A reader sent me the following question:

“What are the current, real world capabilities of the various forms of encryption to resist compromise?

Would like to see stats like: Data encrypted with 128 SSL is capable of withstanding brute force from 10,000 computers for 3 hrs (example only). Of course, the variables are innumerable, but something with a scientific basis and relative ease of understandability for the a client would be good.”

Brute-force cracking is like trying every possible key to unlock a door. Sometimes, the door is an access-control system and the key is a password; sometimes the door is a decryption algorithm and the key is a decryption key. I will continue referring to keys in the rest of the article.

The total number of keys is called the keyspace. The keyspace is a function of the key length and the number of possible values in each position. The number of possible values can be constrained by rules; for example, a key may be up to 10 alphanumeric characters in length but the first character may have to be a letter. Thus the first character would have 26 or 52 possible values (depending on whether case is recognized) but the other nine characters could have 36 or 62 values (because of the extra ten digits).

Another possible constraint affects repetition. For example, certain key rules may preclude more than one occurrence of a particular symbol. So for a ten-character key using only letters but not distinguishing between upper- and lowercase letters, the first character would have 26 possibilities, the second would have 25, the third would have 24, and so on.

The most important rule in calculating keyspace is that we multiply the number of possibilities in each position to arrive at the total. For example, if we have a four-digit personal identification number (PIN) and repetition is not permitted, then the total number of PINs is 10 x 9 x 8 x 7 = 5040.

If repetition is permitted, the calculation is easier; you raise the number of values per position (let’s call that V) to the power of the number of positions (let’s call that P). Thus we write that the keyspace (K) would be

\[ K = V^P \] or \[ K = V^{**P} \] (depending on how you like to write the exponentiation operator).

For example, if you had two positions which could have 10 digits in each with repetition allowed, then there would be \[ 10^2 = 100 \] values; i.e., the numbers from 00 to 99.

Keys can be measured in bits (0s or 1s) if we know something about how the values are
represented in computer storage; for example, an integer value is often stored as a 16-bit number on many computers. Thus in our formula, we can usually manage to set \( V = 2 \).

We can use our formula to state that a 128 bit key \( (P = 128) \) has \( K = 2^{128} \approx 3E38 \) (that stands for 3 x 10^38 different keys) and.

A shortcut is to remember the approximation that \( 2^n = 10^{(\log 2)n} = 10^{0.30103n} \) and so we can compute a rough estimate of the keyspace using simple multiplication if we don’t have a computer handy.

Now, how does brute-force cracking determine if a particular key is correct? That's really quite tricky and I won't go into it now. But let's just suppose that it's possible to program one or more computers to try out portions of the keyspace looking for the right key for a particular puzzle. How long does it take?

The maximum time dependence on these factors:

* the keyspace;
* the operations per second per processor;
* the number of processors working in parallel.

For example, if you had a trillion (1E12) operations per second for each of a trillion processors, trying all the keys in the 3E38 keyspace for a 128-bit key would take about 3E14 seconds. Reducing that to years gives about 10 million years.

I’m sure that anyone interested will be able to produce their own spreadsheet to play with this, but you can download my XLS file from


One other note: it’s not likely that the brute-force attack will have to search the entire key space. There is a principle in probability theory called the central limit theorem that tells us that on average, brute-force attacks will end up finding the right key after half the keyspace has been searched.

But 5 million years, 10 million years: who’s counting?

* * *

My thanks to Prof. Randall Nichols of The George Washington University and the University of Maryland University College for reviewing a draft of this article. He offered the following reference: “Dorothy Denning put out a wonderful table of keysize, operations required, differences in number of characters and work factor. Page 309 ff in her _Information Warfare and Security_ (1998; Addison-Wesley, ISBN 0-201-43303-6).”

For additional reading:


***

A Master’s degree in the management of information assurance in 18 months of online study from Norwich University – see <http://www3.norwich.edu/msia> for details.

M. E. Kabay, PhD, CISSP is Associate Professor in the Division of Business and Management at Norwich University in Northfield, VT. Mich can be reached by e-mail at <mailto:mkabay@norwich.edu>; Web site at <http://www.mekabay.com/index.htm>.

Copyright © 2004 M. E. Kabay. All rights reserved.

Permission is hereby granted to Network World to distribute this article at will, to post it without limit on any Web site, and to republish it in any way they see fit.