Improving web security without sacrificing performance

September 7 2017, by Daniel Tkacik

Credit: Carnegie Mellon University Electrical and Computer Engineering

Chances are, you're reading this article on a web browser that uses HTTPS, the protocol over which data is sent between a web browser and the website users are connected to. In fact, nearly half of all web traffic passes through HTTPS. Despite the "S" for security in "HTTPS," this protocol is far from perfectly secure.

"The HTTPS ecosystem has seen a long and somewhat depressing series of bugs," says Bryan Parno, an associate professor of Computer Science and Electrical & Computer Engineering. "It's a continuous cycle: bug, panic, fix. Bug, panic, fix."

The problem, Parno says, is that software today comes with few, if any, security guarantees. Traditionally, vendors become aware of vulnerabilities after an attack occurs, and then issue a patch that fixes that particular attack.

In a paper presented at the USENIX Security Symposium in Vancouver, Parno and a team of researchers demonstrated a new programming tool called "Vale" that enables high-performance cryptographic code to be verifiably correct and secure. The team demonstrated their verification approach on a several cryptographic components of the HTTPS ecosystem.

"Ours was one of the first demonstrations of verified code performing just as fast, or faster, than unverified code," Parno says. "By verified, I mean you actually have a formal mathematical proof that all of the code that makes up these HTTPS components actually meets some high-level security specification."

One of the main reasons websites have been resistant to using verified code in HTTPS is that, until now, most verified code has performed significantly slower than unverified code. Slower data transfers between the website and the user translate into a lower quality experience.

In addition to proving the cryptographic components were correct, the team demonstrated the success of their verification system at proving resilience to two of the most popular types of side-channel attacks: timing attacks – in which an adversary uses the time delay between requesting and receiving data to deduce information about the

encryption key – and memory-access attacks – in which an adversary monitors a victim's memory accesses in a shared computing environment to deduce an encryption key.

But Parno warned: while it's close, their verification system is not HTTPS' bullet-proof vest. The verification is highly dependent on the security specification the developers feed it.

"You're only as good as your specification," Parno says. "Things that you failed to capture in your specification may still be vulnerable to attack."

Other authors on the study included Microsoft researchers Barry Bond, Chris Hawblitzel, K. Rustan M. Leino, Jacob R. Lorch, and Srinath Setty, Manos Kapritsos of the University of Michigan, Ashay Rane of the University of Texas at Austin, and Laure Thompson of Cornell University.

Provided by Carnegie Mellon University Electrical and Computer Engineering