Securing the cloud

Massachusetts Institute of Technology

Homomorphic encryption is one of the most exciting new research topics in cryptography, which promises to make cloud computing perfectly secure. With it, a Web user would send encrypted data to a server in the cloud, which would process it without decrypting it and send back a still-encrypted result.

Sometimes, however, the server needs to know something about the data it's handling. Otherwise, some computational tasks become prohibitively time consuming — if not outright impossible.

Suppose, for instance, that the task you've outsourced to the cloud is to search a huge encrypted database for the handful of records that match an encrypted search term. Homomorphic encryption [1] ensures that the server has no idea what the search term is or which records match it. As a consequence, however, it has no choice but to send back information on every record in the database. The user's computer can decrypt that information to see which records matched and which didn't, but then it's assuming much of the computational burden that it was trying to offload to the cloud in the first place.

Last week, at the Association for Computing Machinery's 45th Symposium on the Theory of Computing — the premier conference in theoretical computer science — researchers from MIT's Computer Science and Artificial Intelligence Laboratory, together with colleagues at the University of Toronto and Microsoft Research, presented a new encryption scheme that solves this problem. Known as a functional-encryption scheme, it allows the cloud server to run a single, specified computation on the homomorphically encrypted result — asking, say, "Is this record a match?" or "Is this email spam?" — without being able to extract any other information about it.

"This is a very, very general paradigm," says Shafi Goldwasser, the RSA Professor of Electrical Engineering and Computer Science, one of the paper's co-authors and, together with her fellow MIT professor Silvio Micali, the most recent recipient of the Turing Award, the highest award in computer science. "Say we're talking about the surveillance cameras of the future, which come up with encrypted images. Why would we want to do that? It's a question of liberty versus safety. If you're looking for a suspect, you might be interested in doing some computations on an encrypted image, to match to the subject. Another possibility would be a medical database, where all the information is encrypted and … someone [runs] a drug study on those blood samples — but just that drug study, nothing else. Our result is in some sense the first result showing that you can do this very generally."

Joining Goldwasser on the paper are Raluca Ada Popa, a graduate student in the Department of Electrical Engineering and Computer Science, her advisor, associate professor Nickolai Zeldovich, and Yael Kalai of Microsoft Research and Vinod Vaikuntanathan of the University of Toronto, both of whom did their graduate work

at MIT with Goldwasser.

Near misses

The researchers built their functional-encryption scheme by fitting together several existing schemes, each of which has vital attributes of functional encryption, but none of which is entirely sufficient in itself. The first of those is homomorphic encryption.

Another is what's known as a garbled circuit, a technique developed in the mid-1980s and widely used in cryptography. A garbled circuit lets a user decrypt the result of one cryptographically protected operation on one cryptographically protected data item — say, "Is this record a match?" The problem is that, if the garbled circuit is used on a second data item — "How about this record?" — the security breaks.

Moreover, a garbled circuit is a so-called private-key system, in which only the holder of a secret cryptographic key can encrypt data. Homomorphic encryption, by contrast, is intended as a public-key system — like most of the encryption schemes used to protect financial transactions on the Web. With public-key encryption, anyone can encrypt a message using a key that's published online, but only the holder of the secret key can decrypt it.

The final component technique is called attribute-based encryption. Attribute-based encryption is a public-key system, and it's reusable. But unlike garbled circuits and homomorphic encryption, it can't reveal the output of a function without revealing the input, too.

The new system begins with homomorphic encryption and embeds the decryption algorithm in a garbled circuit. The key to the garbled circuit, in turn, is protected by attribute-based encryption. In some sense, the garbled circuit can, like all garbled circuits, be used only once. But the encryption schemes are layered in such a way that one use grants the server access to a general function rather than a single value. It can thus ask, of every record in a database, "Is this a match?"

Zeldovich points out that since the scheme relies on homomorphic encryption, it shares the major drawback of existing homomorphic schemes: They're still too computationally intensive to be practical. On the other hand, he says, "It's so new, there are so many things that haven't been explored — like, 'How do you really implement this correctly?' 'What are the right mathematical constructions?' 'What are the right parameter settings?'" And, Popa adds, in the four years since the invention of the first fully homomorphic encryption scheme, "People have been shaving off many orders of magnitude in performance improvements."

Besides, even a currently impractical functional-encryption scheme is still a breakthrough. "Before, we didn't even know if this was possible," Popa says.

Ran Canetti, a professor of computer science at Boston University, corroborates that assessment. "It's an extremely surprising result," he says. "I myself worked on this

problem for a while, and I had no idea how to do it. So I was wowed. And it really opens up the door to many other applications."

One of those applications, Canetti says, is what's known as program obfuscation, or disguising the operational details of a computer program so that it can't be reverse-engineered. "Not obfuscating the way that people are doing it now, which is just scrambling up programs and hoping nobody will understand, and eventually, these are broken," Canetti says, "but really obfuscating so that it's cryptographically secure."

Canetti acknowledges that the researchers' scheme won't be deployed tomorrow. But "I'm sure it's going to lead to more stuff," he says. "It's an enabler, and people will be building on it."

Links:
[1] http://www2.technologyreview.com/article/423683/homomorphic-encryption/