Password Cracking

Passwords are typically cracked using one or more of the following methods:

Guessing:

Even with all of the advanced programs, algorithms, and techniques computer scientists have come up with, sometimes the most effective way of cracking a user password is by using logic and/or trying commonly used passwords. For many unknowledgeable users, passwords are more of an annoyance than a security, and therefore they use passwords that can be easily remembered, and thus easily guessed, such as:

The word "password"
The same as the user name
Name of the user
Birthdays or birth places
Relatives
Pets
Favorite colors, foods, places, etc.

This method can be more time efficient than using a program, but obviously this method works best when the person "recovering" the password knows the user. One other thing to keep in mind is that the average user does not like coming up with multiple passwords, so if you can figure out one password for one area, you can usually gain access to most other password protected areas using the same password.

Dictionary Attacks:

Dictionary Attacks are a method of using a program to try a list of words on the interface or program that is protecting the area that you want to gain access to. The most simple password crackers using dictionary attacks use a list of common single words, aka a "dictionary". More advanced programs often use a dictionary on top of mixing in numbers or common symbols at the beginning or end of the guessed words.
Some can even be given a set of personal information or a profile of the user and pick out important words to guess, even if they are not proper words, such as pronouns like last names and names of relatives.
A weakness of dictionary attacks is that it obviously relies on words supplied by a user, typically real words, to function. If the password is misspelled, is in another language, or very simply uses a word that is not in the dictionary or profile, it cannot succeed. Most of the time, even using two words in one password can thwart a dictionary attack.

Examples of programs that use dictionary attacks: John the Ripper, L0phtCrack, and Cain And Abel.

Brute Force:

Brute force password attacks are a last resort to cracking a password as they are the least efficient. In the most simple terms, brute force means to systematically try all the combinations for a password. This method is quite efficient for short passwords, but would start to become infeasible to try, even on modern hardware, with a password of 7 characters or larger. Assuming only alphabetical characters, all in capitals or all in lower-case, it would take 26^7 (8,031,810,176) guesses. This also assumes that the cracker knows the length of the password. Other factors include number, case-sensitivity, and other symbols on the keyboard. The complexity of the password depends upon the creativity of the user and the complexity of the program that is using the password.
The upside to the brute force attack is that it will ALWAYS find the password, no matter it's complexity. The downside is whether or not you will still be alive when it finally guesses it.

Examples of programs that use brute force attacks: John the Ripper, Rarcrack, and Oracle.

Rainbow Tables

Rainbow tables are a type of precomputed password attack. The previous two attacks, Dictionary and Brute-Force, enter a password into the locked program, the program then hashes the entry and compares the hash to the correct password hash. Rainbow tables compute hashes for each word in a dictionary, store all of the hashes into a hash table, retrieve the hash of the password to be cracked, and do a comparison between each password hash and the real password hash. This method assumes that you can retrieve the hash of the password to be guessed and that the hashing algorithm is the same between the rainbow table and the password. As the majority of common, low-security hashes are computed using MD5, sometimes SHA-1, this problem isn't very worrisome.
Rainbow tables have only become an efficient technique recently, as the hard drive space needed to store the hashes was slightly combersome until memory became cheaper. To give you an idea of how large a rainbow table can be:

Character Set	Length	Table Size
ABCDEFGHIJKLMNOPQRSTUVWXYZ	14	0.6 GB
ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789	14	3 GB
ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789!@#$%^&*()-_+=	14	24 GB
ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789!@#$%^&*()-_+=~`[]{}|\:;'"<>,.?/	14	64 GB

Figures taken from http://www.codinghorror.com/blog/archives/000949.html

Examples of programs that use rainbow tables: OphCrack, Oracle, and RainbowCrack

Benchmarking

To give an idea of the speed of modern password crackers, according to Red Data Security, on a system with a 3GHz Pentium 4 processor, Oracle in brute-force mode, takes:

- 10 seconds to calculate all 5 ascii character combinations (26^5)
- 5 minutes to calculate all 6 ascii character combinations (26^6)
- 2 hours to calculate all 7 ascii character combinations (26^7)
- 2,1 days to calculate all 8 ascii character combinations (26^8)
- 57 days to calculate all 9 ascii character combinations (26^9)
- 4 years to calculate all 10 ascii character combinations (26^{10})

Salt

One other element of passwords that is becoming more and more common is a technique called "salting." Salting a password means, more or less, adding bits of information (aka the "salt") to the given password before hashing it, so that the password is not merely guessable by a standard rainbow table, as the hashes are not of simple words anymore. The wikipedia article on salt gives a good example of this:

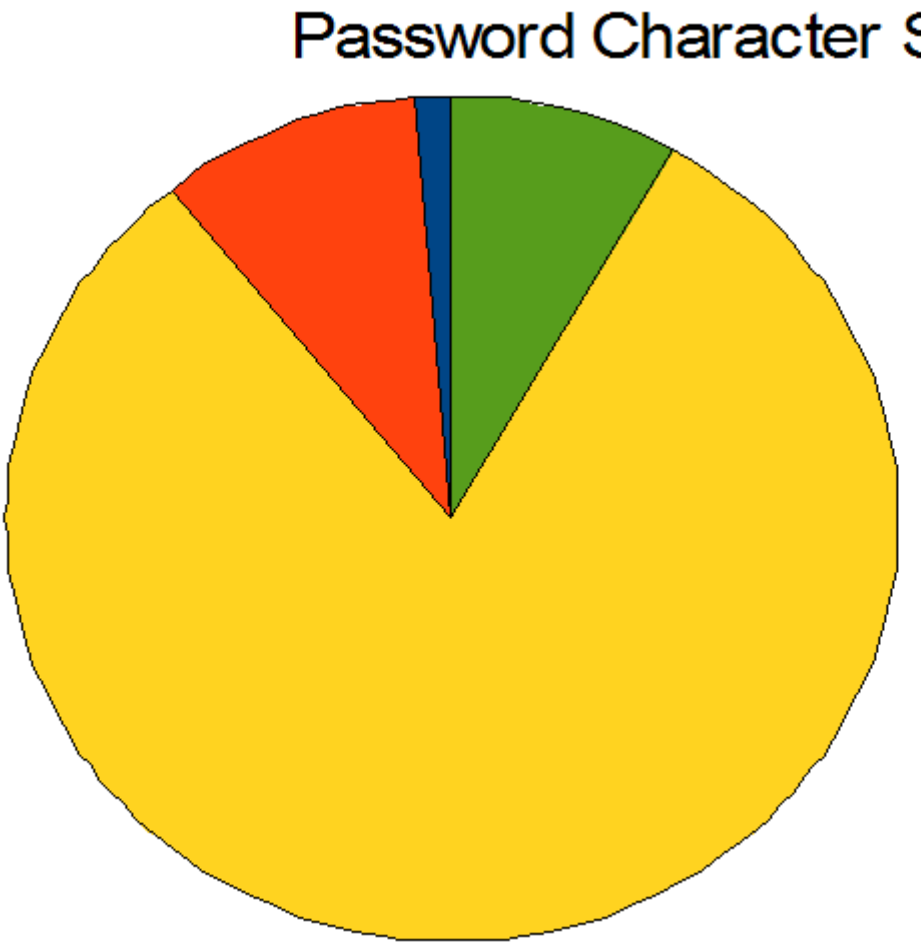
"Assume a user's (encrypted) secret key is stolen and he is known to use one of 200,000 English words as his password. The system uses a 32-bit salt. The salted key is now the original password appended to this random 32-bit salt. Because of this salt, the attacker's pre-calculated hashes are of no value. He must calculate the hash of each word with each of 232 (4,294,967,296) possible salts appended until a match is found. The total number of possible inputs can be obtained by multiplying the number of words in the dictionary with the number of possible salts:
$2^{32} * 200{,}000 = 8.58993459 * 10^{14}$

To complete a brute-force attack, the attacker must now compute about 800 trillion hashes, instead of only 200,000. Even though the password itself is known to be simple, the secret salt makes breaking the password radically more difficult."

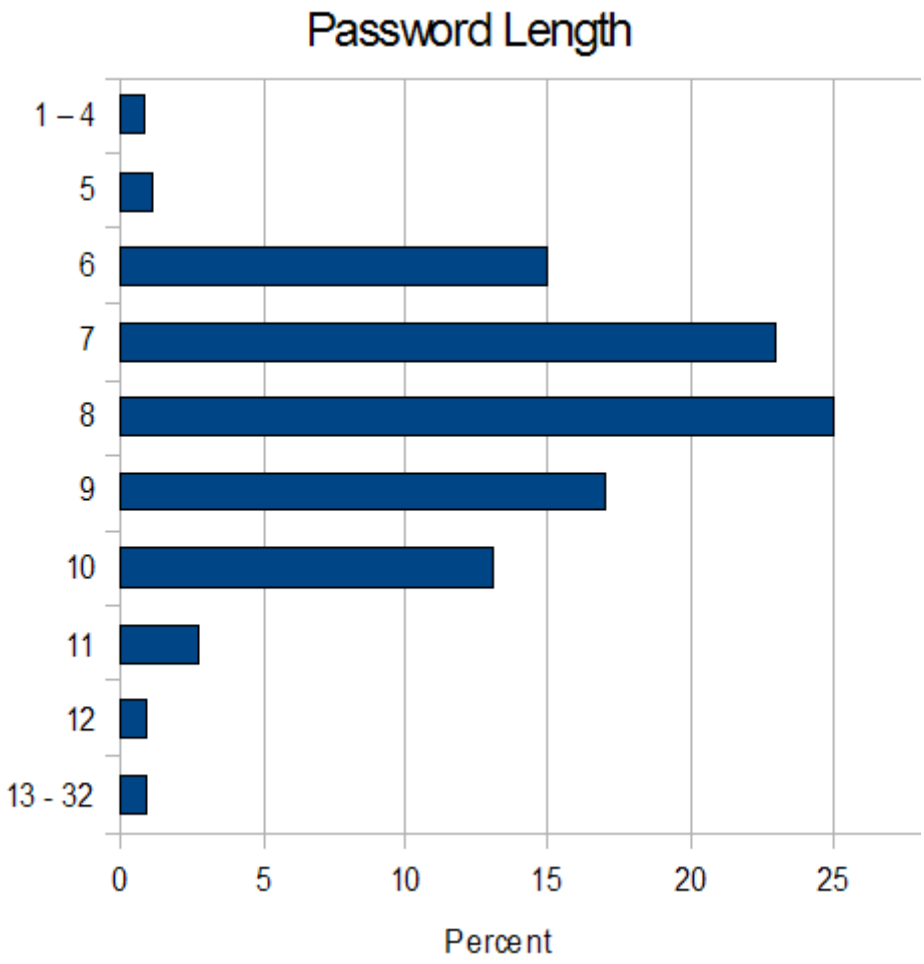
As you can see, salting makes cracking a password much more difficult, but it should be noted that this only complicates cracking programs that use hashes rather than rapid input. Normal dictionary and brute-force attacks are not affected by the salt.

Analysis of Passwords

In 2006, a massive password phishing scam was conducted on Myspace users. It is estimated that 34,000 Myspace users had their passwords and usernames stolen. An analysis of this attack can show us a glimpse of the characteristics of common passwords used today. Out of the 34,000 users who were hacked:



numbers only	1.3 percent
letters only	9.6 percent
alphanumeric	81 percent
non-alphanumeric	8.3 percent



1-4	0.82 percent
5	1.1 percent
6	15 percent
7	23 percent
8	25 percent
9	17 percent
10	13 percent
11	2.7 percent
12	0.93 percent
13-32	0.93 percent

It seems that the younger generation of computer users are beginning to see the importance of selecting longer passwords, but what are the most common passwords you might ask? Bruce Schneier mentions "the top 20 passwords are (in order): password1, abc123, myspace1, password, blink182, qwerty1, fuckyou, 123abc, baseball1, football1, 123456, soccer, monkey1, liverpool1, princess1, jordan23, slipknot1, superman1, iloveyou1 and monkey."

These common passwords give us an idea of what the typical user's creativity is towards forming a password. Given that these common passwords only make up less than .5% of the passwords used in the pool of stolen passwords, we cannot assume too much, but one can consider that the majority of users use passwords with the same elements as those mentioned above. The most important thing to keep in mind about passwords is that the typical user uses a password that will be easily remembered, thus one that almost always includes a real word of some sort.

To read more about the myspace password attack, see Bruce Schneier's blog article at:
http://www.schneier.com/blog/archives/2006/12/realworld_passw.html
The wikipedia article referenced in this article can be found at:
http://en.wikipedia.org/wiki/Salt_(cryptography)
To read more about dictionary and brute force benchmarks:
http://www.red-database-security.com/whitepaper/oracle_password_cracker.html
To read more about RainbowCrack and it's benchmarks:
http://project-rainbowcrack.com/index.htm#doc