SearchSecurity.com

Experts question customized TLS implementation after Amazon s2n flaw

By null

A report recently detailed a timing attack vulnerability in Amazon's Signal to Noise (s2n) implementation of the TLS protocol, and although Amazon fixed the vulnerability quickly, experts were still wary of customized TLS implementations.

Lucky Microseconds: A Timing Attack on Amazon's s2n Implementation of TLS described how s2n was vulnerable to the Lucky 13 timing attack which could be extended, in some cases, to complete plaintext recovery of HTTP session cookies and user credentials by running malicious JavaScript in a victim's browser and performing a man-in-the-middle (MitM) attack.

When Amazon first announced s2n, it cited the slimmed down amount of code needed for the TLS implementation as a major benefit, using only around 6,000 lines of code compared to approximately 70,000 lines of code using OpenSSL to implement TLS. Because of this simplification, Amazon claimed it would make the code easier to review when a vulnerability arose.

The turnaround time on this latest fix seems to support that claim. The *Lucky Microseconds* report noted that the vulnerability was disclosed to Amazon on July 5, 2015 and Amazon's blog post on the flaw said a patched version of s2n was available on GitHub as of July 11th.

According to Amazon, the s2n code went through three external security evaluations before its initial release and did include mitigations against Lucky 13 attacks, but only made "attacks tens of millions of times more difficult for attackers, rather than the trillions of times more difficult that [Amazon] had intended." Amazon also contends that the vulnerability did not impact its customers and was not likely to be exploited in the real world given the mitigations in place and because "the versions of s2n concerned have never been used in production."

Robert Hansen, vice president of WhiteHat Labs at WhiteHat Security, said the mechanics and underpinnings of SSL and TLS are so incredibly complex that flaws like this aren't uncommon.

"Timing attacks are quite common in many different applications -- especially in crypto," Hansen said. "There have been dozens of timing attacks against SSL/TLS in the past, so this shouldn't be considered a rare occurrence or unique to them. The bug itself is unique, but historically there have been many timing attacks against SSL/TLS implementations -- especially in the browsers."

Authors of the *Lucky* Microseconds report, Martin Albrecht and Kenneth Paterson from the Information Security at Group Royal Holloway, University of London, noted the vulnerability in s2n would not affect "most modern browsers" because they prefer TLS 1.2 AEAD cipher suites by default. In this regard, "modern browsers" appear to include anything released after early 2014 as being immune to the vulnerability under default settings. According to Hansen, just because Amazon s2n can be implemented with fewer lines of code doesn't necessarily reduce complexity.

"Nothing about SSL/TLS implementation is easy and I can't fathom this would make things easier," Hansen said. "It's more compact, and possibly easier to audit from a code-review perspective -- however, the complexity is still present. The fact that features are omitted is in and of itself something that has to be thoroughly tested. Less complexity might actually make things worse -- it totally depends on the details, and that's not an easy thing to diagnose, even for professionals."

According to Amazon, the primary goal of s2n was to implement the TLS/SSL protocols in a secure way and include risk reduction via reduced complexity.

"Besides hard-to-attack timing issues, our observation is that other software coding errors can lead to much more practical forms of attacks including remote execution, such as the ShellShock vulnerability, or memory disclosure such as the HeartBleed vulnerability in OpenSSL," wrote Colm MacCarthaigh, principal engineer for Amazon Web Services. "In short: these latter kinds of vulnerabilities, due to small coding errors, can be catastrophic and rank as higher impact threats. We must carefully balance the risk of additional complexity and code against the benefit. This leads us to be extremely conservative in adding code to s2n and to prioritize readability, auditability and simplicity in our implementation."

Andrew Lewman, vice president of data development at Norse Corp., warned that complexity isn't in the lines of code, but rather in the "correct usage of the cryptographic primitives."

"OpenSSL accounts for 17 years of history, backwards compatibility, and platform diversity. TLS is just one protocol in the OpenSSL library (libssl), the other is more general cryptographic primitives (libcrypto)," Lewman said. "Amazon implemented the current protocol, TLS, in its library s2n. This is the equivalent of just libssl from the OpenSSL library. The better comparison is lines of code between libssl and s2n. The challenge is in the use of the cryptographic primitives to make TLS work and interoperate with other devices and to use secure coding techniques."

03 Dec 2015