How secure are password controlled self-decrypting files?

Summary

The use of passwords to control access to self decrypting executable files is not defensible as a security technique and should be avoided in favor of much stronger techniques such as public key cryptography. Organizations continuing to use password techniques are increasingly putting themselves at risk by using a technique that is more flawed than the cryptographic algorithm DES, already abandoned by industry. Also, since you cannot prove the source of these files, hackers and virus writers can send them and persuade you to run them. Solutions exist that solve these problems completely.

Introduction

One of the commonest security features you find supplied with file encryption products is the ability to send a protected file that the recipient can read providing they know a secret password for the file.

The argument goes that you (the recipient) does not need a copy of the security product to be able to read the contents of the file as long as you know the secret password that the sender has told you about by some means other than including it in an e-mail with the file.

In practice, organizations set up 'command line' interfaces so that they can process files for many recipients automatically because it is operationally simpler to process information routinely than to have a member of staff sorting out what is happening on an individual basis.

Analysis of weaknesses – user generated passwords

Now you don't need to be much of an expert to realize that the very first weakness in any password controlled system is the password itself. Just go and read a few articles on the subject of (supposedly) strong passwords to see how confused both the security industry and the practitioners are about using passwords.

If the passwords are chosen by human beings then they will have all the usual weaknesses – short, easy to guess, easy to remember, likely to be used again next time.

Another weakness in using passwords at all in this situation is the 'Internet effect.' This is where the attacker has infinite retries at finding the password. This happens because there is no mechanism that can realistically stop the attacker after so many attempts. What it means in reality is that passwords using less than 8 or 10 character positions are capable of being broken so quickly that the attacker can just use brute force the first time, and, by maintaining a dictionary of successful breaks, quickly build up a fast access means of breaking the code. (Most likely an attacker will already have all this sorted out on a 'script kiddy' basis.)

Analysis of weaknesses – computer generated passwords

In theory the computer system will be able to generate all manner of random passwords that should make the attacker's life difficult. After all, that's what session encryption keys are. However, we are dealing with the rich mixture of human beings and computers and behavioral characteristics.

The human recipient of regular information does not want to go to the trouble of having a password, especially a nice difficult one, sent by some other means than the protected file, every time a file comes along. This makes life personally difficult.

Now you can get the computer to generate passwords that are easier to remember when transcribing from one system to another, but that's still hard work.

There is also the problem of the product interface. Not so many products are set up to generate good long random passwords. It's much easier to leave it to the humans to sort out so at least they are to blame. And in most 'command line interfaces' the password has to be input on the command line because there is no mechanism in the product for sending the password to the intended recipient by some other route. Again, this is technically complex and therefore an SEP (someone else's problem).

So what actually happens more often than not is that the same password is used every time with the same recipient. This makes it easy for the recipient because then they are able to get other people to open the protected file if they are not there, without compromising their personal security mechanisms. However, once the password has escaped or been broken the whole protection scheme has failed.

In some ways this situation is worse than letting the user pick a password because it institutionalizes the fixing of passwords that probably don't even comply with the organization's policies for their own logon passwords. This is a strange situation.

But what to do?

The reason password controlled services were developed was really because the encryption product manufacturers insisted that the recipients had to buy full copies of their product if it was going to work properly, and password files were a "poor man's house." (In a limited number of cases it avoided problems with encryption export laws also.) No-one explained the real security implications.

Today, using public key cryptography, it is possible to provide very strong cryptography to protect information and prove its source. However, manufacturers, apart from ArticSoft, have not risen to the challenge of providing free readers so that recipients can make use of real security techniques instead of flawed ones.

The great cry is that people will not load programs or cope with sending identities before you can send them secrets. Apparently, suppliers other than ArticSoft and PGP don't believe that people can be trusted to manage their own identities. A digital signature that you have already registered as acceptable is significantly stronger than unproven executables that can compromise your systems.

Reality is that you never send secrets to people (or representatives of organizations) you don't have any knowledge of. To send e-mail you have to get their e-mail address(es). So why the big problem about getting their protection key (or telephone number, address, social security number, ….). The mystique of PKI, almost elevated to a religious rite of some kind, has been used to make a simple solution unnecessarily complex.

The introduction of PKI technologies can be achieved without compromising real security or commercial prudence. It does not require development or implementation of full scale PKI methods and techniques. A small change can convert systems from inadequate security to effective security.