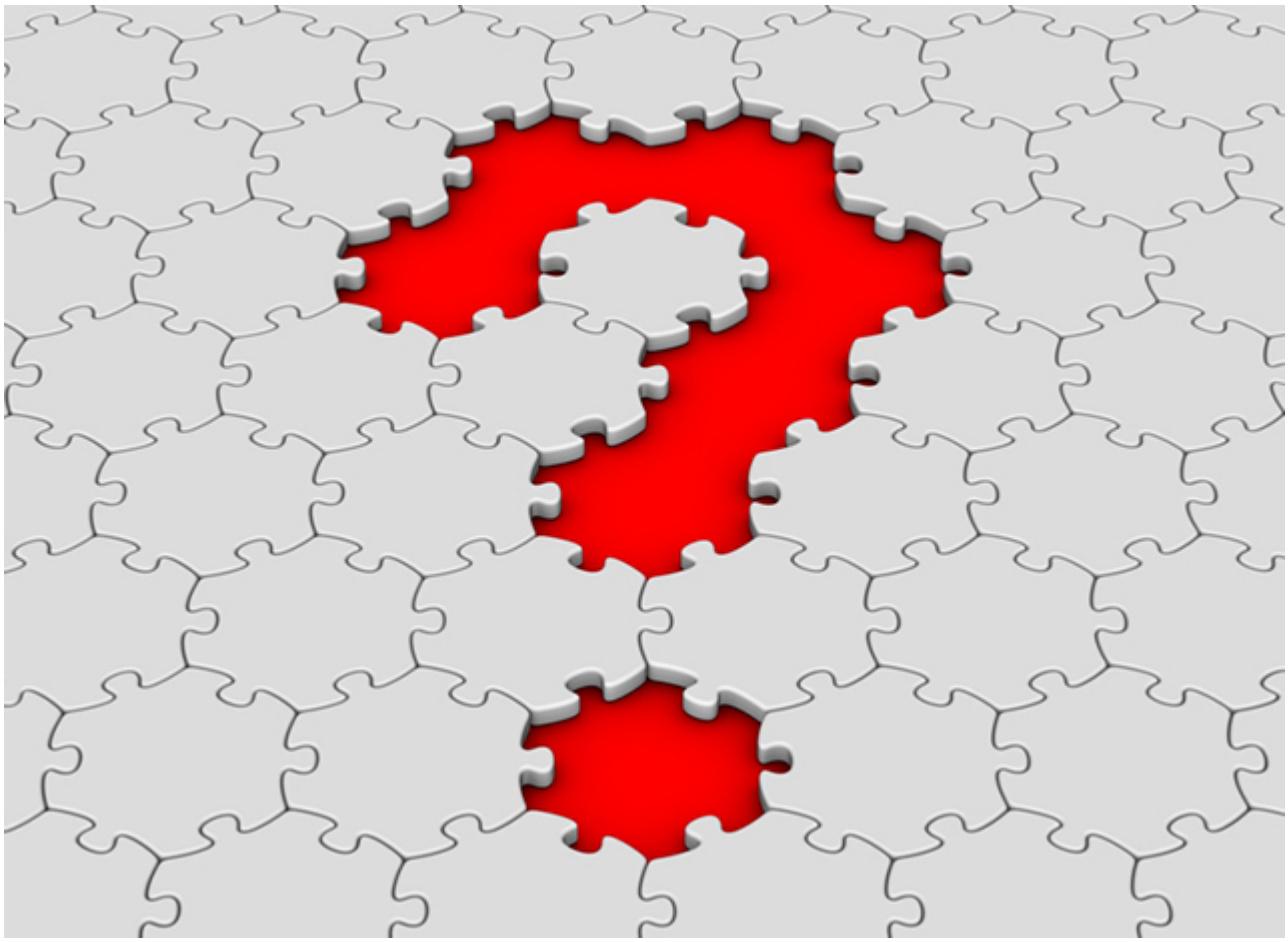
Perfecting the Art of Sensible Nonsense

By Erica Klarreich



A new cryptographic scheme obfuscates computer programs by transforming them into something akin to a jigsaw puzzle in which random elements make each individual piece look meaningless.

As a graduate student at the Massachusetts Institute of Technology in 1996, Amit Sahai was fascinated by the strange notion of a “zero-knowledge” proof, a type of mathematical protocol for convincing someone that something is true without revealing any details of why it is true. As Sahai mulled over this counterintuitive concept, it led him to consider an even more daring notion: What if it were possible to mask the inner workings not just of a proof, but of a computer program, so that people could use the program without being able to figure out how it worked?

The idea of “obfuscating” a program had been around for decades, but no one had ever developed a rigorous mathematical framework for the concept, let alone created an unassailable obfuscation scheme. Over the years, commercial software companies have engineered various techniques for

garbling a computer program so that it will be harder to understand while still performing the same function. But hackers have defeated every attempt. At best, these commercial obfuscators offer a "speed bump," said Sahai, now a computer science professor at the University of California, Los Angeles. "An attacker might need a few days to unlock the secrets hidden in your software, instead of a few minutes."

Secure program obfuscation would be useful for many applications, such as protecting software patches, obscuring the workings of the chips that read encrypted DVDs, or encrypting the software controlling military drones. More futuristically, it would allow people to create autonomous virtual agents that they could send out into the computing "cloud" to act on their behalf. If, for example, you were heading to a remote cabin in the woods for a vacation, you could create and then obfuscate a computer program that would inform your boss about emails you received from an important client, or alert your sister if your bank balance dropped too low. Your passwords and other secrets inside the program would be safe.

Amit Sahai, a computer science professor at the University of California, Los Angeles, and his collaborators have developed an "indistinguishability" obfuscator that many see as a watershed moment for cryptography.

"You could send that agent into the computing wild, including onto untrusted computers," Sahai said. "It could be captured by the enemy, interrogated, and disassembled, but it couldn't be forced to reveal your secrets."

As Sahai pondered program obfuscation, however, he and several colleagues quickly realized that its potential far surpassed any specific applications. If a program obfuscator could be created, it could solve many of the problems that have driven cryptography for the past 40 years — problems about how to conduct secure interactions with people at, say, the other end of an Internet connection, whom you may not know or trust.

"A program obfuscator would be a powerful tool for finding plausible constructions for just about any cryptographic task you could conceive of," said Yuval Ishai, of the Technion in Haifa, Israel.

Precisely because of obfuscation's power, many computer scientists, including Sahai and his colleagues, thought it was impossible. "We were convinced it was too powerful to exist," he said.

Their earliest research findings seemed to confirm this, showing that the most natural form of obfuscation is indeed impossible to achieve for all programs.

Then, on July 20, 2013, Sahai and five co-authors posted a paper on the Cryptology ePrint Archive demonstrating a candidate protocol for a kind of obfuscation known as "indistinguishability obfuscation." Two days later, Sahai and one of his co-authors, Brent Waters, of the University of Texas, Austin, posted a second paper that suggested, together with the first paper, that this somewhat arcane form of obfuscation may possess much of the power cryptographers have dreamed of.

"This is the first serious positive result" when it comes to trying to find a universal obfuscator, said Boaz Barak, of Microsoft Research in Cambridge, Mass. "The cryptography community is very excited." In the six months since the original paper was posted, more papers have appeared on the ePrint archive with "obfuscation" in the title than in the previous 17 years.

However, the new obfuscation scheme is far from ready for commercial applications. The technique turns short, simple programs into giant, unwieldy albatrosses. And the scheme's security rests on a new mathematical approach that has not yet been thoroughly vetted by the cryptography community. It has, however, already withstood the first attempts to break it.

Researchers are hailing the new work as a watershed moment for cryptography. For many cryptographers, the conversation has shifted from whether obfuscation is possible to how to achieve it.

"Six or seven years ago, you could have looked at this question and wondered if we'll ever know the answer," said Leonard Schulman, of the California Institute of Technology in Pasadena. "The fact that there's now a plausible construction is huge."

Too Powerful to Exist

When Sahai started thinking about obfuscation 17 years ago, the first task was simply to define it. After all, users can always learn something about a garbled version of a program simply by feeding it inputs and seeing what comes out.

The most natural, and also the strongest, definition was the idea of a "black box" obfuscator, which would jumble a program so thoroughly that a person with the best available computational resources could figure out nothing at all about it, except for what might be gleaned from inputs and outputs. You could not figure out the value of a password hidden inside the software, unless that password was one of the program's outputs, nor could you reassemble parts of the program to compute anything meaningful other than what the program was originally designed to compute.

A black box obfuscator, if it existed, would be immensely powerful, providing instant solutions to many cryptography problems that took decades to figure out or in some cases remain unsolved. Take, for example, public key encryption, whose development in the 1970s paved the way for Internet commerce. Prior to its creation, two people who wanted to communicate secretly had to meet in advance to choose an encryption scheme and share a secret key for encoding and decoding messages. Public key encryption allows you to announce a key to the entire world that permits people you've never met to send you messages that only you can decrypt. The innovation so revolutionized cryptography that its early developers have been recognized with one award after another.

But if you have a black box obfuscator, creating a public key encryption protocol becomes a simple

matter of choosing your favorite secret-key encryption scheme, expressing its workings as a computer program, obfuscating the program, and making the obfuscated version widely available. Anyone can then use it to encrypt a message to send to you, but no one can tease the decryption key out of the obfuscated software.

Similarly, a black box obfuscator would provide a way to instantly convert any private cryptography scheme to a public one that could be performed over the Internet by strangers. In a sense, obfuscation is the key to all cryptographies.

“Modern cryptography is about the transition from private to public,” Sahai said. “Obfuscation gives you a remarkable ability to move between these two worlds that, for decades, we thought of as fundamentally different.”

The power of universal black box obfuscation seemed too good to be true, and it was. In 2001, Sahai, Barak and several co-authors showed that it is impossible. Some programs, the researchers demonstrated, are like people who insist on sharing their most private moments on Twitter or Facebook — they are so determined to reveal their secrets that no obfuscator can hide them.

Still, Sahai couldn’t stop thinking about the problem. The computer programs the team had devised, which spilled their guts so insistently, were contrived objects unlike any real-world program. Might some weaker notion than black box obfuscation protect the secrets of programs that hadn’t been specifically constructed to resist obfuscation? And if so, just how powerful would such an idea be?

Jigsaw Puzzle Programs

Sahai, Barak and their colleagues had put forward one definition of a weaker kind of obfuscation in their 2001 paper, a rather esoteric concept called indistinguishability obfuscation. A program-garbling procedure qualifies as an indistinguishability obfuscator if, whenever two programs that do exactly the same thing pass through the obfuscator, no one is able to tell which garbled program came from which original.

There’s no obvious reason why this concept should be particularly useful. After all, even if no one can distinguish the sources of the two garbled programs, it might still be possible to glean important secrets — a decryption key, or classified instructions — from looking at the garbled software.

Craig Gentry, of the IBM Thomas J. Watson Research Center, worked with Sahai on the new protocol.

“It’s a very weak notion of obfuscation,” said Craig Gentry, of the IBM Thomas J. Watson Research Center in Yorktown Heights, N.Y.

But in 2007, Shafi Goldwasser of MIT and Guy Rothblum of Microsoft Research Silicon Valley in Mountain View, Calif., showed that an indistinguishability obfuscator, if it could be built, would be the best possible obfuscator. The idea is that if some other obfuscator were the best, you could use it to garble the program and then put both the original program and the garbled version through the indistinguishability obfuscator for an additional layer of distortion. Someone looking at the resulting two programs wouldn’t be able to tell which one came from the original program, meaning that the indistinguishability obfuscator was at least as good at hiding the program’s secrets as that other, “best” obfuscator.

Goldwasser and Rothblum’s result meant that indistinguishability obfuscation was the best hope for protecting all of a computer program’s secrets that are protectable. But no one knew how to build such an obfuscator, or even knew which of a program’s secrets are protectable. Would an indistinguishability obfuscator, Sahai wondered, protect the secrets people really cared about?

For Sahai, the decade leading up to the new finding was marked by dead ends and incremental results. “There was a long period of banging my head against the wall and hoping a dent would form,” he said. “We were all very pessimistic, but it was such a beautiful problem that I was completely hooked.”

In the fall of 2012, Sahai started collaborating with Gentry and Waters, along with Sanjam Garg and

Shai Halevi of the IBM Thomas J. Watson Research Center and Mariana Raykova of SRI International in Menlo Park, Calif., on a problem called functional encryption, which deals with how to give different people particular levels of access to encrypted data. After what Sahai called "an incredibly intense period" of putting forward ideas, breaking them, and returning to the drawing board, in the spring of 2013 the team came up with a complicated solution to the problem. "What we had was a mess, with so many moving parts and subscripts of subscripts, but it was the first thing we couldn't break," Sahai recalled.

As the researchers tried to simplify their construction, they discovered that it went much further than anticipated: It presented a way to perform indistinguishability obfuscation on all computer programs.

"That's a moment I'll never forget," Sahai said.

Sahai and Waters proceeded to show that their indistinguishability obfuscator seems to offer much of the all-encompassing cryptographic protection that a black box obfuscator would offer. It can be used, for example, to create public key encryption, digital signatures (which enable a website to convince its visitors that it is legitimate) and a laundry list of other fundamental cryptographic protocols, including two major ones that were previously unsolved, functional encryption and deniable encryption.

The team's obfuscator works by transforming a computer program into what Sahai calls a "multilinear jigsaw puzzle." Each piece of the program gets obfuscated by mixing in random elements that are carefully chosen so that if you run the garbled program in the intended way, the randomness cancels out and the pieces fit together to compute the correct output. But if you try to do anything else with the program, the randomness makes each individual puzzle piece look meaningless.

This obfuscation scheme is unbreakable, the team showed, provided that a certain newfangled problem about lattices is as hard to solve as the team thinks it is. Time will tell if this assumption is warranted, but the scheme has already resisted several attempts to crack it, and Sahai, Barak and Garg, together with Yael Tauman Kalai of Microsoft Research New England and Omer Paneth of Boston University, have proved that the most natural types of attacks on the system are guaranteed to fail. And the hard lattice problem, though new, is closely related to a family of hard problems that have stood up to testing and are used in practical encryption schemes.

Sahai's hope is that not only will this hard problem stand the test of time, but computer scientists will figure out ways to base the obfuscation scheme on more conventional cryptographic assumptions. Cryptographers are already jumping on the indistinguishability obfuscation bandwagon, searching for ways to make the scheme more efficient, bolster its security assumptions, and further elucidate just which secrets it can protect.

The proposed obfuscator has already produced a sea change in many cryptographers' views of program obfuscation. "It seems that the problem is not impossible," said Daniele Micciancio, of the University of California, San Diego.

Beyond the immediate task of refining the team's obfuscation protocol lies a deeper question: If the problem of obfuscation has been solved, what remains for cryptographers?

"What is the next major cryptographic frontier that is not solved, at least in principle, by obfuscation?" Sahai said. "That's one of the big questions for our field."

This article was reprinted on Wired.com.