Another Promising Code Falls

A code that looked too good to be true has a fatal weakness and now can be broken in a few seconds

About 8 years ago, two Stanford University computer scientists proposed a new cryptographic code that seemed easy to implement and very difficult to break. Scientists at Mitre in Bedford, Massachusetts, have been experimenting with the code for their own corporate electronic mail and engineers at Hewlett-Packard have laid out the instructions for the code on a computer chip. But the code no longer looks so attractive. Donald Coppersmith of IBM's Thomas J. Watson Research Center in Yorktown Heights, New York, has now discovered how to break it. Coppersmith's results show, once again, that no matter how good a code looks, there is no way of telling whether there might be a clever way around it. In other words, the cryptographer's dream of a code that is "provably secure" remains elusive.

The code that Coppersmith broke is called "discrete exponentials" and was first suggested by Martin Hellman of Stanford and Whitfield Diffie, who is now at BNR in Palo Alto. The idea of this code, as with other so-called public key cryptosystems that these investigators proposed, is to use a mathematical procedure that is easy to compute but nearly impossible to reverse unless you have special information. You would publish the easy procedure and anyone who wanted to send you a message could encode with it. You would keep secret the information about how to reverse the procedure—and thus decode your messages. Anyone, then, could use your public encoding procedure and send you a message that only you could read.

In the discrete exponentials system, computer messages, which are strings of 0's and 1's, are encoded by raising them to a power. They are decoded by the reverse process—looking at a number that you know was formed by raising something to a power and learning what was raised to what power. The premise, then, is that raising to powers is easy and reversing that procedure—taking logs—is hard.

The system that Diffie and Hellman originally proposed was later modified so that the encoding and decoding were done in a particular mathematical system, called a Galois field, with 2^n elements. The only numbers allowed in this field are 0's and 1's and the field consists of polynomials of degree less than n. The number n determines how the code is constructed and is called the key size. The larger n is, the harder it is to break the code but the more difficult it is to encode and decode. The only restriction on n is that $2^n - 1$ either be a prime number or have a large prime factor. The experimental Hewlett-Packard chip uses an n of 127 and $2^{127} - 1$ is a prime.

The advantage of working in such a Galois field, Coopersmith says, is that it is custom-made for computers that calculate everything in terms of 0's and 1's. By working in such a mathematical system, engineers found that the discrete exponentials system was simple and fast enough to be of some use.

Coppersmith's results show, once again, that no matter how good a code looks, there is no way of telling whether there might be a clever way around it.

But it was the very advantages of the Galois field that led to the downfall of the code. In that field, squaring is a simple operation. Because only 1's and 0's are used, $1 + 1 = 0$, not 2. For that reason, the expression $(x + y)^2$ does not equal $x^2 + 2xy + y^2$. Instead, it equals $x^2 + y^2$. Coppersmith says that this quirk led him to crack the code. "I was able to take advantage of that information," he says. He was influenced by a recent paper by Ian Blake, R. Fuji-Hara, R. C. Mullin, and S. A. Van Stone of the University of Waterloo in which these investigators broke the code for a key size of 127. "I pushed the idea farther and was able to get a faster attack against this same field," Coppersmith says. "My method is about 20 times faster than theirs."

To compute a key—and thus break the code—when the key size is 127 bits, Coppersmith goes through two procedures. The first he calls precomputation. "You do it once and for all. You build a huge database to solve equations," he says. The precomputation stage takes less than 1 hour on a mainframe computer. The next phase is to find particular keys, and this takes only a few seconds with the new method. What if the key size were larger? "The next choice of keys that people have been talking about is 241 bits. This would take several weeks or a month on a computer for the precomputation," Coppersmith says, "and a few hours of computer time to find specific keys. That's a lot of time if there's nothing at stake. But if you can use the information in the military, that's not a lot."

Brian Schanning of the Mitre Corporation says that Coppersmith's work probably makes the discrete exponentials code unsatisfactory for practical use. At Mitre, computer scientists were using the code to distribute encoding keys for a more traditional cryptographic system, the DES, that was used to scramble messages sent within the company. The key size for the discrete exponentials code is 127 bits and it takes, says Schanning, less than 10 seconds to exchange DES keys with it. "If we had to go to 241 bits, it would take minutes and that would be an inconvenient delay. If we start talking of 1000 bits, it would probably take hours." In addition, says Schanning, if you try to put the code on a computer chip, you start to run into problems if the key size is greater than 127 bits. The chips can only hold about 200 bits.

Joel Birnbaum, director of computer research at Hewlett-Packard, says, "Like everyone else in the industry, we've been looking at public key cryptography. Obviously, if Coppersmith's result works out, we'll have to take it into consideration." Asked if he believes Coppersmith's result is correct, Birnbaum says that he has no doubt that it is.

"What this seems to say," Schanning concludes, "is that you have to increase the key size so much that there is no advantage in using the code. You don't get something for nothing." But he says he is not surprised by Coppersmith's result. "We've been nervous about the code for several years. Any time you get something that seems so easy and that seems to have no disadvantages, you get nervous."—GINA KOLATA