerabilities:** This includes lack of password awareness on the part of end users and the lack of password policies that are enforced within the organization.
- **Technical vulnerabilities:** This includes weak encryption methods and insecure storage of passwords on computer systems.

Before computer networks and the Internet, the user's physical environment was an additional layer of password security. Now that most computers have network connectivity, that protection is gone.

Organizational password vulnerabilities

It's human nature to want convenience. This makes passwords one of the easiest barriers for an attacker to overcome. Almost 3 trillion (yes, trillion with a *t* and 12 zeros) eight-character password combinations are possible by using the 26 letters of the alphabet and the numerals 0 through 9. However, most people prefer to create passwords that are easy to remember. Users like to use such passwords as "password," their login name, or a pet's name.

Unless users are educated and reminded about using strong passwords, their passwords usually are

- Weak and easy to guess.
- Seldom changed.
- Reused for many security points. When bad guys crack a password, they try to access other systems with the same password and user name.
- Written down in nonsecure places. The more complex a password is, the more difficult it is to crack. However, when users create more complex passwords, they're more likely to write them down. Hackers can find these passwords and use them against you.

A case study in Windows password vulnerabilities with Philippe Oechslin

In this case study, Dr. Philippe Oechslin, a researcher and independent information security consultant, shared with me his recent research findings on Windows password vulnerabilities.

The Situation

In 2003, Dr. Oechslin discovered a new method for cracking Windows passwords. While testing a brute-force password-cracking tool, he thought it was a waste of time for everyone using the same tool to have to generate the same hashes over and over again. He believed that generating a huge dictionary of all possible hashes would make it easier to crack Windows passwords, but then he quickly realized that a dictionary of the LAN Manager (LM) hashes of all possible alphanumerical passwords would require over a terabyte of storage.

During his research, Dr. Oechslin discovered a technique called time-memory trade-offs, where hashes are computed in advance but only a small fraction are stored (approximately one in a thousand). He discovered that how the LM hashes are organized allows you to find any password if you spend some time recalculating some of the hashes. This technique saves memory but takes a lot of time. Studying this method, he found a way to make it more efficient, making it possible to find any of the 80 billion unique hashes by using a table of 250 million entries (1GB worth of data) and performing only 4 million hash calculations. This process is much faster than a brute-force attack, which must generate 50 percent of the hashes (40 billion) on average.

This research is based on the absence of a random element when Windows passwords are hashed. This is true for both the LM hash and the NT hash built into Windows. As a result, the same password produces the same hash on any Windows machine. Although it is known that Windows hashes have no random element, no one has used a technique like the one that Dr. Oechslin discovered to crack Windows passwords.

For a short time, Dr. Oechslin and his team had an interactive tool on their Web site (`lasecwww.epfl.ch`) that enabled visitors to submit hashes and have them cracked. Over a six-day period, the tool cracked 1,845 passwords in an average of 7.7 seconds! They deactivated the demo after a week (and a million hits) and did not release the tool because they didn't want to help hackers. Dr. Oechslin did say that he has heard about other tools (such as RainbowCrack) that use the same method but are being developed independently.

The Outcome

So what's the big deal, you say? This password-cracking method can crack any alphanumerical password in a few seconds, whereas current brute-force tools can take several hours. Dr. Oechslin and his research team have generated a table with which they can crack any password made of letters, numbers, and 16 other characters in less than a minute, demonstrating that passwords made up of letters and numbers aren't good enough. He also stated that this method is useful for ethical hackers who have only limited time to perform their testing. Unfortunately, hackers have the same benefit and can perform their attacks before anyone detects them!

Philippe Oechslin, PhD, CISSP, is a lecturer and senior research assistant at the Swiss Federal Institute of Technology in Lausanne and spends his spare time as an independent information-security consultant.

Technical password vulnerabilities

You can often find these serious technical vulnerabilities after exploiting organizational password vulnerabilities:

- Weak password-encryption schemes. Hackers can break weak password storage mechanisms by using cracking methods that I outline in this chapter. Many vendors and developers believe that passwords are safe from hackers if they don't publish the source code for their encryption algorithms. *Wrong!* A persistent, patient hacker can usually crack this *security by obscurity* fairly quickly. After the code is cracked, it is soon distributed across the Internet and becomes public knowledge.

  Password-cracking utilities take advantage of weak password encryption. These utilities do the grunt work and can crack any password, given enough time and computing power.
- Software that stores passwords in memory and easily accessed databases.
- End-user applications that display passwords on the screen while typing.

The ICAT Metabase (an index of computer vulnerabilities) currently identifies over 460 technical password vulnerabilities, 230 of which are labeled as high-severity. You can search for some of these issues at `icat.nist.gov/icat.cfm` to find out how vulnerable some of your systems are from a technical perspective.

Cracking Passwords

Password cracking is one of the most enjoyable hacks for the bad guys. It fuels their sense of exploration and desire to figure things out. You may not have a burning desire to explore everyone's passwords, but it helps to approach password cracking with this thinking. So where should you start hacking the passwords on your systems? Generally speaking, any user's password works. After you obtain one password, you can obtain others — including administrator or root passwords.

Administrator passwords are the pot of gold. With unauthorized administrative access, you can do virtually anything on the system. When looking for your organization's password vulnerabilities, I recommend first trying to obtain the highest level of access possible (such as administrator) through the most discreet method possible. That's what the hackers do.

You can use low-tech ways and high-tech ways to exploit the vulnerabilities and obtain passwords. For example, you can deceive users into divulging passwords over the telephone or simply observe what a user has written down on a piece of paper. Or you can capture passwords directly from a computer or over a network or the Internet with tools covered in the following sections.

Cracking passwords the old-fashioned way

A hacker can use low-tech methods to crack passwords. These methods include using social-engineering techniques, shoulder surfing, and simply guessing passwords from information that you know about the user.

Social engineering

The most popular low-tech method is *social engineering,* which is covered in detail in Chapter 5. Social engineering takes advantage of the trusting nature of human beings to gain information that can later be used maliciously.

Techniques

To obtain a password through social engineering, you just ask for it. For example, you can simply call a user and tell him that he has some important-looking e-mails stuck in the mail queue and you need his password to log in and free them up. This is how hackers try to get the information!

If your colleague gives you his password, make sure that he changes it.

Countermeasures

User awareness is the best defense against social engineering. Train users to spot attacks (such as suspicious phone calls or deceitful e-mails) and respond effectively. Their best response is to not give out any information and to alert the appropriate information-security officer in the organization to see whether the inquiry is legitimate and whether a response is necessary. For this defense to be successful, the organization must enforce a security policy and provide ongoing security-awareness training to users.

Shoulder surfing

Shoulder surfing is an effective, low-tech password hack.

Techniques

To mount this attack, you must be near the user and not look obvious. Simply watch either the user's keyboard or screen when logging in.

A hacker with a good eye may watch whether the user is glancing around his desk for either a reminder of the password or the password itself.

Many folks have experienced shoulder surfing at the grocery-store checkout line. You swipe your debit card to pay for your chips and dip; you enter your PIN to authorize the transaction; and before you know it, the guy in line behind you has your PIN! He simply watched you enter it into the keypad.

You can try shoulder surfing yourself — though preferably not in the grocery-store checkout line. Just walk around the office and perform random spot checks. Go to users' desks, and ask them to log in to their computers, the

network, or even their e-mail applications. Just don't tell them what you're doing beforehand, or they'll be on to you and attempt to hide what they're typing or where they're looking for their password — two things that they should've been doing all along!

Countermeasures

Encourage users to be aware of their surroundings and not enter their passwords when they suspect that someone is looking over their shoulder. Instruct users that if they suspect someone is looking over their shoulder while they're logging in, they should politely ask the person to look away.

Inference

Inference is simply guessing passwords from information you know about users — such as their date of birth, favorite television show, and phone numbers. It sounds silly, but you can determine passwords by guessing!

The best defense against an inference hack attack is to educate users about creating secure passwords that do not include information that can be associated with them. You can't easily enforce this practice with technical controls, so you need a sound security policy and ongoing awareness training to remind users of the importance of secure password creation.

Weak authentication

Hackers can obtain — or simply avoid having to use — passwords by taking advantage of older operating systems, such as Windows 9*x* and Me. These operating systems don't require passwords to log in.

Bypassing authentication

On a Windows 9*x* or similar workstation that's prompting for a password, you can press Esc on the keyboard to get right in. After you're in, you can find other passwords stored in such places as dial-up networking connections and screen savers. These weak systems can serve as *trusted* machines — meaning that it's assumed that they're secure — and provide good launching pads for network-based password attacks as well.

Countermeasures

The only true defense against this hack is to not use operating systems that employ weak authentication. To eliminate this vulnerability, upgrade to Windows XP, or use Linux or the flavors of UNIX, including Mac OS X.

More modern authentication systems (such as Kerberos, which is used in newer versions of Windows), directory services (such as Novell's eDirectory), and network-based e-mail systems (such as Exchange) encrypt user passwords or don't communicate the passwords across the network. These measures create an extra layer of security, but these authentication systems still have some vulnerabilities, which I discuss shortly.

High-tech password cracking

High-tech password cracking involves using a program that tries to guess a password by determining all possible password combinations. These high-tech methods are mostly automated after you access the computer and password database files.

Password cracking software

You can try to crack your organization's operating-system and Internet-application passwords with various password cracking tools:

- LC4 (previously called L0phtcrack) can sniff out password hashes from the wire. Go to `www.atstake.com/research/lc`
- NetBIOS Auditing Tool (NAT) specializes in network-based password attacks. Go to `www.securityfocus.com/tools/543`
- Chknull (`www.phreak.org/archives/exploits/novell`) for Novell NetWare password testing
- These tools require physical access on the tested computer:
  - John the Ripper (`www.openwall.com/john`)
  - pwdump2 (`razor.bindview.com/tools/desc/pwdump2_readme.html`)
  - Crack (`coast.cs.purdue.edu/pub/tools/unix/pwdutils/crack`)
  - Brutus (`www.hoobie.net/brutus`)
  - Pandora (`www.nmrc.org/project/pandora`)
  - NTFSDOS Professional (`www.winternals.com`)
- Various other handy password tools exist, such as
  - GetPass for decrypting login passwords for Cisco routers (`www.boson.com/promo/utilities/getpass/getpass_utility.htm`)
  - Win Sniffer for capturing FTP, e-mail, and other types of passwords off the network
  - Cain and Abel for capturing, cracking, and even calculating various types of passwords on a plethora of systems (`www.oxid.it/cain.html`)

You may be wondering what value a password-cracking tool offers if you need physical access to your systems to test them. Some would say that if a hacker can obtain physical access to your systems and password files, you have more than just basic information-security problems to worry about. But this kind of access is entirely possible! What about a summer intern, a disgruntled employee, or an outside consultant with malicious intent?

Password-cracking utilities take a set of known passwords and run them through a password-hashing algorithm. The resulting hashes — or an encrypted form of a data set — are then compared at lightning speed to the password hashes extracted from the original password database. When a match is found between the newly generated hash and the hash in the original database, the password has been cracked. It's that simple.

Other password-cracking programs simply attempt to logon using a predefined set of user IDs and passwords. In fact, NAT can do just that. NAT takes advantage of some known weaknesses in Microsoft's Server Message Block (SMB) protocol, which is used for file and print sharing.

Try running NAT in a real-world scenario. Simply download NAT from the preceding address, and extract it to a temporary directory on your hard drive. NAT comes with some predefined usernames and passwords in the `userlist.txt` and `passlist.txt` files, but you can modify them or add your own. For a quick test of a Windows NT or 2000 machine across the network, enter this basic NAT command at a command prompt:

```
nat -u userlist.txt -p passlist.txt IP_address_of_the_computer_you're_testing
```

Figure 7-1 shows the output of my test server when I ran NAT against it. NAT used the default password list to crack the administrator password in just a few seconds. If you don't have any luck, consider using one of the dictionary files listed in the next section. Just give the test some time. If you use one of the larger lists, the process may take quite a while.

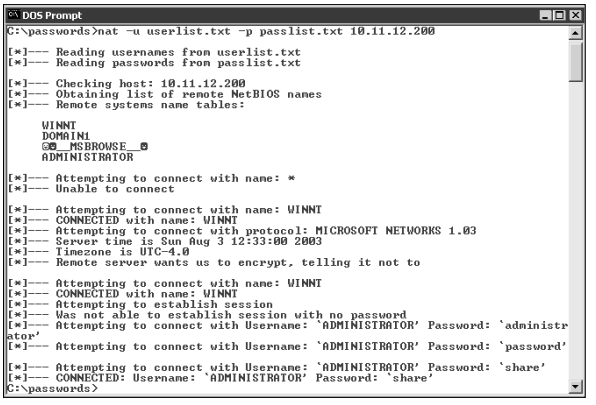

Figure 7-1: Output from the NetBIOS Auditing Tool.

Passwords that are subjected to cracking tools eventually lose. You have access to the same tools as the bad guys. These tools can be used for both legitimate auditing and malicious attacks. You want to audit your passwords before the bad guys do, and in this section, I show you some of my favorite methods for auditing Windows and Linux/UNIX passwords.

When trying to crack passwords, the associated user accounts may be locked out, which could interrupt your users. Be careful if you have intruder lockout enabled — you may have to go back in and reenable locked accounts.

Passwords are typically stored on a computer in an encrypted fashion, using an encryption or one-way hash algorithm such as DES or MD5. Hashed passwords are then represented as fixed-length encrypted strings that always represent the same passwords with exactly the same strings. These hashes are irreversible for all practical purposes, so passwords can never be decrypted.

Password storage locations vary by operating system:

- Windows usually stores passwords in these locations:
  - Security Accounts Manager (SAM) database (`c:\winnt\system32\config`)
  - Active Directory database file that's stored locally or spread across domain controllers (`ntds.dit`)

  Windows sometimes stores passwords in either a backup of the SAM file in the `c:\winnt\repair` directory or on an emergency repair disk.

  Some Windows applications store passwords in the Registry or as plaintext files on the hard drive!
- Linux and other UNIX variants typically store passwords in these files:
  - `/etc/passwd` (readable by everyone)
  - `/etc/shadow` (accessible by root only)
  - `/etc/security/passwd` (accessible by root only)
  - `/.secure/etc/passwd` (accessible by root only)

Two high-tech password-cracking methods are dictionary attacks and brute-force attacks.

Dictionary attacks

Dictionary attacks against passwords quickly compare a set of words — including many common passwords — against a password database. This database is a text file with thousands of words typically listed in alphabetical order. For instance, suppose that you have a dictionary file that you downloaded from one of the sites in the following list. The English dictionary file at the Purdue site contains one word per line starting with *10th, 1st* . . . all the way to *zucchini* and *zygote*.

Many password-cracking utilities can use a separate dictionary that you create or download from the Internet. Here are some popular sites that house dictionary files and other miscellaneous word lists:

- ftp://ftp.cerias.purdue.edu/pub/dict
- ftp://ftp.ox.ac.uk/pub/wordlists
- packetstormsecurity.nl/Crackers/wordlists
- www.outpost9.com/files/WordLists.html

Most dictionary attacks are good for *weak* (easily guessed) passwords. However, some special dictionaries have common misspellings of words such as pa$$w0rd (password) and 5ecur1ty (security), non-English words, and thematic words from religions, politics, or *Star Trek*.

Brute-force attacks

Brute-force attacks can crack any password, given sufficient time. Brute-force attacks try every combination of numbers, letters, and special characters until the password is discovered. Many password-cracking utilities let you specify such testing criteria as the characters and password length to try.

A brute-force test can take quite a while, depending on the number of accounts, their associated password complexities, and the speed of the computer that's running the cracking software.

Smart hackers attempt logins slowly or at random times so the failed login attempts aren't as predictable or obvious in the system log files. Some malicious users may even call the IT help desk to attempt a reset of the account they've just locked out. This social-engineering technique could be a major issue, especially if the organization has no or minimal mechanisms in place to verify that locked-out users are who they say they are.

Can an expiring password deter a hacker's attack and render password-cracking software useless? Yes. After the password is changed, the cracking must start again if the hacker wants to test all the possible combinations. This is one reason why passwords must be changed periodically. Shortening the change interval can reduce the risk of a password's being cracked.

Exhaustive password-cracking attempts usually aren't necessary. Most passwords are fairly weak. Even minimum password requirements, such as a password length, can help you in your testing; you may be able to give your cracking programs more defined cracking parameters, which eliminates combinations for faster results.

Cracking passwords with pwdump2 and John the Ripper

The following steps use two of my favorite utilities to test the security of current passwords on Windows systems:

- pwdump2 (to extract password hashes from the Windows SAM database)
- John the Ripper (to crack the hashes of Windows and UNIX passwords)

This test requires administrative access to either your Windows NT/2000 stand-alone workstation or server:

1. **Create a new directory called `passwords` from the root of your Windows C: drive.**
2. **Download and install a decompression tool, if you don't have one.**

   

   FreeZip (`members.ozemail.com.au/~nulifetv/freezip`) and IZArc (`www.webattack.com/get/izarc.shtml`) are free Windows decompression tools. Windows XP includes built-in decompression.
3. **Download, extract, and install the following software, if you don't already have it on your system:**
   - pwdump2 — download the file from `razor.bindview.com/tools/desc/pwdump2_readme.html`
   - John the Ripper — download the file from `www.openwall.com/john`
4. **Enter the following command to run pwdump2 and redirect its output to a file called `cracked.txt`:**

   ```
   pwdump2 > cracked.txt
   ```

   This file will be used to store the Windows SAM password hashes that will later be cracked with John the Ripper. Figure 7-2 shows the contents of the `cracked.txt` file that contains the local Windows SAM-database password hashes.

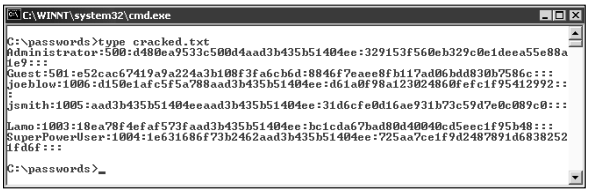

Figure 7-2: Output from pwdump2.

5. **Enter the following command to review the contents from the resulting hashes:**

   ```
   type cracked.txt
   ```

   All the users on your system are listed (similar to Figure 7-3), whether you run this on a stand-alone Windows NT/2000 system or Windows Primary Domain Controller (PDC).

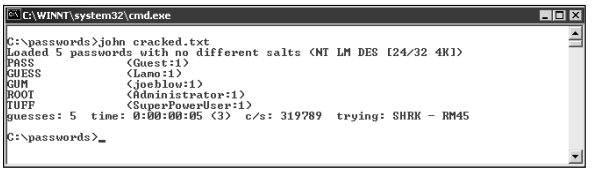

Figure 7-3: Cracked password file hashes from pwdump2.

6. **Enter the following command to run John the Ripper against the Windows SAM password hashes to display the cracked passwords:**

   ```
   john cracked.txt
   ```

   You should see something similar to the following:

   ```
   Loaded 3 passwords with no different salts (NT LM DES [24/32 4K])
   123         (Weak:1)
   PASS        (Newuser:1)
   GUESS       (Lame:1)
   guesses: 3 time: 0:00:00:00 (3) c/s: 165146 trying: SAMELL - SANDIT
   ```

   This process can take seconds or days, depending on the number of users and the complexity of their associated passwords. My Windows example took only five seconds to crack five weak passwords.

John the Ripper can crack UNIX passwords. You need root access to your system and to the password (`/etc/passwd`) and shadow password (`/etc/shadow`) files. Perform the following steps for cracking UNIX passwords:

1. **Download the UNIX source files from `www.openwall.com/john`.**
2. **Extract the program by entering the following command:**

   ```
   tar -zxf john-1.6.tar.gz
   ```

3. **Change into the `/src` directory that was created when you extracted the program, and enter the following command:**

   ```
   make generic.
   ```

4. **Change into the `/run` directory, and enter the following command to use the unshadow program to combine the passwd and shadow files and copy them to the file `cracked.txt`:**

   ```
   ./unshadow /etc/passwd /etc/shadow > cracked.txt
   ```

5. **Enter the following command to start the cracking process:**

   ```
   ./john cracked.txt
   ```

   When John the Ripper is complete (and this could take some time), you get an output similar to the results of the preceding Windows process.

After completing the preceding Windows or UNIX steps, you can either

- Force users to change passwords that don't meet specific password policy requirements.
- Create a password policy from scratch.

Be careful handling the results of your password cracking. Password information for others is confidential and should be treated with care.

Checking for null passwords in NetWare

Using the chknull program, you can test for NetWare users that have empty passwords, passwords that match their username, or passwords that match a specific password that you supply on the command line. Figure 7-4 shows the output of a chknull session against a NetWare server without being logged in: Four users have blank passwords, three users have the password "123," and one user's password is the same as his username (avadminuser).

Figure 7-4: NetWare password weaknesses found with chknull.

General password-hacking countermeasures

A password for one system usually equals passwords for many other systems, because many people use the same passwords on every system they use. For this reason, instruct users to create different passwords for different systems, especially on the systems that protect more sensitive information.

Strong passwords are important, but balance security and convenience:

- You can't expect users to memorize passwords that are insanely complex and changed every week.
- You can't afford weak passwords or no passwords at all.

Passwords by the numbers

One hundred twenty-eight different ASCII characters are used in typical computer passwords. (Technically, only 126 characters are used, because you can't use the NULL and the carriage return characters.) A truly random eight-character password that uses 126 different characters can have 63,527,879,748,485,376 different combinations. Taking that a step further, if it were possible (and it is, in Linux and UNIX) to use all 256 ASCII characters (254, without NULL and carriage return) in a password, 17,324,859,965,700,833,536 different combinations are possible. This is approximately 2.7 billion times more combinations than there are people on earth!

A text file containing all these possible passwords would require millions of terabytes of storage space. Even if you included just the more realistic combination of 95 or so ASCII letters, numbers, and standard punctuation characters, such a file would still fill thousands of terabytes of storage space. These storage requirements require password-cracking programs to form the password combinations on the fly, instead of reading all possible combinations from a text file. That's why brute-force attacks are more effective at cracking passwords than dictionary attacks.

Given the effectiveness of brute-force password attacks, it's not unrealistic to think that in the future, anyone will be able to crack all possible password combinations, given the current technology and average lifespan. It probably won't happen, but many of us also thought in the mid-1980s that 640KB of RAM and 10MB hard drives in our PCs were all we needed.

Storing passwords

If you have to choose between weak passwords that your users can memorize and strong passwords that your users must write down, I recommend having readers write down passwords and store the information securely. Train users to store their written passwords in a secure place — not on keyboards or in easily cracked password-protected computer files (such as spreadsheets). Users should store a written password in either of these locations:

- A locked file cabinet or office safe
- An encrypted file or database, using such tools as
  - PGP (`www.pgpi.org` for the free open-source version or `www.pgp.com` for the commercial version)
  - Open-source Password Safe, originally developed by Counterpane (`passwordsafe.sourceforge.net`)

No sticky notes!

Policy considerations

As an ethical hacker, you should show users the importance of securing their passwords. Here are some tips on how to do that:

- Demonstrate how to create secure passwords. You may want to refer to them as pass codes or pass phrases, because people tend to take the word *passwords* literally and use only words, which can be less secure.
- Show what can happen when weak passwords are used or passwords are shared.
- Diligently build user awareness of social-engineering attacks.

Enforce (or encourage the use of) a strong password-creation policy that includes the following criteria:

- Use upper- and lowercase letters, special characters, and numbers. (Never use only numbers. These passwords can be cracked quickly.)
- Misspell words or create acronyms from a quote or a sentence. (An *acronym* is a word created from the initials of a phrase. For example, *ASCII* is an acronym for *American Standard Code for Information Interchange.)*
- Use punctuation characters to separate words or acronyms.
- Change passwords every 6 to 12 months.
- Use different passwords for each system. This is especially important for network-infrastructure hosts, such as servers, firewalls, and routers.
- Use variable-length passwords. This can throw off the hackers, because they won't know the required minimum or maximum length of passwords and must try all password length combinations.
- Don't use common slang words or words that are in a dictionary.
- Don't use similar-looking characters, such as *3* instead of *E, 5* instead of *S,* or *!* instead of *1.* Password-cracking programs can check for this.
- Don't reuse the same password within 12 months.
- Use password-protected screen savers.
- Don't share passwords.
- Avoid storing user passwords in a central place, such as an unsecured spreadsheet on a hard drive. This is an invitation for disaster. Use PGP, Password Safe, or a similar program to store user passwords.

Other considerations

Here are some other password-hacking countermeasures that I recommend:

- Enable security auditing to help monitor and track password attacks.
- Test your applications to make sure they aren't storing passwords in memory or writing them to disk.

Some password-cracking Trojan-horse applications are transmitted through worms or simple e-mail attachments, such as `VBS.Network.B` and `PWSteal.SoapSpy`. These applications can be lethal to your password-protection mechanisms if they're installed on your systems. The best defense is malware protection software, such as antivirus protection (from a vendor like Norton or McAfee), spyware protection (such as PestPatrol or Spybot), or malicious-code behavioral protection (such as Finjan's offerings).

- Keep your systems patched. Passwords are reset or compromised during buffer overflows or other DoS conditions.
- Know your user IDs. If an account has never been used, delete or disable the account until it's needed. You can determine unused accounts by manual inspection or by using a tool such as DumpSec (`www.somarsoft.com`), which can enumerate the Windows operating system and gather user ID and other information.

As the security administrator in your organization, you can enable *account lockout* to prevent password-cracking attempts. Most operating systems and some applications have this capability. Don't set it too low (less than five failed logins), and don't set it too high to give a malicious user a greater chance of breaking in. Somewhere between 5 and 50 may work for you. I usually recommend a setting of around 10 or 15.

- To use account lockout and prevent any possibilities of a user DoS condition, require two different passwords, and don't set a lockout time for the first one.
- If you permit auto reset of the account after a certain time period — often referred to as *intruder lockout* — don't set a short time period. Thirty minutes often works well.

A failed login counter can increase password security and minimize the overall effects if the account is being compromised by an automated attack. It can force a password change after a number of failed attempts. If the number of failed login attempts is high, and they all occurred in a short period of time, the account has likely experienced an automated password attack.

Some more password-protection countermeasures include the following:

- Use stronger authentication methods, such as challenge/response, smart cards, tokes, biometrics, or digital certificates.
- Automate password reset. This functionality lets users to manage most of their password problems without getting others involved. Otherwise, this support issue becomes expensive, especially for larger organizations.
- Password-protect the system BIOS (basic input/output system). This is especially important on servers and laptops that are susceptible to physical-security threats and vulnerabilities.

Password-protected files

Do you wonder how vulnerable word-processing, spreadsheet, and zip files are as users send them into the wild blue yonder? Wonder no more. Some great utilities can show how easily passwords are cracked.

Cracking files

Most password-protected files can be cracked in seconds or minutes. You can demonstrate this "wow-factor" security vulnerability to users and management. Here's a real-world scenario:

- Your CFO wants to send some confidential financial information in an Excel spreadsheet to the company's outside financial advisor.
- She protects the spreadsheet by assigning a password to it during the file-save process in Excel 2002.
- For good measure, she uses WinZip to compress the file, and adds another password to make it *really* secure.
- The CFO sends the spreadsheet as an e-mail attachment, assuming that it will reach its destination securely.

  The financial advisor's network has content filtering, which monitors incoming e-mails for keywords and file attachments. Unfortunately, the financial advisory firm's network administrator is looking in the content-filtering system to see what's coming in.
- This rogue network administrator finds the e-mail with the confidential attachment, saves the attachment, and realizes that it's password-protected.
- The network administrator remembers some great password-cracking utilities from ElcomSoft (`www.elcomsoft.com`) that can help him out. He may see something like Figures 7-5 and 7-6.

Cracking password-protected files is as simple as that! Now all that the rogue network administrator must do is forward the confidential spreadsheet to his buddies or the company's competitors.

If you carefully select the right options in Advanced ZIP Password Recovery and Office XP Password Recovery, you can drastically shorten your testing time. For example, if you know that a password is not over 5 characters or is lowercase letters only, you can cut the cracking time in half.

I recommend performing these file password-cracking tests on files that you capture with a content-filtering or network-analysis tool.

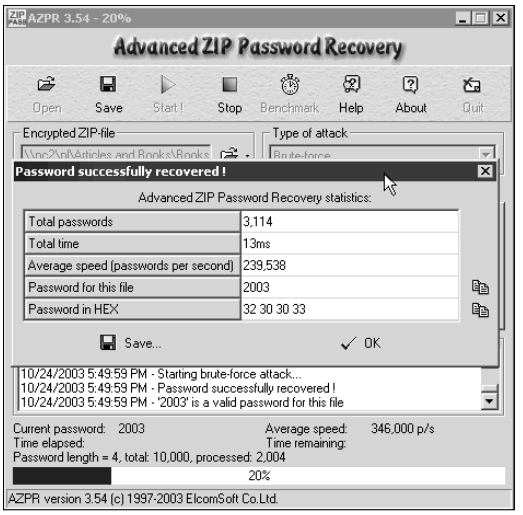

Figure 7-5: ElcomSoft's Advanced ZIP Password Recovery cracking a zip file.

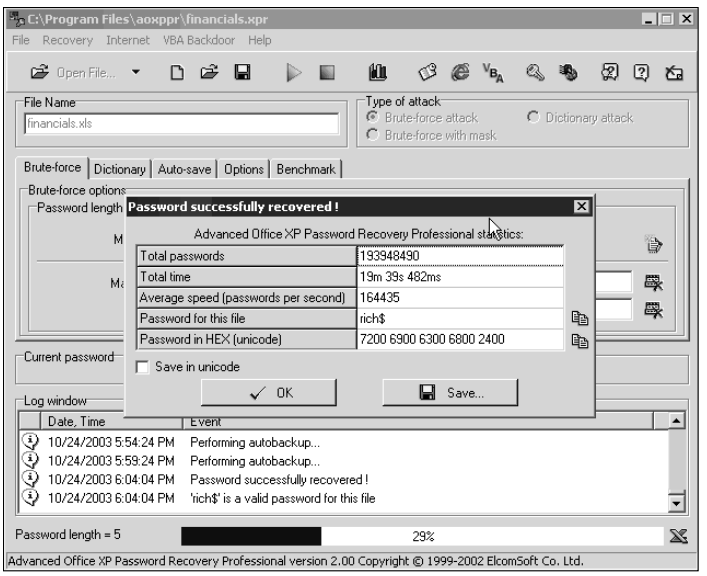

Figure 7-6: ElcomSoft's Advanced Office XP Password Recovery cracking a spreadsheet.

Countermeasures

The best defense against weak file password protection is to require your users to use a stronger form of file protection, such as PGP, when necessary. Ideally, you don't want to rely on users to make decisions about what they should use this method to secure, but it's better than nothing. Stress that a file-encryption mechanism such as PGP is secure only if users keep their passwords confidential and never transmit or store them in clear text.

If you're concerned about nonsecure transmissions through e-mail, consider one of these options:

- Block all outbound e-mail attachments that aren't protected on your e-mail server.
- Use an encryption program, such as PGP, to create self-extracting encrypted files.
- Use content-filtering applications.

Other ways to crack passwords

Over the years, I've found other ways to crack passwords, both technically and through social engineering.

Keystroke logging

One of the best techniques for cracking passwords is remote *keystroke logging* — the use of software or hardware to record keystrokes as they're being typed into the computer.

Be careful with keystroke logging. Even with good intentions, monitoring employees can raise some legal issues. Discuss what you'll be doing with your legal counsel, and get approval from upper management.

Logging tools

With keystroke-logging tools, you can later assess the log files of your application to see what passwords people are using:

- Keystroke-logging applications can be installed on the monitored computer. I recommend that you check out eBlaster and Spector Pro by SpectorSoft (`www.spectorsoft.com`). Another popular tool that you can use is Invisible KeyLogger Stealth, at `www.amecisco.com/iks.htm`, as well as the hardware-based KeyGhost (`www.keyghost.com`). Dozens of other such tools are available on the Internet.
- Hardware-based tools fit between the keyboard and the computer or replace the keyboard altogether.

A shared computer can capture the passwords of every user who logs in.

Countermeasures

The best defense against the installation of keystroke-logging software on your systems is a spyware-detection program or popular antivirus products.

The potential for hackers to install keystroke-logging software is another reason to ensure that your users aren't downloading and installing random shareware or opening attachments in unsolicited e-mails. Consider locking down your desktops by setting the appropriate user rights through local or group security policy in Windows. Alternatively, you could use a commercial lock-down program, such as Fortres 101 (`www.fortres.com`) for Windows or Deep Freeze (`www.deepfreezeusa.com`) for Windows and Mac OS X.

Weak password storage

Many legacy and stand-alone applications such as e-mail, dial-up network connections, and accounting software store passwords locally, making them vulnerable to password hacking. By performing a basic text search, I've found passwords stored in clear text on the local hard drives of machines.

Searching

You can try using your favorite text-searching utility — such as the Windows search function, `findstr`, or `grep` — to search for *password* or *passwd* on your drives. You may be shocked to find what's on your systems. Some programs even write passwords to disk or leave them stored in memory.

This is a hacker's dream. Head it off if you can.

Countermeasures

The only reliable way to eliminate weak password storage is to use only applications that store passwords securely. This may not be practical, but it's your only guarantee that your passwords are secure.

Before upgrading applications, contact your software vendor or search for a third-party solution.

Network analyzer

A network analyzer sniffs the packets traversing the network. This is what the bad guys do if they can gain control over a computer or gain physical network access to set up their network analyzer. If they gain physical access, they can look for a network jack on the wall and plug right in!

Testing

Figure 7-7 shows how crystal-clear passwords can be through the eyes of a network analyzer. This figure shows the password packet from an EtherPeek capture of a POP3 session using Microsoft Outlook to download messages from an e-mail server. Look in the POP — Post Office Protocol section for the password of "MyPassword". These same clear-text password vulnerabilities can apply to instant messaging, Web-site logins, telnet sessions, and more. Basically, if traffic is not being tunneled through a VPN, SSH, SSL, or some other form of encrypted link, it's vulnerable to attack.

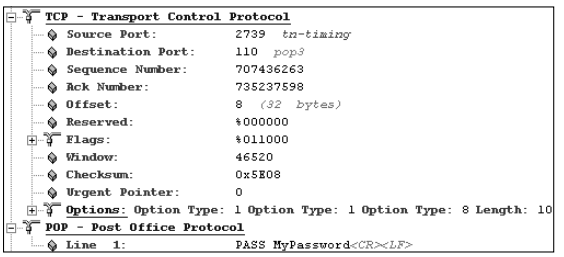

Figure 7-7: An EtherPeek capture of a POP3 password packet.

Although you can benefit from using a commercial network analyzer such as EtherPeek, you don't need to buy one for your testing. An open-source program, Ethereal, runs on Windows and UNIX platforms. You can search for password traffic on the network a million ways. For example, to capture POP3 password traffic, set up a trigger to search for the PASS command. When the network analyzer sees the PASS command in the packet, it starts capturing data until your specified time or number of packets.

Capture this data on a hub segment of your network, or plug your network-analyzer system into a monitor port on a switch. Otherwise, you can't see anyone else's data traversing the network — just yours. Check your switch's user's guide for whether it has a monitor or mirror port and instructions on how to configure it. You can connect your network analyzer to a hub on the public side of your firewall. You'll capture only those packets that are entering or leaving your network — not internal traffic.

Countermeasures

Here are some good defenses against network-analyzer attacks:

- Use switches on your network, not hubs.

  If you must use hubs on network segments, a program such as sniffdet, cpm, and sentinel can detect network cards in *promiscuous mode* (accepting all packets, whether destined for it or not). Network cards in this mode are signs of a network analyzer running on the network.

- Don't let a hacker gain physical access to your switches or the network connection on the public side of your firewall. With physical access, a hacker can connect to a switch monitor port, or tap into the unswitched network segment outside the firewall and capture packets.

Switches do not provide complete security because they are vulnerable to ARP poisoning attacks, which I cover in Chapter 9.

Most computer BIOSs allow power-on passwords and/or setup passwords to protect the computer's hardware settings that are stored in the CMOS chip. Here are some ways around these passwords:

- You can usually reset these passwords by either unplugging the CMOS battery or changing a jumper on the motherboard.
- Password-cracking utilities for BIOS passwords are available.

Some systems (especially laptops) can't be reset easily. You can lose all the hardware settings and lock yourself out of your own computer. If you plan to hack your own BIOS passwords, check for information in your user manual or on `labmice.techtarget.com/articles/BIOS_hack.htm` on doing this safely.

Weak passwords in limbo

Bad guys often exploit user accounts that have just been reset by a network administrator or help desk. Accounts may need to be reset if users forget their passwords, or if the accounts have been locked out because of failed attempts.

Weaknesses

Here are some reasons why user accounts can be vulnerable:

- When user accounts are reset, they often are assigned an easily cracked password (such as the user's name or the word *password*). The time between resetting the user account and changing the password is a prime opportunity for a break-in.
- Many systems have either default accounts or unused accounts with weak passwords or no passwords at all. These are prime targets.

Countermeasures

The best defenses against attacks on passwords in limbo are solid help-desk policies and procedures that prevent weak passwords from being available at *any* given time during the password-reset process. Perhaps the best ways to overcome this vulnerability are as follows:

- Require users to be on the phone with the help desk, or have a help-desk member perform the reset at the user's desk.
- Require that the user immediately log in and change his password.
- If you need the ultimate in security, implement stronger authentication methods, such as challenge/response, smart cards, or digital certificates.
- Automate password-reset functionality on your network so users can manage most of their password problems without help from others.

For a good list of default system passwords for vendor equipment, check `www.cirt.net/cgi-bin/passwd.pl`.

Password-reset programs

Network administrators occasionally use administrator password-resetting programs, which can be used against a network.

Tools

One of my favorites for Windows is NTAccess (www.mirider.com/ntaccess.html). This program isn't fancy, but it does the job.

Countermeasures

The best safeguard against a hacker using a password-reset program against your systems is to ensure the hacker can't gain physical access. When a hacker has physical access, all bets are off.

Securing Operating Systems

You can implement various operating-system security measures to ensure that passwords are protected.

Regularly perform these low-tech and high-tech password-cracking tests to make sure that your systems are as secure as possible — perhaps as part of a monthly, quarterly, or biannual audit.

Windows

The following countermeasures can help prevent password hacks on Windows systems:

- Some Windows passwords can be gleaned by simply reading the clear text or crackable cipher text from the Windows Registry. Secure your registries by doing the following:
  - Allowing only administrator access.
  - Hardening the operating system by using well-known hardening best practices, such as such as those from SANS (www.sans.org), NIST (csrc.nist.gov), the National Security Agency Security Recommendation Guides (www.nsa.gov/snac/index.html), and the ones outlined in *Network Security For Dummies,* by Chey Cobb (Wiley Publishing, Inc.).
- Use SYSKEY for enhanced Windows password protection.
  - By default, Windows 2000 encrypts the SAM database that stores hashes of the Windows account passwords. It's not the default in Windows NT.
  - You can use the SYSKEY utility to encrypt the database for Windows NT machines and to move the database-encryption key from Windows 2000 and later machines.

  Don't rely only on the SYSKEY utility. Tools such as ElcomSoft's Advanced EFS Data Recovery program can crack SYSKEY encryption.

- Keep all SAM-database backup copies secure.
- Disable the storage of LM hashes in Windows for passwords that are shorter than 15 characters.

  For example, in Windows 2000 SP2 and later, you can create and set the NoLMHash registry key to a value of 1 under `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa`.
- Use passfilt.dll or local or group security policies to help eliminate weak passwords on Windows systems before they're created.
- Disable null sessions in your Windows version:
  - In Windows XP, enable the Do Not Allow Anonymous Enumeration of SAM Accounts and Shares option in the local security policy.
  - In Windows 2000, enable the No Access without Explicit Anonymous Permissions option in the local security policy.
  - In Windows NT, enable the following Registry key:

```
HKLM/System/CurrentControlSet/Control/LSA/RestrictAnonymous=1
```

Linux and UNIX

The following countermeasures can help prevent password cracks on Linux and UNIX systems:

- Use shadowed MD5 passwords.
- Help prevent weak passwords from being created. You can use either built-in operating-system password filtering (such as cracklib in Linux) or a password auditing program (such as npasswd or passwd+).
- Check your `/etc/passwd` file for duplicate root UID entries. Hackers can exploit such entries as root backdoors.