Proof of randomness builds future of digital security

December 22 2017

Credit: CC0 Public Domain

In an effort to block emerging threats to online security, researchers at Princeton University have developed a method to verify the strength of random number generators that form the basis of most encryption systems.

Nearly all secure online traffic—from shopping to banking to communications—relies on a technique of randomly generating a

number that serves as a key to unlock encrypted communication. The problem is that small programming errors can make these systems vulnerable, and those vulnerabilities can often be very difficult to detect.

"Whenever you connect up to Amazon to give them your credit card number, whenever you log in somewhere through a secure connection, you're depending on randomly generated cryptographic keys," said Andrew Appel, the Eugene Higgins Professor of Computer Science at Princeton and leader of the research team. "And if the adversary, the spy who is trying to read your messages or impersonate you, could guess what random number your computer was using, then it could know what key you're going to be using and it could impersonate your traffic and read your messages."

In a paper presented to the Association for Computing Machinery 2017 Conference on Computer and Communications Security on Nov. 2, the researchers said it may be impossible to tell whether a number generator is compromised without examining the generators' source code (and without proper methods, difficult to guarantee security even with access to the code). The programs, called Deterministic Random Bit Generators or DRBGs, are tested typically by analyzing their outputs, either statistically or by using a set of tests to check the results. But the researchers said these methods cannot guarantee the generators' proper function.

"Despite the importance of DRBGs, their development has not received the scrutiny it deserves," the researchers write in their article.

Although often called random number generators, these programs are actually pseudorandom number generators. The programs are algorithms that produce numbers that seem to be random and can practically work as random numbers for many applications. The DRBGs use a variety of methods to create a truly random number called a seed. The program

then mathematically expands this seed into a much longer number. The long number is not actually random, but it must appear random enough that an adversary (who does not know the seed) can't predict the output.

The researchers said flaws in number generators, or their implementation, have caused several security breaches in the past few years. "Many security researchers have found these bugs in random number generators," said Katherine Ye of the Class of 2016, a member of the research team who is now a graduate student at Carnegie Mellon University. She said that, in some cases, the bugs were accidental and, in others, they were deliberately added or exploited to breach security.

Ye began working on methods to check number generators as part of her senior thesis at Princeton. She and her co-authors wrote proofs in several existing frameworks for verifying programs, including Appel's Verified Software Toolchain, which includes a logic for reasoning about programs written in the C language.

It was time-consuming and difficult work, and many proofs had to be done manually. Working with colleagues at Princeton, Johns Hopkins University and Oracle, Ye, Appel and their collaborators examined a widely used pseudorandom number generator called HMAC-DRBG. They produced a comprehensive and machine-checked proof that HMAC-DRBG is indeed secure, meaning that its output is sufficiently difficult to distinguish from truly random output.

Ye said the recent results show that it is practical to apply secure tests to other generators, although doing so would require new sets of proofs. (The researchers said the National Institute of Standards and Technology has approved three DRBGs for use by the U.S. government.)

Eugene Spafford, professor and executive director emeritus of Purdue University's Center for Education and Research in Information

Assurance and Security, said the research is "an advancement, without a doubt."

The mathematical assurance of the proof provides a "very high level of assurance" of the security of the number generator, he said.

"That means we can use it with great confidence that an observer isn't going to be able to break it and … interfere with our communications," Spafford said.

Spafford agreed that it is feasible, with more engineering work, to adapt the Princeton team's methods to other number generators used for critical security applications. He noted that the checks would not necessarily be needed for generators used for other types of applications. "If all I'm using a random number generator for is to run simulations, I may not have to prove it's unbreakable at all because they're just simulations," he said.

Ye believes that expanding the research to other number generators is an important step.

"I think our work could be more impactful if someone extended it to apply to DRBGs that are even more widely used than HMAC-DRBG," she said.

In the decades to come, new cryptographic tools using number generators will be developed, and as those tools are introduced, there will be debate over how secure they really are, Appel said.

Machine-checked proofs may help with that process, Appel added.

"It's a very nice result," Spafford said. "Like a lot of other research, it may not directly apply to your life and mine at the moment, but it's

building up a set of results that could [lead to] very important results in the future."

More information: Katherine Q. Ye et al. Verified Correctness and Security of mbedTLS HMAC-DRBG, *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security - CCS '17* (2017). DOI: 10.1145/3133956.3133974

Provided by Princeton University